Global Information Assurance Certification Paper

GIAC Certified Intrusion Analyst

Practical Assignment

Version 4,1 (revised September 22, 2004)

By Brett Hutley

Submitted November 29th, 2004

Executive Summary

Five consecutive days of log file data were used in order to conduct the site audit. During the course of performing this analysis I have identified a number of vulnerabilities and potentially compromised machines.

■ Findings

During the course of this analysis I found a large number of internal hosts that have been actively compromised with trojan software, particularly the Adore worm, the mstream denial of service software, and the XDCC IRC file serving software. Not only are these machines compromised, but there are people actively exploiting the trojan software in order to distribute files and attack other machines. This should be of concern, not only because of the risk of losing valuable data, bandwidth and intellectual property, but also because of the legal liabilities involved.

■ Areas to Review

Firstly, the machines that have been flagged as probably compromised should be handled. Some of these machines may need to imaged, and then reformatted and rebuilt. A list of the top 30 machines believed to be compromised is included in the Appendix.

A number of internal machines seem to be under threat because they are not up-to-date with the latest patches and do not have adequate anti-virus software installed. The Nimda worm has infected a number of Windows hosts and the Adore worm has infected a number of Linux machines. This malware would not be so much of a problem if the machines internally are patched regularly. All Windows machines in particular should have anti-virus software installed. It is thoroughly recommended that a practice of "defense in depth" is followed.

There seems to be an alarmingly high amount of traffic passing into the internal network. Connections are being established with internal machines from outside the network. An urgent review of your firewall rules are is in order. What machines from outside your network are allowed to connect to your machines internally and why? There are a large number of scans and exploits penetrating your border gateway.

Also, the log files being analyzed seem to have been generated using a pre-1.8 version of Snort, running a customized rule-set. The version of Snort should probably be upgraded to the latest stable release, and the Snort ruleset should be reviewed. The log files generated by Snort were corrupt in a number of places. The reason for the corruption should be investigated and corrected.

■ Future Directions

Some areas to look into in this environment are the promising technology of passive vulnerability monitoring systems, such as Tenable Security's product NeVO. I believe the company SourceFire is also developing a similar product. These systems should be helpful in alerting for unpatched software, violations of policy, and anomolous behaviour.

■ Report Layout

This analysis is presented in several sections. Firstly, I describe the data used in the analysis. I then go on to examine the alert data picking three alerts to examine in detail. Finally, I will finish up by looking at defensive recommendations, both for the short term, and for the longer term.

Detailed Analysis

Data Used

The files used to perform the audit were:

Date	Alert File	Scan File	Out - of - Spec File
2004 - 04 - 07	alert040407	scans040407	oos_report _ 040407
2004 - 04 - 08	alert040408	scans040408	oos_report _ 040408
2004 - 04 - 09	alert040409	scans040409	oos_report _ 040409
2004 - 04 - 10	alert040410	scans040410	oos_report _ 040410
2004 - 04 - 11	alert040411	scans040411	oos_report _ 040411

All files are generated by the Snort intrusion detection system. The Snort sensor(s) seems to be located between the firewall and the internal network. The rule set used is not known. The alert files are Snort alert files running in “fast” mode, which logs each alert in one line but does not include the packet headers.

There are three different types of files used in the analysis. The "alert" files are the alert events generated from the Snort rule-set. The "scan" files record events when scanning activity is observed on the network. The "out-of-spec" files record packets that do not correspond to the TCP/IP standards.

Across the five days of log file data, there are 7956 unique IP addresses out of which 424 are internal IP addresses and 7532 external IP addresses.

■ Corruptions

These files contained corruptions in quite a large number of places. The corruption could be caused by a number of reasons, but what is interesting is that the time stamps for logged events seem to jump back in time after some of the corruptions. I have included an example of this below. This corruption could be due to having multiple Snort processes running at the same time, but logging to the same log files.

■ An example of log file corruption

In the example below, the log file corruption is highlighted in bold. Note how the event timestamp jumps from 8:21 in the morning to 8:02.

```
04/08-08:02:05.085822  [**] MY.NET.30.3 activity [**] 66.149.110.200:1036 ->
MY.NET.30.3:524
04/08-08:21:20.054287  [**] spp_portscan: portscan status from MY.NET.1.4: 8
connections across 8 hosts: TCP(0), UDP(8) [**]
04/08-08:02:05.155074  [**] MY.NET.30.3 activity [**] 66.149.110.20004/08-
08:21:20.073167  [**] spp_portscan: portscan status from MY.NET.111.51: 20 con-
nections across 20 hosts: TCP(20), UDP(0) [**]
:1036 -> MY.NET.30.3:524
04/08-08:02:05.436618  [**] MY.NET.30.3 activity [**] 66.149.110.200:1036 ->
MY.NET.30.3:524
```

Implied Roles of Various Hosts on the Network

By using the information contained in the log files, I have postulated the listed roles for the following hosts:

Host	Service / Role
MY.NET .1 .3	Primary DNS
MY.NET .1 .4	DNS
MY.NET .1 .5	DNS
MY.NET .150 .83	Web Server
MY.NET .53 .29	FTP Server / HelpDesk
MY.NET .70 .49	FTP Server / HelpDesk
MY.NET .70 .50	FTP Server / HelpDesk
MY.NET .24 .27	FTP Server
MY.NET .24 .47	FTP Server
MY.NET .24 .15	LPD Print Server
MY.NET .30 .3	Possibly Netware File Services / Web
MY.NET .30 .4	Possibly Netware NetStorage Server

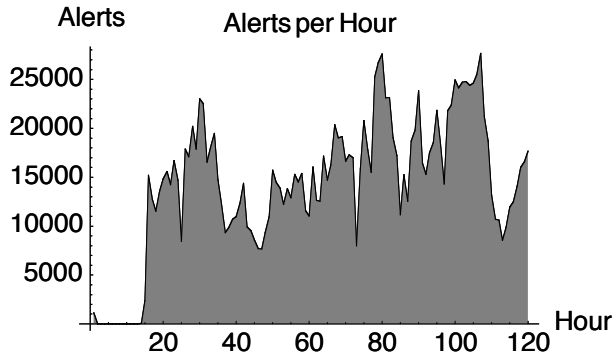
These roles were determined by analyzing the port numbers described in the alerts, and also drawing inferences from the names of various rules.I

Time-Based Analysis of Events

All our logs seem to be missing data from between the 1am-12pm on the 7th April. To make things more difficult with regard to analysing the log file data,the scans data had not been properly obfuscated.I managed the correlate the IP addresses in the scans data with the information in the other files to determine the MY.NET network address. I substituted this information back into the scans data in order to normalize it.

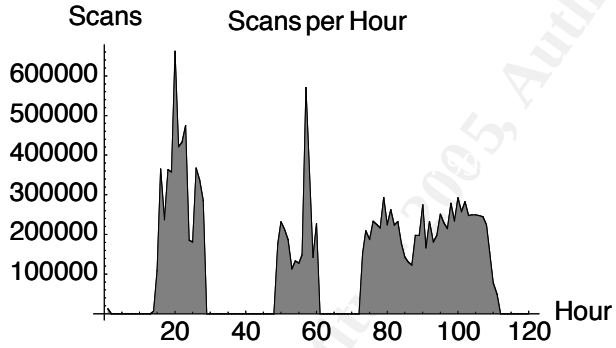
The graphs below indicate the frequency of the various activity recorded by the sensors over the course of the five days. The data was broken into hourly chunks, and the number of events occuring during each hour was determined.

■ Time-Based Analysis of Alerts



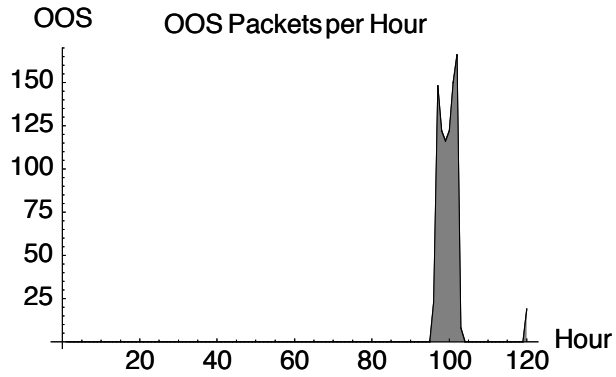
■ Time-Based Analysis of Scans

Our scan log also has three suspicious looking gaps in it – on the 8th between 4am and midnight,on the 9th between midday and midnight,and on the 11th between 3pm and midnight.



■ Time-Based Analysis of Out-of-Specification Packets

About 855 out-of-spec packets occur between 11pm on the 10th through to 7am on the 11th April. There is also 19 out-of-spec packets that occur at the end of our log file data after 11pm on the 11th April.

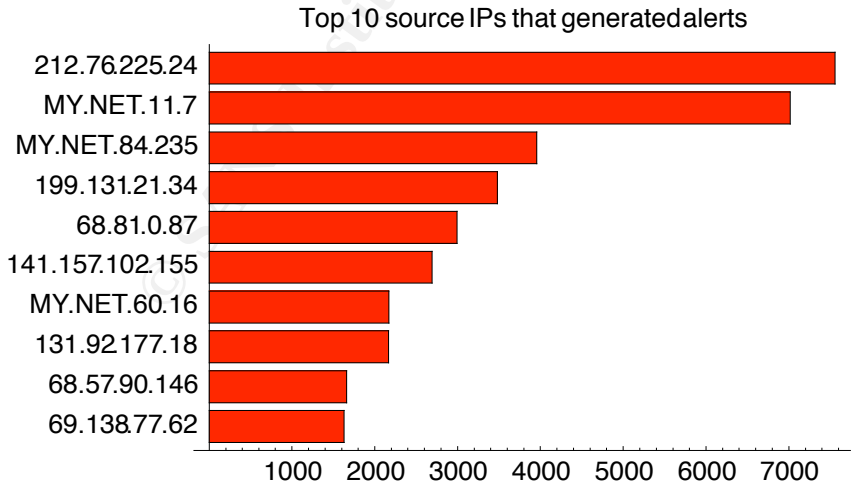


Analysis of Alerts

I have seperated out the alert file data into two types, the alerts which are generated by the snort pre-processor, and "directed alerts" which gives the raw alert generated by a matching snort rule along with source and destination IP addresses. Because the alert data generated by the snort pre-processor is better analyzed by looking at the scan data itself, I have excluded this data from the analysis below.

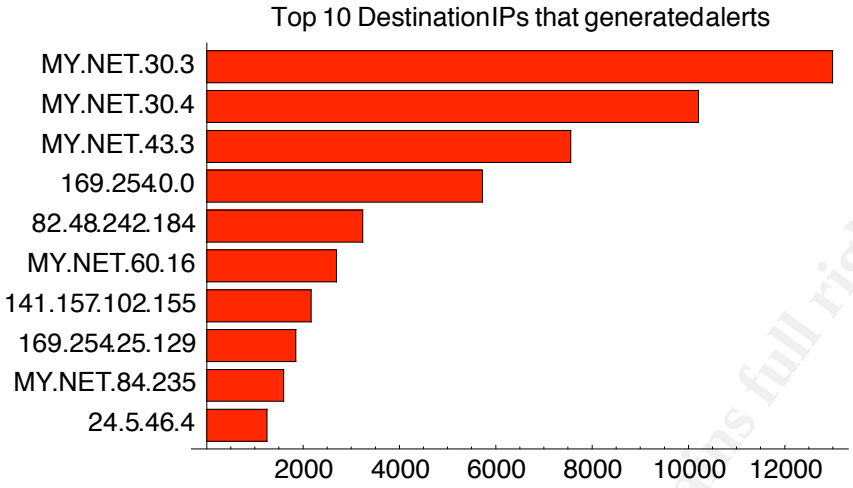
■ Top Talkers - Alerts

As is indicated in the graph below, the IP address generating the most alerts over the five days was 212.76.225.24 which generated 7559 alerts. This was followed closely by the internal IP address MY.NET.11.7 which generated 7016 alerts.



Note that these are machines are the purported origin of packets that are generating these Snort alerts, and we will be investigating the alert generated in more detail to determine whether these are indeed malicious attempts to compromise other machines, or are false positives. But first we will turn our attention to the

destination machines in these alerts. The next graph shows the top ten destinations of the packets that generated the Snort alerts.



If we examine the information presented in these graphs we can see many hosts of interest. We will be examining the hosts MY.NET.84.235, MY.NET.60.16 in more detail in the mstream and red worm alert sections.

The host MY.NET.11.7 is associated with a lot of the 169.254.0.0 and 169.254.25.129 traffic, and is often generated when a Windows-based machine fails to get a DHCP address.

The hosts MY.NET.30.3 and MY.NET.30.4 are associated with the custom alerts "MY.NET.30.x activity", which does not give us much information regarding the nature of the alerts. In examining the destination ports, we can hypothesize that these servers are possibly Netware file servers.

The external host 212.76.225.24 seems to have performed a lot of scanning activity against the host MY.-NET.43.3, particularly against port 0. The "whois" information for the host 212.76.225.24 is given in the appendix, but it appears to be part of the netblock allocated to Coditel Internet Services, which, according to their website - http://www.coditel.be/home/english/default.htm - are the largest cable operator in the Brussels region of Belgium.

■ Analysis of all Alert Types

In total, 51 different types of alert rules were triggered. In the table below I have summarized all the different types of alerts, and indicated next to each what I feel is the severity of the alert. The severity number is from zero to five. Zero indicates that the alert should be largely ignored. I have assigned this number to traffic that I feel indicates either misconfigured devices attaching to the network, or forged packets generated by Denial of Service programs. In these cases, it is extremely difficult to identify the offending host.

The highest severity I have assigned is five, and it indicates that I feel the host is almost certainly compromised. A severity of four indicates that I feel there is a strong possibility

that the host is compromised, but there is also a possibility of a false positive. A severity of three indicates that the alert should be investigated, but there is also a reasonable change of a false positive. Severities of two or one indicate scanning activity, or bad traffic. These should be monitored, and correlated with higher severity alerts.

I selected the mstream trojan, the red worm alerts and the XDCC alerts to analyze, because I feel that with these alerts there is a very high probability of system compromise. The internal hosts MY.NET.84.235 and MY.NET.60.16 are in our top list of alert-generating hosts.

Alert	Count	Category	Severity
DDOS mstream handler to client	3263	Trojan	5
DDOS mstream client to handler	6	Trojan	5
High port 65535 tcp – possible Red Worm – traffic	10667	Trojan	4
Possible trojan server activity	1082	Trojan	4
High port 65535 udp – possible Red Worm – traffic	245	Trojan	4
[UMBC NIDS] Internal MiMail alert	158	MiMail	4
DDOS shaft client to handler	142	Trojan	4
[UMBC NIDS IRC Alert] Possible sdbot floodnet detected attempting to IRC	108	IRC	4
IRC evil – running XDCC	72	IRC	4
[UMBC NIDS] External MiMail alert	47	MiMail	4
[UMBC NIDS IRC Alert] Possible drone command detected.	25	Trojan	4
[UMBC NIDS IRC Alert] Possible Incoming XDCC Send Request Detected.	17	IRC	4
NIMDA – Attempt to execute cmd from campus host	15	Worm	4
TFTP – Internal UDP connection to external tftp server	14	TFTP	4
EXPLOIT NTPDX buffer overflow	10	NTP	4
TFTP – External TCP connection to internal tftp server	4	TFTP	4
[UMBC NIDS IRC Alert] User joining XDCC channel detected. Possible XDCC bot	2	IRC	4
[UMBC NIDS IRC Alert] XDCC client detected attempting to IRC	1	IRC	4
EXPLOIT x86 NOOP	28842	Exploit	3
MY.NET .30 .3 activity	13011	Warning	3
MY.NET .30 .4 activity	10212	Monitoring	3
Tiny Fragments – Possible Hostile Activity	8011	Bad Traffic	3
External RPC call	930	RPC	3
[UMBC NIDS IRC Alert] IRC user /kill detected, possible trojan.	147	IRC	3
TCP SMTP Source Port traffic	83	Bad Traffic	3
EXPLOIT x86 setuid 0	66	Exploit	3
connect to 515 from outside	46	LPD	3
EXPLOIT x86 setgid 0	33	Exploit	3
EXPLOIT x86 stealth noop	28	Exploit	3
RFB – Possible WinVNC – 010708 – 1	24	Win Networking	3
EXPLOIT x86 NOPS	8	Exploit	3
"[UMBC NIDS IRC Alert] K\:line'd user detected, possible trojan."	2	IRC	3
PHF attempt	2	Web	3
External FTP to HelpDesk MY.NET .70 .49	1	FTP	3
External FTP to HelpDesk MY.NET .53 .29	1	FTP	3
External FTP to HelpDesk MY.NET .70 .50	1	FTP	3
Fragmentation Overflow Attack	1	Bad Traffic	3
Null scan!	1127	Mapping	2
NMAP TCP ping!	1098	Mapping	2
SUNRPC highport access!	638	RPC	2
FTP passwd attempt	100	FTP	2
SMB C access	55	Win Networking	2
FTP DoS ftpd globbing	22	FTP	2
Attempted Sun RPC high port access	14	RPC	2
SYN – FIN scan!	13	Mapping	2
Probable NMAP fingerprint attempt	6	Mapping	2
NETBIOS NT NULL session	3	Win Networking	2
SMB Name Wildcard	12178	Win Networking	1
Incomplete Packet Fragments Discarded	512	Bad Traffic	1
TCP SRC and DST outside network	309	Scatter	0
ICMP SRC and DST outside network	210	Scatter	0

Analysis of Specific Alerts

■ DDOS mstream handler to client and client to handler alerts

■ Description of detect

Mstream is a distributed denial of service attack tool. The source code for mstream was released to the security mailing lists BugTraq and vuln-dev on April 29th 2000. There are mstream clients, mstream handlers, and mstream agents. The mstream client is responsible for communicating with the mstream handler and directing them to use the agents to launch denial of service attacks against specific machines. Once the client has connected and the password has been validated, a ">" character is sent as a prompt. The Snort rules match the connection establishment, and then the presentation of this prompt.

There is CVE reference CVE-2000-0138 has information regarding the mstream DDOS tool at the following URL: http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=2000-0138

■ Reason this detect was selected

This detect was selected for further analysis because the mstream tool is strong evidence of a compromised host. This DDOS tool not only affects the host that has been compromised, but also offers a risk to other machines on the network.

■ Detect was generated by

This detect was generated by Snort running in fast alerting mode.

There are a number of snort rules that will trigger these alerts (Snort rule Identifiers 247-249):

Snort rules

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 12754 (msg:"DDOS mstream client to
handler"; flow:to_server,established; content:">"; reference:cve,2000-0138;
classtype:attempted-dos; sid:247; rev:4;)

alert tcp $HOME_NET 12754 -> $EXTERNAL_NET any (msg:"DDOS mstream handler to
client"; flow:to_client,established; content:">"; reference:cve,2000-0138;
classtype:attempted-dos; sid:248; rev:4;)

alert tcp $EXTERNAL_NET any -> $HOME_NET 15104 (msg:"DDOS mstream client to
handler"; flags:S,12; flow:stateless; reference:arachnids,111;
reference:cve,2000-0138; classtype:attempted-dos; sid:249; rev:7;)
```

Sample Log Event

```
alert.040408:04/08-22:57:37.525390  [**] DDOS mstream client to handler [**]
213.231.96.32:4662 -> MY.NET.84.235:12754
alert.040410:04/10-16:56:02.207945  [**] DDOS mstream client to handler [**]
212.195.102.30:4662 -> MY.NET.84.235:12754
```

```
alert.040409:04/09-05:29:41.180250  [**] DDOS mstream handler to client [**]
MY.NET.60.17:15104 -> 65.54.252.99:25
alert.040409:04/09-05:29:41.284717  [**] DDOS mstream handler to client [**]
MY.NET.60.17:15104 -> 65.54.252.99:25
```

- **Probability the source address was spoofed**

It is extremely unlikely the source address given in the alert was spoofed. The Snort alert triggers after the connection has been established - after TCP's 3-way handshake has been completed. Although it is possible to connect to a host using a spoofed source IP address by using the Mitnick attack, loose source routing, or a machine in the path between the spoofed source IP and the destination IP, these attacks are quite difficult to accomplish and it is most likely that the source address has not been spoofed.

- **Attack Mechanism**

This alert indicates that the machine in question has most likely already been compromised and infected with the mstream distributed denial of service attack tool.

The table below displays hosts triggering the "client to handler" alert. The top connection could be a false positive, although we have some other alerts in our logs for host MY.NET.60.38, so I'm suspicious of it. I'm positive that the host MY.NET.84.235 is compromised however. Also note how the client that is connecting to the handler, connects using the same source port - 4662, even though it is connecting from different hosts. This indicates the same tool is probably being used when connecting to the host MY.NET.84.235. The destination port 15104 is associated with mstream handlers that have been observed in the wild, whereas the port number 12754 is the default port in published sources.

Count	Source IP	Source Port	Dest. IP	Dest Port
1	213.180 .193 .68	45101	MY.NET .60 .38	15104
1	212.195 .102 .30	4662	MY.NET .84 .235	12754
1	213.231 .96 .32	4662	MY.NET .84 .235	12754
3	62.42 .66 .52	4662	MY.NET .84 .235	12754

The following table shows the connection data related to the "handler to client" alert. Note the large number of alerts triggering when communicating to the IP address 82.48.242.184. Also, the source address in almost all cases is the host MY.NET.84.235. The other host deemed to be a mstream handler is the host MY.-NET.60.17 communicating to the external host 65.54.252.99 on port 25 (SMTP). Notice that the source port for this traffic is the same as the source port for much of the traffic coming from MY.NET.84.235.

Count	Source IP	Source Port	Dest. IP	Dest Port
5	MY.NET .60 .17	15104	65.54 .252 .99	25
3	MY.NET .84 .235	12754	62.42 .66 .52	4662
3240	MY.NET .84 .235	12754	82.48 .242 .184	4662
1	MY.NET .84 .235	15104	217.236 .97 .47	4662
1	MY.NET .84 .235	15104	80.15 .47 .94	4662
2	MY.NET .84 .235	15104	81.102 .85 .92	4662
1	MY.NET .84 .235	15104	81.69 .163 .174	4662

- **Attribution**

The IP address 62.42.66.52 seems to be part of a cable modem block registered to a company in Spain. The whois information for this host is in the Appendix.
The IP address 82.48.242.184 seems to be part of an ADSL modem pool registered to a company in Italy.

- **Correlations**

There is CVE reference CVE-2000-0138 has information regarding the mstream DDOS tool at: http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=2000-0138

David Goch has an analysis of a mstream handler to client Snort alert triggered from his work IDS. His analysis can be found on the GIAC site: http://www.giac.org/practical/David_Goch_GCIA.doc.

- **Evidence of active targetting**

There is strong evidence here of active targetting. The tool has already been installed on the host machines and the alerts being generated are from the handler directly connecting with the client. The external hosts do not appear to be doing any scanning activity

- **Severity**

Given the following formula:

Severity = (criticality + lethality) - (system countermeasures + network countermeasures)

I would make the following classifications:

Criticality = 4 (The hosts compromised are not on the list of critical hosts, but is on the internal network)
Lethality = 5 (This attack has succeeded and the attack has access to the system)
System Countermeasures = 1
Network Countermeasures = 2 (This attack has been observed at the IDS)

Severity = (4 + 5) - (1 + 2)

Severity = 6

- **References**

"The mstream distributed denial of service attack tool" by David Dittrich, George Weaver, Sven Dietrich and Neil Long
http://staff.washington.edu/dittrich/misc/mstream.analysis.txt

Mike Murphy wrote up an excellent analysis of the mstream architecture at http://www.giac.org/practical/-Michael_Murphy_GCIH.doc

■ XDCC Server Events

■ Description of detect

The following four alerts seem to indicate XDCC server activity:

Alerts
[UMBC NIDS IRC Alert] User joining XDCC channel detected. Possible XDCC bot IRC evil – running XDCC
[UMBC NIDS IRC Alert] Possible Incoming XDCC Send Request Detected.
[UMBC NIDS IRC Alert] XDCC client detected attempting to IRC

An XDCC file server is a common way of making files available for download through IRC. Specifically an IRC "bot" or automated user is established that allows other IRC users to download files by using the DCC protocol. Files made available in this way are referred to as "packs" or "packets". The file can be downloaded by messaging the XDCC automated user with the packet number of the file requested. XDCC servers are installed on compromised machines to make pirated files available for download. They can also be used to provide backdoor access to the compromised system.

There are no CVE entries that specifically address XDCC as it is not so much a vulnerability, as evidence of a compromise.

■ Reason this detect was selected

This detect was selected for further analysis because communication with an XDCC server can often be a sign of a compromised host. Compromising machines and installing an XDCC bot on them seems to have become popular in early 2002, judging from traffic on the unisog mailing list. It seems to have been especially popular to attack University machines with plenty of bandwidth.

■ Detect was generated by

This detect was generated by Snort running in fast alerting mode.

The Snort rules that are triggering these alerts appear to have been created by the client. It is assumed that they match on specific text returned from the XDCC server to the IRC client.

Chris Cramer posted the following Snort rule on the unisog mailing list that generates the XDCC evil alert:
http://www.dshield.org/pipermail/unisog/2002-May/009048.php

```
alert tcp any any -> any 6667 (msg:"IRC evil - running XDCC"; content:"To
request a file type"; nocase;)

alert udp any any -> any 6667 (msg:"IRC evil - running XDCC"; content:"To
request a file type"; nocase;)
```

Gerry Sneeringer notes that he is using essentially the same signature but with an "any any" destination address, given that he has seen XDCC bots connecting to alternative ports on IRC servers.
http://www.dshield.org/pipermail/unisog/2002-May/012034.php

At http://coders.meta.net.nz/~perry/irc.rules I found the following rules:

```
alert tcp $EXTERNAL_NET 6660:7000 -> $HOME_NET any \
(content: " |3a 01|XDCC "; \
msg: "Possible Incoming XDCC Send Request Detected.";\
classtype: misc-activity; \
)

alert tcp $EXTERNAL_NET 6660:7000 -> $HOME_NET any\
(content: " 324 "; offset:5;\
content: "xdcc";\
msg: "User joining XDCC channel detected. Possible XDCC bot";\
classtype:misc-activity;)

alert tcp $HOME_NET any -> $EXTERNAL_NET 6660:7000 \
(content: "USER ";\
content: "dcc"; nocase;\
msg: "XDCC client detected attempting to IRC";\
classtype:misc-activity;)
```

Sample Log Event

```
alert.040408:04/08-11:06:36.078559   [**] IRC evil - running XDCC [**] MY.-
NET.43.2:1916 -> 64.246.60.72:6667
alert.040408:04/08-11:13:30.523419   [**] IRC evil - running XDCC [**] MY.-
NET.43.2:1916 -> 64.246.60.72:6667


alert.040408:04/08-02:32:40.669538   [**] [UMBC NIDS IRC Alert] User joining
XDCC channel detected. Possible XDCC bot [**] 82.96.101.10:7000 -> MY.-
NET.43.2:3088
alert.040408:04/08-02:38:52.200723   [**] [UMBC NIDS IRC Alert] User joining
XDCC channel detected. Possible XDCC bot [**] 82.96.101.10:6665 -> MY.-
NET.43.2:2017
```

- **Probability the source address was spoofed**

Although we cannot say definitely how the Snort rule was crafted, it is most likely that the alert is based on the rules described above and alerts on the contents of IRC traffic detected after a TCP connection has been established between the IRC client and server. This means that the source IP address was most likely not spoofed.

- **Attack Mechanism**

This alert is generated after the host running XDCC has been compromised. In the correlations section, I have mentioned a paper called "XDCC - An .EDU Admin's Nightmare". This paper describes the process of scanning for MS Windows-based machines that have weak passwords, and then installing the iroffer program to act as an XDCC bot. The initial attack vector for Windows machines seems to primarily be Windows networking - NetBIOS/SMB, attacking weak or non-existant passwords and open shares.

The following table shows the IRC Evil connections. We can see numerous alerts for the internal hosts MY.NET.92.79 and MY.NET.43.2.

Count	Source IP	Source Port	Dest. IP	Dest Port
60	MY.NET .43 .2	1916	64.246 .60 .72	6667
1	MY.NET .43 .7	2472	64.62 .196 .26	6667
1	MY.NET .82 .79	1250	207.36 .180 .241	6669
10	MY.NET .82 .79	1275	207.36 .180 .241	6663

The following table shows the alerts related to the detection of a user joining an XDCC channel:

Count	Source IP	Source Port	Dest. IP	Dest Port
1	82.96 .101 .10	6665	MY.NET .43 .2	2017
1	82.96 .101 .10	7000	MY.NET .43 .2	3088

The following table shows the alerts related to incoming XDCC connections:

Count	Source IP	Source Port	Dest. IP	Dest Port
2	207.36 .180 .241	6663	MY.NET .82 .79	1275
1	207.36 .180 .241	6669	MY.NET .82 .79	1250
1	207.44 .214 .88	6667	MY.NET .43 .2	3191
13	64.246 .60 .72	6667	MY.NET .43 .2	1916

Count	Source IP	Source Port	Dest. IP	Dest Port
1	MY.NET .80 .5	1085	69.50 .174 .222	6667

Because of the consistently low source port addresses in these communications, I would say that most of the machines involved are running Windows-based software.

- **Attribution**

The host 64.246.60.72 is associated with alerts on internal hosts MY.NET.43.2 and host MY.NET.80.5. The address block that the IP 64.246.60.72 is a member of is associated with "Everyone's Internet Inc." based in Houston, Texas.This company boasts on it's web site http://my.ev1.net/ that it is the largest independent ISP in the United States.
There is a comment in the registration information that "ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE", I assume that means that addresses in this range are primarily statically-assigned to customers of this company.
The whois for this IP address is included in the appendix.

```
$ grep '64.246.60.72' all_alerts.txt | cut -f 4 | sort | uniq -c
  60 IRC evil - running XDCC
  13 [UMBC NIDS IRC Alert] Possible Incoming XDCC Send Request Detected.
```

- **Correlations**

Chris Cramer posted the snort alert that I believe generates the "IRC evil" messages on the unisog mailing list:
http://www.dshield.org/pipermail/unisog/2002-May/009048.php

Donald Merchant, Philip LjungBerg, Navid Khalili and Rick Larabee describe compromised machines running XDCC in their practicals:
http://www.giac.org/practical/GCIA/Donald_Merchant_GCIA.doc
http://www.giac.org/practical/GCIA/Philip_Ljungberg_GCIA.doc
http://www.giac.org/practical/GCIA/Rick_Larabee_GCIA.doc
http://www.giac.org/practical/GCIA/Navid_Khalili_GCIA.doc

- **Evidence of active targetting**

The usual process for installing an XDCC server on a compromised machine, is to first perform a scan in order to find machines with weak or non-existant security. These machines are then compromised using standard Windows networking techniques. These machines would definitely have been actively targetted after having been found from the scanning activity.

- **Severity**

Given the following formula:

Severity = (criticality + lethality) - (system countermeasures + network countermeasures)

I would make the following classifications:

Criticality = 4 (The hosts compromised are not on the list of critical hosts but are on the internal network)
Lethality = 5 (This attack has succeeded and the attacker has access to the system)
System Countermeasures = 1 (System security is extremely weak)
Network Countermeasures = 2 (Network security is weak, although the XDCC server has been detected)

Severity = (4 + 5) - (1 + 2)

Severity = 6

- **References**

On May 9th, 2002 ion.storm (at) verizon.net posted an email on the unisog mailing list describing a number of compromised machines running an XDCC bot and connecting to the #x-dcc channel on IRC.
http://www.dshield.org/pipermail/unisog/2002-May/009029.php

An analysis of the files found on a compromised machine running an XDCC bot is later given by ion.storm (at) verizon.net. It turns out that the machines are Windows machines and the firedaemon process is being run on them.
http://www.dshield.org/pipermail/unisog/2002-May/009035.php

The paper "XDCC - An .EDU Admin's Nightmare" written by TonikGin is posted on September 11th, 2002 and available at http://www.cs.rochester.edu/~bukys/host/tonikgin/EduHacking.html . It describes the danger

of allowing XDCC within a University network, and also describes the process of compromising a Windows machine and running iroffer and the firedaemon process on the compromised machine.

An analysis of the XDCC infection is given by Dave Dittrich at http://staff.washington.edu/dittrich/talks/core02/xdcc-analysis.txt.

■ High port 65535 tcp/udp - possible Red Worm - traffic

■ Description of detect

The Red Worm which is generally called the Adore worm is associated with activity on port 65535. It targets Linux hosts for compromise, exploiting vulnerabilities in services such as LPRng, rpc-statd, wu-ftpd and BIND. Once it has compromised a host, the Adore worm will attempt to find other Linux machines that are vulnerable to exploitation on these services. We should therefore expect to see a lot of scan activity from compromised machines. Adore can open a backdoor on port 65535 upon receiving a specially crafted ICMP packet.

Note that the triggering of this rule means that not only could the host be infected with the Adore worm, but also someone is communicating with it.

■ Reason this detect was selected

This detect was selected for further analysis because an Adore worm infection is particularly dangerous for Linux hosts. This type of worm will actively infect other machines on the network.

■ Detect was generated by

This detect was generated by Snort running in fast alerting mode.

These appear to be custom rules created by the University, but they seem to be alerting on traffic flowing to and from port 65535. Obviously, there is a possibility of this being a valid ephemeral port, or communication with another service or trojan, so this should be correlated with scanning or other suspicious activity. However any recurring traffic to port 65535 should be investigated.

Sample Log Event

```
04/07-20:26:00.921228  [**] High port 65535 tcp - possible Red Worm - traffic
[**] 66.109.26.25:25 -> MY.NET.25.69:65535
04/07-20:26:00.951688  [**] High port 65535 tcp - possible Red Worm - traffic
[**] 66.109.26.25:25 -> MY.NET.25.69:65535
04/07-20:26:00.951788  [**] High port 65535 tcp - possible Red Worm - traffic
[**] MY.NET.25.69:65535 -> 66.109.26.25:25


04/07-21:38:35.633489  [**] High port 65535 udp - possible Red Worm - traffic
[**] 63.250.197.40:65535 -> MY.NET.84.234:1536
04/07-21:28:34.653697  [**] High port 65535 udp - possible Red Worm - traffic
[**] 206.190.44.106:65535 -> MY.NET.84.234:65280
04/07-21:28:35.028729  [**] High port 65535 udp - possible Red Worm - traffic
[**] 206.190.44.106:65535 -> MY.NET.84.234:65280
```

- **Probability the source address was spoofed**

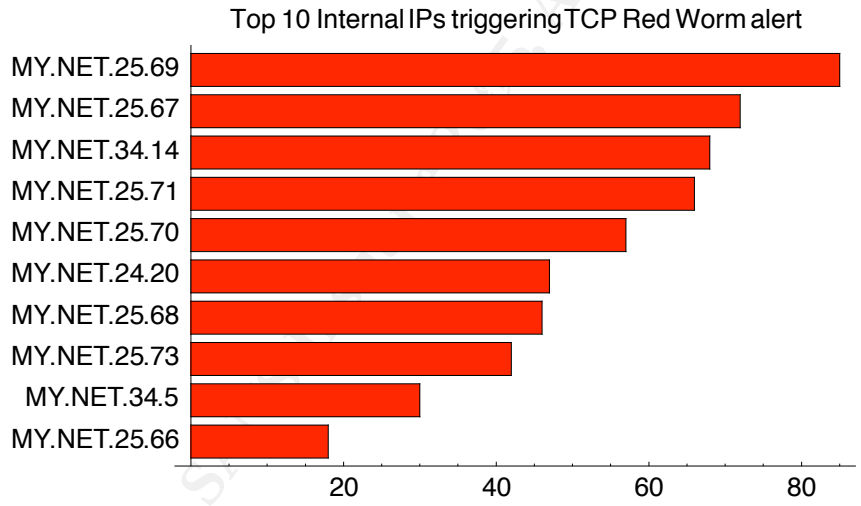
It is extremely unlikely the source address given in the alert was spoofed. The Snort alert triggers after the connection has been established - after TCP's 3-way handshake has been completed. Although it is possible to connect to a host using a spoofed source IP address by using the Mitnick attack, loose source routing, or a machine in the path between the spoofed source IP and the destination IP, these attacks are quite difficult to accomplish and it is most likely that the source address has not been spoofed.

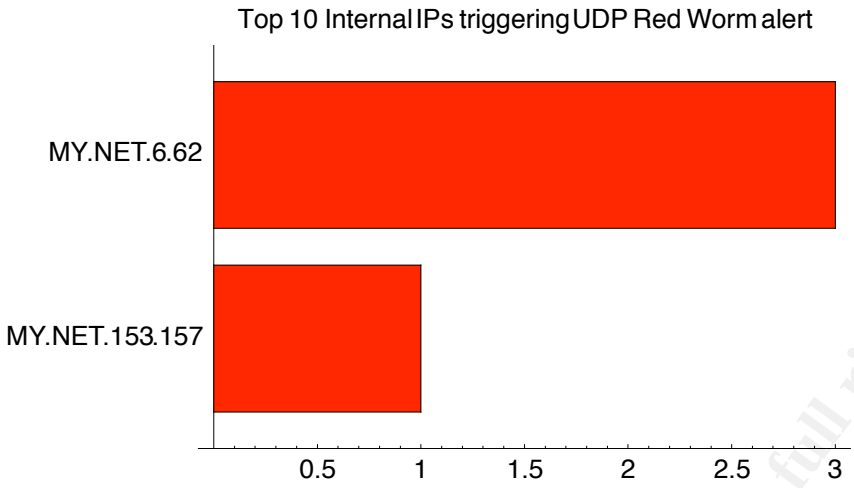
- **Attack Mechanism**

The Adore worm uses documented vulnerabilities in several Linux services to compromise the targetted host. The services attacked include LPRng

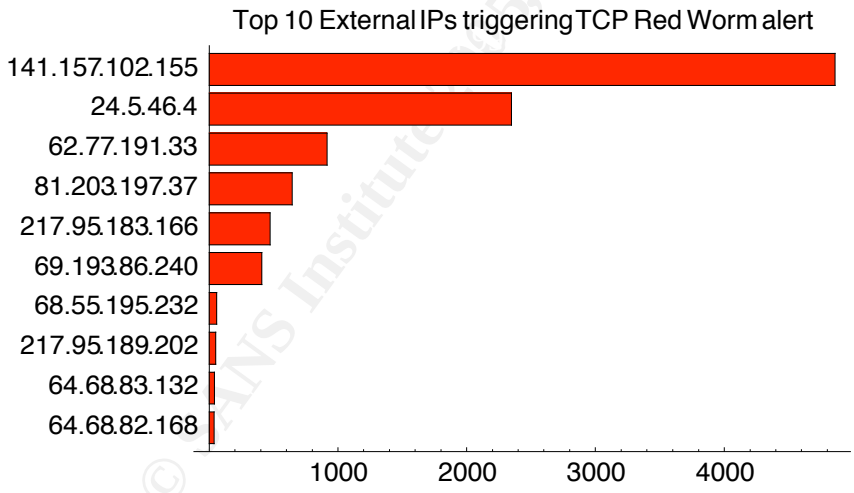
- **Connections**

The graphs below show the number of red worms alerts for the top ten generators of traffic on port 65535. This means that the host can be either be receiving on port 65535 or sending on port 65535. The first graph is the top ten recipients on the TCP port, and second graph is for the top ten recipients on the UDP port. As we can see, there is far less UDP traffic to hosts on port 65535 than TCP traffic being detected, increasing the likelyhood that the UDP traffic could be false positives. We will need to correlate these alerts with other data in order to discriminate between compromised machines and normal activity.





Also of interest is the communication with **external** hosts on this port. This could be evidence of someone internally actively compromising other machines, or the possibility of an internal machine having been compromised and used as a platform to launch other attacks. I will only examine the TCP alerts because the volume of these alerts are far higher than the UDP alerts increasing the chance of infection over false alert.



Obviously with this much traffic to these external machines, it appears as though someone is actively communicating with a trojan on the external machine. The following report shows which internal machines are talking to the top three external machines with a suspected Adore infection.

- **Internal top five hosts communicating with 141.157.102.155**

Internal Host	Count
MY.NET .60 .16	4861

- **Internal top five hosts communicating with 24.5.46.4**

Internal Host	Count
MY.NET .97 .51	1128
MY.NET .97 .92	715
MY.NET .97 .104	197
MY.NET .97 .196	143
MY.NET .97 .182	136

- **Internal top five hosts communicating with 62.77.191.33**

Internal Host	Count
MY.NET .153 .83	915

Additional evidence indicating compromise

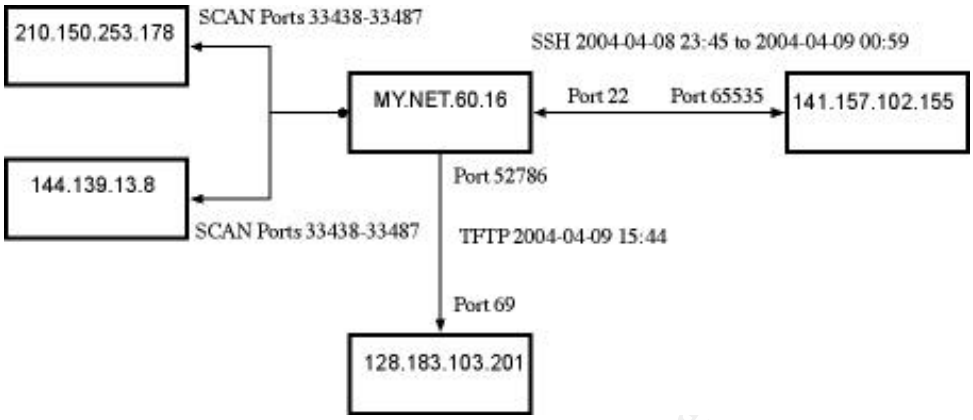
If we examine some hosts uncovered by our investigation into these alerts we find much evidence of compromised machines. These machines have generated a lot of high severity alerts and are actively involved in scan activity.

```
$ grep 'MY.NET.25.69' all_alerts.txt | cut -f 4 | sort | uniq -c | grep -v ports-
can
  85 High port 65535 tcp - possible Red Worm - traffic
   4 Null scan!

$ cut -f 2 all_scans.txt | grep 'MY.NET.25.69' | wc -l
  115998
```

The host below, communicating outbound to an external server on port 65535, has not generated a large number of alerts, other than triggering the red worm alert. At first I though that this could be game activity, or file sharing activity. After doing further analysis however, I found there was bidirectional communication happening between MY.NET.60.16 on port 22 and 141.157.102.155 on port 65535. Port 22 is commonly associated with sshd, but it is also associated with the trojaned version of sshd installed with Adore. The fact that is connecting to an external server using tftp is also additional evidence of a compromised host.

```
$ grep 'MY.NET.60.16' all_alerts.txt | cut -f 4 | sort | uniq -c | grep -v ports-
can
  23 EXPLOIT x86 NOOP
4861 High port 65535 tcp - possible Red Worm - traffic
   1 TFTP - Internal UDP connection to external tftp server
```

```
$ cut -f 2 all_scans.txt | grep 'MY.NET.60.16' | wc -l
       95

$ grep 'MY.NET.60.16' all_alerts.txt | grep 65535 | cut -f 5 | sort | uniq -c
2693 141.157.102.155:65535 -> MY.NET.60.16:22
2168 MY.NET.60.16:22 -> 141.157.102.155:65535
```



■ Attribution

The netblock associated with the IP address 141.157.102.155 is registered to Verizon Internet Services. It looks like this address might be associated with a DSL modem pool. The whois information is displayed below:

```
$ whois  141.157.102.155
Verizon Internet Services VIS-141-149 (NET-141-149-0-0-1)
                                141.149.0.0 - 141.158.255.255
Verizon Internet Services VZ-DSLDIAL-CYVLMD-9 (NET-141-157-57-0-1)
                                141.157.57.0 - 141.157.126.255

# ARIN WHOIS database, last updated 2004-11-04 19:10
# Enter ? for additional hints on searching ARIN's WHOIS database.
```

■ Correlations

The original email from Matt Fearnow of SANS describing the worm is archived at https://ucsb.edu/pipermail/security-linux/2001-April/000121.html.

The following CVE entries describe the vulnerabilities exploited by the worm:
CVE-2000-0666: rpc.statd - http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0666
CVE-2000-0917: LPRng - http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0917
CVE-2001-0010: BIND 8 - http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0010
CVE-2001-0011: BIND 4 - http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0010
CVE-2000-0573: wu-ftpd - http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0573

There is a wealth of additional correlations in the practicals of Christine Chan, Dennis Ruck, Dongmei Huang, Ernest Eustace and Clifford Yago.

http://www.giac.org/practicals/GCIA/Christine_Chan_GCIA.doc
http://www.giac.org/practicals/GCIA/Dennis_Ruck_GCIA.doc
http://www.giac.org/practicals/GCIA/Dongmei_Huang_GCIA.doc
http://www.giac.org/practicals/GCIA/Ernest_Eustace_GCIA.doc

- **Evidence of active targetting**

The Adore worm scans the network looking for vulnerable services open to exploitation, it is only after determining that the port is open that an exploit is attempted. This attack is therefore targetted towards hosts who are potentially vulnerable to exploitation on the targetted services.

- **Severity**

Given the following formula:

Severity = (criticality + lethality) - (system countermeasures + network countermeasures)

I would make the following classifications:

Criticality = 4 (The hosts compromised are not on the list of critical hosts)
Lethality = 5 (This attack has succeeded and the attacker has access to the system. There is evidence that an attacker has connected to a compromised host)
System Countermeasures = 1
Network Countermeasures = 2 (The adore worm infections have been detected by the IDS).

Severity = (4 + 5) - (1 + 2)

Severity = 6

- **References**

SANS has an excellent overview of Adore at http://www.sans.org/y2k/adore.htm.
The program adorefind is available at http://www.ists.dartmouth.edu/library/classroom/adorefind.php

Clifford Yago has written about the Adore worm in his GCIA practical http://www.giac.org/practical/Clifford_-Yago_GCIA.doc

Analysis of Out-of-Spec Packets

A breakdown of the port numbers associated with out-of-spec packets appears below. Notice the traffic that could indicate the presence of the eDonkey2000 file sharing software.
Of interest is also traffic that I believe is associated with the ResNet application. There is more about ResNet here: http://www.umbc.edu/oit/resnet/orig/faq.html

Port	Number of Packets	Associated Service
25	2029	SMTP
110	1701	POP
80	825	HTTP
113	255	ident
4662	249	eDonkey2000
24842	156	ResNet
8080	116	HTTP
3247	19	Unknown
443	18	HTTPS
22	16	sshd

Defensive Recommendation

■ Recovery

The first recommendation I would make is to perform an investigation into all the machines believed to be infected. In the case of the Linux boxes infected with Adore and mstream, take them off-line. The Adore infected boxes with almost certainly need to be rebuilt.

■ Policy

Internal policy should be reviewed. Is the XDCC IRC bot allowed on machines in your internal network? What is the policy on Anti-Virus software and Personal Firewalls? How frequently should machines be patched?

The University could potentially look into providing discounts on Anti-Virus and Personal Firewall software to the students and faculty.

■ Firewall Rules

The firewall rules should be reviewed. The University needs to decide how permissive they can afford to be with regard to what services they allow into their network. At the moment, there just seems to be too may scans and exploit attacks making it through the border perimeter. There should also be an established process for reviewing the firewall rules on a regular basis.

There seems to be legitimate sites that are able to access certain machines through the firewall. This policy should be reviewed. For example, the machine 209.132.177.100 that belongs to Red Hat, seems to be able to access via RPC the machines MY.NET.70.154 and MY.NET.70.37. While this access may be legitimate, the granting of such access should be reviewed on a regular basis. Of some concern is the sustained access to internal host MY.NET.82.106 from the machines 64.12.25.40 and 64.12.25.43 on the 8th April from 13:47 to 15:25. These machines are registered to the AOL net block.

■ IDS Recommendations

The version of Snort should be upgraded to the latest stable. A review of the Snort ruleset is in order.

Analysis Process

■ Tools Used

The analysis process was conducted on an Apple Powerbook running OS/X. I used PERL to parse and extract data from the log files in meaningful ways. I processed the data, produced the graphs and wrote up this report using Wolfram's *Mathematica* version 5. I would thoroughly recommend this setup as it allowed me to export the results in tab deliimited files, which I then would import into *Mathematica* and use *Mathematica* to do detailed analysis on them. If any of the scripts needed to be re-run, I could reproduce the text file for *Mathematica* to re-import.

Another benefit on *Mathematica* is that I was able to document the analysis process as I went along.

■ The Analysis Process

I started off by concatenating all the log files into 3 different tab-delimited files: all_alerts.txt for the alerts, all_scans.txt for the scans, and all_oos.txt for the out of spec packets. I then used a combination of PERL scripts and Unix command line tools to extract and summarize the data. I wrote over 15 different PERL scripts used to perform this analysis. I will put all the PERL scripts on my web site at http://hutley.net/brett/security/gcia_tools.tgz.

I also used various Unix command line tools quite liberally. Examples of this are presented below:

One of the problems I found was with using grep in filtering for IP addresses. When trying to filter for an address - say; MY.NET.1.3, be aware that the 'dot' tells grep to match any character. So if you just type

```
$ grep MY.NET.1.3 all_alerts.txt
```

This will not only match MY.NET.1.3, but also addresses starting with MY.NET.103. etc. I had to be very careful when scripting on the command line to escape any dots that were passed to grep.

■ Getting the ordered list of RPC accesses / count

$ grep -i 'rpc high' all_alerts.txt | cut -f 9 | sort | uniq -c | sort -n

■ Determining the alerts which were triggered on a source port or destination port of 515

$ cat all_alerts.txt | perl -e 'while (<>) { my @fields = split(/\t/); if ($fields[7] == 515 || $fields[9] == 515) { print $_; } }' | cut -f 9,10 | sort | uniq -c

- **Reporting the top ten external hosts generating the red worm tcp alerts.**

```
$ grep 'port 65535 tcp - possible Red Worm - traffic' all_alerts.txt  | perl -e
'while (<>) { my @fields = split(/\t/); if ($fields[7] == 65535) { print
$fields[6]."\n"; } elsif ($fields[9] == 65535) { print $fields[8]."\n"; } }' |
sort | uniq -c | sort -rn | grep -v 'MY.NET' | head -10 >
a_red_worm_top_external_infected_tcp.txt
```

- **Determining a count of the different alerts triggered for a host.**

```
$ grep 'MY.NET.153.35' all_alerts.txt | cut -f 4 | sort | uniq -c | grep -v
portscan
  65 EXPLOIT x86 NOOP
   1 EXPLOIT x86 setgid 0
   3 EXPLOIT x86 setuid 0
   3 High port 65535 tcp - possible Red Worm - traffic
  59 High port 65535 udp - possible Red Worm - traffic
   2 Incomplete Packet Fragments Discarded
  14 NMAP TCP ping!
  51 Null scan!
   1 SYN-FIN scan!
   2 Tiny Fragments - Possible Hostile Activity
```

- **Generating a report of which internal hosts are communicating with the top three external Adore infected machines**

```
$ for d in `cut -f 1 a_red_worm_top_external_infected_tcp.txt | head -3`; do
echo $d; echo "==========="; grep $d all_alerts.txt  | perl -e 'while (<>)
{ my @fields = split(/\t/); if ($fields[7] == 65535) { print $fields[8]."\n"; }
elsif ($fields[9] == 65535) { print $fields[6]."\n"; } }' | sort | uniq -c |
sort -rn |  head -10; echo; echo; done
```

- **Determining the scans performed by a particular IP address**

```
$ cat all_scans.txt | perl -e 'while (<>) { my @flds = split(/\t/); if
($flds[1] eq "130.85.60.16") { print $flds[3]."\t".$flds[4]."\n"; } }'
```

Appendix

Score-based list of potentially compromised internal hosts

The following table list hosts that are potentially compromised, determined by automatically analyzing the following data associated with the host: Count of the different severity levels for alerts based on the severities I have assigned the alerts earlier. A count of the number of scans detected originating from the machine. A count of the number of different alert types associated with the host. This information was used to produce a score, or likelyhood the machine is compromised. I have listed the highest 30 scoring hosts below.

Host	Sev 3	Sev 4	Sev 5	Scans	Num Alert Types	Scan Score	Total Score
MY.NET.84.235	832	1403	3255	295221	10	100	326996300
MY.NET.60.16	0	4861	0	95	2	0	4863000
MY.NET.97.51	0	1127	0	2224	1	1	1128001
MY.NET.153.83	6	915	0	730	3	0	918600
MY.NET.43.3	7492	0	0	4109	8	1	757201
MY.NET.97.92	2	715	0	0	4	0	719200
MY.NET.60.17	65	7	5	16	4	0	517500
MY.NET.97.213	2	408	0	0	3	0	411200
MY.NET.12.6	106	201	0	32	8	0	219600
MY.NET.97.104	2	196	0	212	2	0	198200
MY.NET.110.82	3	155	0	56	3	0	158300
MY.NET.97.196	0	142	0	1084	1	1	143001
MY.NET.97.182	0	135	0	0	1	0	136000
MY.NET.84.236	1055	0	0	0	1	0	106500
MY.NET.60.38	0	1	1	0	3	0	104000
MY.NET.24.34	17	93	0	0	5	0	99700
MY.NET.24.44	61	84	0	0	6	0	96100
MY.NET.17.3	915	0	0	0	1	0	92500
MY.NET.43.2	17	79	0	1905	8	1	88701
MY.NET.24.74	93	74	0	0	5	0	88300
MY.NET.25.69	0	84	0	115998	2	100	86100
MY.NET.70.74	831	0	0	0	1	0	84100
MY.NET.153.35	70	62	0	1522547	10	1000	80000
MY.NET.34.14	0	73	0	233328	5	100	78100
MY.NET.6.7	46	68	0	0	4	0	76600
MY.NET.25.67	0	71	0	65599	2	1	73001
MY.NET.70.225	41	59	0	78559	7	1	70101
MY.NET.25.71	0	65	0	145611	2	100	67100
MY.NET.25.70	1	56	0	148578	3	100	59200

■ Whois for 62.42.66.52

```
# ARIN WHOIS database, last updated 2004-11-03 19:10
# Enter ? for additional hints on searching ARIN's WHOIS database.
% This is the RIPE Whois secondary server.
% The objects are in RPSL format.
%
% Rights restricted by copyright.
% See http://www.ripe.net/db/copyright.html

inetnum:      62.42.0.0 - 62.42.127.255
netname:      ONO-SCOPES-4
descr:        Cableuropa - ONO
descr:        ONO net in whole Spain
country:      ES
admin-c:      OIM1-RIPE
tech-c:       OIM1-RIPE
status:       ASSIGNED PA
remarks:      mail spam reports: abuse@ono.com
remarks:      security incidents: security@ono.com
notify:       ripe-tech@ono.es
mnt-by:       ONO-MNT
changed:      ripe-tech@ono.es 20030318
source:       RIPE

route:        62.42.0.0/16
descr:        Cableuropa - Ono
descr:        Ono network in whole Spain
origin:       AS6739
remarks:      mail spam reports: abuse@ono.com
remarks:      security incidents: security@ono.com
mnt-by:       ONO-MNT
changed:      ripe-tech@ono.es 20020619
source:       RIPE

role:         ONO IP MANAGER
address:      C/ Basauri, 5
address:      Urbanizacion La Florida
address:      E-28023 Aravaca, Madrid
address:      SPAIN
phone:        +34911809300
fax-no:       +34911809245
e-mail:       ripe-tech@ono.es
admin-c:      JMD-RIPE
tech-c:       JMD-RIPE
tech-c:       JABM1-RIPE
tech-c:       MJS6-RIPE
tech-c:       MJC7-RIPE
tech-c:       AGG20-RIPE
tech-c:       FRL9-RIPE
nic-hdl:      OIM1-RIPE
changed:      ripe-tech@ono.es 20030422
```

```
source:       RIPE
```

■ Whois for 82.48.242.184

```
# ARIN WHOIS database, last updated 2004-11-03 19:10
# Enter ? for additional hints on searching ARIN's WHOIS database.
% This is the RIPE Whois secondary server.
% The objects are in RPSL format.
%
% Rights restricted by copyright.
% See http://www.ripe.net/db/copyright.html

inetnum:      82.48.240.0 - 82.48.255.255
netname:      TELECOM-ADSL-3
descr:        Telecom Italia S.p.A.
descr:        E@sy.ip service
descr:        Wholesale service for ISP
country:      IT
admin-c:      BS104-RIPE
tech-c:       BS104-RIPE
status:       ASSIGNED PA
remarks:      Please send abuse notification to abuse@telecomitalia.it
notify:       net_ti@telecomitalia.it
mnt-by:       TIWS-MNT
changed:      net_ti@telecomitalia.it 20040101
source:       RIPE

route:        82.48.0.0/16
descr:        INTERBUSINESS
origin:       AS3269
notify:       network@cgi.interbusiness.it
mnt-by:       TIWS-MNT
mnt-routes:   INTERB-MNT
changed:      net_ti@telecomitalia.it 20031016
source:       RIPE

person:       BBBEASYIP STAFF
address:      Via Val Cannuta, 250
address:      I-00100 Roma
address:      Italy
phone:        +39 06 36881
e-mail:       ripe-staff@telecomitalia.it
nic-hdl:      BS104-RIPE
notify:       ripe-staff@telecomitalia.it
changed:      net_ti@telecomitalia.it 20001019
source:       RIPE
```

■ Whois for 64.246.60.72

```
$ whois 64.246.60.72

OrgName:    Everyones Internet, Inc.
OrgID:      EVRY
Address:    2600 Southwest Freeway
Address:    Suite 500
City:       Houston
StateProv:  TX
PostalCode: 77098
Country:    US

NetRange:   64.246.0.0 - 64.246.63.255
CIDR:       64.246.0.0/18
NetName:    EVRY-BLK-9
NetHandle:  NET-64-246-0-0-1
Parent:     NET-64-0-0-0-0
NetType:    Direct Allocation
NameServer: NS1.EV1.NET
NameServer: NS2.EV1.NET
Comment:    ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate:    2001-10-05
Updated:    2003-03-31

TechHandle: RW172-ARIN
TechName:   Williams, Randy
TechPhone:  +1-713-400-5400
TechEmail:  admin@ev1.net

OrgAbuseHandle: ABUSE477-ARIN
OrgAbuseName:   ABUSE
OrgAbusePhone:  +1-713-400-5400
OrgAbuseEmail:  abuse@ev1.net

OrgNOCHandle: NOC1445-ARIN
OrgNOCName:   NOC
OrgNOCPhone:  +1-713-400-5400
OrgNOCEmail:  noc@ev1.net

OrgTechHandle: RW172-ARIN
OrgTechName:   Williams, Randy
OrgTechPhone:  +1-713-400-5400
OrgTechEmail:  admin@ev1.net

OrgTechHandle: VST3-ARIN
OrgTechName:   Stinson, Valarie
OrgTechPhone:  +1-713-400-5400
OrgTechEmail:  admin2@ev1.net

# ARIN WHOIS database, last updated 2004-11-04 19:10
```

```
# Enter ? for additional hints on searching ARIN's WHOIS database.
```

■ Whois for 212.76.225.24

```
# ARIN WHOIS database, last updated 2004-11-27 19:10
# Enter ? for additional hints on searching ARIN's WHOIS database.
% This is the RIPE Whois tertiary server.
% The objects are in RPSL format.
%
% Rights restricted by copyright.
% See http://www.ripe.net/db/copyright.html

inetnum:      212.76.225.0 - 212.76.225.255
netname:      CODITEL
descr:        Coditel - Internet Services
country:      BE
admin-c:      XD6
tech-c:       YB490-RIPE
status:       ASSIGNED PA
notify:       tech.registry@coditel.be
mnt-by:       CODITEL-MNT
mnt-lower:    CODITEL-MNT
changed:      xavier.darche@coditel.be 20010109
changed:      xavier.darche@coditel.be 20030513
changed:      xavier.darche@coditel.be 20030514
source:       RIPE
```

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
SANS Las Vegas 2018	Las Vegas, NV	Jan 28, 2018 - Feb 02, 2018	Live Event
Las Vegas 2018 - SEC503: Intrusion Detection In-Depth	Las Vegas, NV	Jan 28, 2018 - Feb 02, 2018	vLive
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