Global Information Assurance Certification Paper

Security Audit for A University

GIAC Intrusion Analyst Certification

Version 4.1

Prathaben Kanagasingham

Date Submitted: 02/12/05

Table of Contents

Part .1 Executive Summary

The primary goal of this audit is to examine university security logs for security violations that had occurred on the University network and to provide information on easier management of IDS based on the activity and some policy changes on the firewall.

Three days worth of logs have been examined to detect possible malicious activities and they are from 03/02/06, 03/02/07, 03/02/08, 03/02/09 and 03/02/10. Three different types of logs were used for analysis and they were taken from snort Intrusion Detection System (IDS) device. The log types are:

- Alert
- Scan
- OOS (Out of Spec)

Of the activities detected, that require special attention are those that involve internal hosts scanning either university hosts or external hosts, since these hosts are most probably compromised or this might be a special effort made by the university security staff to take proactive measures to keep the university network secure. Such activities involve but does not limit to scanning of hosts on different ports in an attempt to reveal the services being offered by them. Secondly, several campus hosts are involved in file sharing using peer-to-peer (P2P) applications. Use of peer-to-peer applications for file sharing can result in a number of security / policy violations and they are consumption of bandwidth, copyright violations, circumventing security policies and spreading malicious codes.

Following the analysis of the alerts, I have also produced a list of top twenty alerts, top five destination hosts and top five source hosts for easy perusal. Of these, the most critical alerts that pose serious risks to the university network

were examined thoroughly and reported in detail.

One of the biggest challenges in managing IDS is dealing with positives. A signature triggering alerts on legitimate activity is known as false positives. While, an IDS device can never be tuned to completely eliminate false positives, it is highly possible that they can be reduced to a point, where an Analyst spends more time on investigating the actual threats to the network. However, such real time monitoring done in house is very costly and will require more security staff. I have made a comparison in the cost effectiveness between in house IDS monitoring and managed security service providers (MSSP) in order to enhance security on the university network.

Part .2 Detailed Analysis

2.1 Alert, Scan and OOS logs

Alert, scan and OOS (Out of spec) files used for analysis are shown below.

Alert	Scan	OOS (Out Of Spec)
alert.030206.gz	**Scan.030206.gz**	**oos_report_030206.gz**
alert.030207.gz	**Scan.030207.gz**	**oos_report_030207.gz**
alert.030208.gz	**Scan.030208.gz**	**oos_report_030208.gz**
alert.030209.gz	**Scan.030209.gz**	**oos_report_030209.gz**
alert.030210.gz	**Scan.030210.gz**	**oos_report_030210.gz**

Table 2.1 Names and Dates of logs used in the analysis

First line and last line of all three log files have been copied below for easier identification of the logs.

Snip from alert logs

```
02/06-00:00:01.595897  [**] SMB Name Wildcard [**] 65.66.227.29:137 ->
172.16.201.206:137
02/10-23:46:03.427280  [**] SMB Name Wildcard [**] 202.88.142.8:1127 ->
MY.NET.43.248:137
```

Snip from scan logs

```
Feb  6 00:04:59 65.214.36.150:48983 -> 172.16.24.34:80 SYN 12****S*
RESERVEDBITS
Feb 10 23:33:49 65.80.163.85:0 -> 172.16.212.42:0 NULL ********
```

Snip from OOS logs

```
02/05-00:06:24.539729 68.50.34.213:50640 -> MY.NET.24.34:80
TCP TTL:48 TOS:0x0 ID:37263 IpLen:20 DgmLen:60 DF
12****S* Seq: 0x8EED3720  Ack: 0x0  Win: 0x16D0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 304815 0 NOP WS: 0

02/09-23:49:21.617773 148.64.22.71:1483 -> MY.NET.87.232:6346
TCP TTL:114 TOS:0x0 ID:19045 IpLen:20 DgmLen:157 DF
****P*** Seq: 0x8AB1AE0A  Ack: 0x0  Win: 0x2000  TcpLen: 20
47 4E 55 54 45 4C 4C 41 20 43 4F 4E 4E 45 43 54  GNUTELLA CONNECT
2F 30 2E 36 0D 0A 55 73 65 72 2D 41 67 65 6E 74  /0.6..User-Agent
3A 20 4D 79 4E 61 70 73 74 65 72 20 33 2E 34 2E  : MyNapster 3.4.
33 2E 30 0D 0A 58 2D 55 6C 74 72 61 70 65 65 72  3.0..X-Ultrapeer
3A 20 46 61 6C 73 65 0D 0A 58 2D 51 75 65 72 79  : False..X-Query
2D 52 6F 75 74 69 6E 67 3A
```

2.2 Top talkers

Of the several alert reports generated using SnortSnarf, I have tabulated the top twenty alerts below based on the alert count.

Signature	Number of Alerts	Number of Sources	Number of Destinations
SMB Name Wildcard	75827	16111	31805
Watchlist 000220 IL-ISDNNET-990517	16652	151	266
High port 65535 tcp –possible Red Worm – traffic	11835	119	130
Spp_http_decode: IIS Unicode attach detected	9573	547	769
CS WEBSERVER – external web traffic	6672	2567	1
spp_http_decode: CGI Null Byte attack detected	2471	97	105
TFTP - Internal TCP connection to external tftp server	2035	21	22
High port 65535 udp - possible Red Worm – traffic	1417	181	195
SYN-FIN scan!	1335	2	1046
TFTP - External UDP connection to internal tftp server	1241	5	1
Null scan!	1145	97	82
External RPC call	1054	6	985

Incomplete Packet Fragments Discarded	1025	84	36
SUNRPC highport access!	1005	39	25
Possible trojan server activity	687	39	243
EXPLOIT x86 NOOP	647	74	71
Queso fingerprint	644	197	70
NIMDA - Attempt to execute cmd from campus host	622	2	269
Port 55850 tcp - Possible myserver activity - ref. 010313-1	542	59	60
TCP SMTP Source Port traffic	419	2	419

Table 2.2 Alert name and total count

Five source hosts and destination hosts, which triggered the most number of alerts have been listed below on table 2.3 and table 2.4 respectively with information on the signatures they triggered the alerts.

Source IP	No. of Alerts	Signatures triggered
172.16.239.126	3017	Possible Trojan server activity Port 55850 tcp – Possible myserver activity High port 65535 tcp - possible Red Worm traffic
212.179.19.161	2514	Watchlist 000220 IL-ISDNNET-990517 SMB Name Wildcard EXPLOIT x86 NOOP
212.179.103.225	1996	Watchlist 000220 IL-ISDNNET-990517
212.179.102.211	1961	Watchlist 000220 Il-ISDNNET-990517 SMB NAME Wildcard
172.16.201.146	1945	High port 65535 tcp – possible Red worm traffic Spp_http_decode:IIS Unicode attack detected

Table 2.3 Top five source hosts with alert description

Destination IP	No. of Alerts	Signatures triggered

172.16.100.165	7071	CS WEBSERVER – external we traffic CS WEBSERVER – external ftp traffic SMB Name Wildcard Queso fingerprint NMAP TCP ping Watchlist 000220 IL-ISDNNET-990517
24.50.140.77	3004	High port 65535 tcp – possible Red Worm – traffic
172.16.218.78	2615	Watchlist 000200 IL-ISDNNET-990517 SMB Name Wildcard
172.16.242.158	1999	Watchlist 000200 IL-ISDNNET-990517 SMB Name Wildcard
172.16.234.118	1969	Watchlist 000200 IL-ISDNNET-990517 SMB Name Wildcard Exploit x86 NOOP

Table 2.4 Top five destination hosts with alert description

2.3 Top five targeted service ports

In this section, I will be discussing about the most targeted services. Having analyzed the alert logs, I came up with the following summary of top five targeted ports based on the number of alerts on each port.

Port Number	Alert count
137	75824
80	7490
1214	5337
2708	1959
3166	1739

Table 2.6 Port numbers and alert counts

Port 137 – This is the most targeted port on the five days worth of alert logs and this port is associated with NetBIOS traffic. Most of the alerts are SMB Name Wildcard alerts. Most of this traffic in this activity seems to be portsweep scanning on internal hosts. This traffic should be monitored for any further suspicious activity.

Port 80 – This is the second most targeted port and it is the webserver activity

on host 172.16.100.165. The alerts triggered are CS WEBSERVER –external web traffic. With the information available, I am unable to confirm what the name CS WEBSERVER refers to. This probably refers to external host accessing university website, which seems to be an informational alert or a way of logging users accessing the website.

Port 1214 – This is the third most targeted port and this port is associated with Kazaa file sharing. Allowing this activity is not recommended.

Port 2708 – This is the fourth most targeted port and is associated with Banyan-net. Alerts triggered on this port are Watchlist 000220 IL-ISDNNET-990517. As the name indicates, host 212.179.102.211 might have been on the watchlist.

Port 3166 - Lastly, port 3166 has been fifth most targeted port and this port is associated with quest repository. Alerts triggered are possible red worm on traffic from host 24.43.34.17 to 172.16.201.146. Involvement of port 65535 from host 24.43.34.17 caused the IDS to trigger the alerts and very likely false positives.

Table 2.5 Top five services targeted

2.4 Three most suspicious external IP addresses

For the selection of top three suspicious IPs, random scanning, Kazaa traffic (port 1214), Gnutella traffic (port 6346) and any other P2P file sharing application traffic have been excluded. The most suspicious hosts are picked based on any possible crafting of packet or any active scanning to a particular service on campus hosts and they are the following.

Source IP	Alert count
210.187.4.222	57
130.240.13.82	9
65.57.64.224	50

Table 2.6 Top three suspicious source IP addresses

Attacking IP 210.187.4.222

```
Feb 6 11:18:38 210.187.4.222:4112 -> 172.16.244.118:1313 INVALIDACK
***APRSF
Feb 6 11:30:01 210.187.4.222:4170 -> 172.16.244.118:1313 NOACK 12U**R**
RESERVEDBITS
Feb 6 11:45:18 210.187.4.222:4170 -> 172.16.244.118:1313 NMAPID **U*P*SF
Feb 6 11:46:08 210.187.4.222:4170 -> 172.16.244.118:1313 NOACK **U*PRS*
Feb 6 11:50:23 210.187.4.222:4170 -> 172.16.244.118:1313 SYNFIN *2****SF
RESERVEDBITS
Feb 6 12:04:03 210.187.4.222:4618 -> 172.16.244.118:1313 NOACK **U**R**
```

Only target in this attack has been host 172.16.244.118, hence appears to be an active target by the attacking IP. These packets are definitely crafted and tweaks used in manipulating the TCP flags point to possible “Queso” or “Nmap” scan. Alerts with the phrase “RESERVEDBITS” indicate that reserved bits have been used. Reserved bits are CWR (Congestion Window reduced and ECN (Explicit Congestion Notification) and they are used to handle congestion notification between the sender and receiver. Scanning tools use a mutant combination of the TCP flags including reserved bits to determine the operating system.

I am unable to run any passive OS fingerprinting due to lack of binary data. However based on the TTL value and win size, I have approximated the operating system of the attacker. These alerts contain a TTL value of 109 and win size of 0x42E3, which translates to 17123 in decimal. Windows 2000 by default sets the TTL value to be 128 and has a windows size of anywhere between 17000 and 18000. Since the attacking IP 210.187.4.222 is not reachable, an accurate TTL value cannot be determined and moreover traceroute will have to be done from the University premises for further accuracy. Netblock in the range of 210.187.0.0 is about to 15 hops away. With this information I have approximated the operating system to be windows 2000. This OS detection is based on the values found the link below and a report of which has been included in the Appendix A.
http://www.honeynet.org/papers/finger/traces.txt

Attacking host belongs to Malaysia based on the following information.

inetnum:	210.187.0.0 - 210.187.127.255
netname:	TMNET-MY
descr:	TMnet Telekom Malaysia
country:	MY
admin-c:	AS115-AP
admin-c:	EU3-AP
admin-c:	SM135-AP
tech-c:	AS115-AP
tech-c:	EU3-AP
tech-c:	SM135-AP
mnt-by:	APNIC-HM
mnt-lower:	TM-NET-AP
changed:	hostmaster@apnic.net 20010529
status:	ALLOCATED PORTABLE
source:	APNIC

Attacking IP 130.240.13.82

```
02/06-13:44:18.342548 130.240.13.82:80 -> 172.16.82.38:2588 TCP TTL:46
TOS:0x0 ID:8331 IpLen:20 DgmLen:40 ******** Seq: 0x0  Ack: 0x0  Win: 0x800
TcpLen: 20

02/06-13:44:29.602047  130.240.13.82:80  ->  172.16.82.38:2776  TCP  TTL:46
TOS:0x0 ID:8758 IpLen:20 DgmLen:40 ******** Seq: 0x0  Ack: 0x0  Win: 0x800
TcpLen: 20
```

I find it rather strange that ACK flag has a value of 0 for all 9 alerts. This can happen only in one of two scenarios. It can either be a crafted packet with ACK flag set to '0' value or the SYN packet that this packet is replying to, had the last possible value on the 32 bit field. There are no SYN and ACK flags or RST and ACK flags set. Thus, this traffic is not a reply to any SYN traffic. It rather fits the criteria of an Nmap null scan, which turns off all flags as the log shows above. The intention of this traffic is to determine the open ports on the target system. Target systems will respond with RST and ACK flags set on closed ports and possibly no reply on open ports. The only target of this activity has been 172.16.82.38 in all the alerts, which indicates an active targeting. Logs did not contain any other traffic from this attacking host. Neither did it contain any reply from host 172.16.82.38 to the scan. Parameters in the link posted under "Attacking IP 210.187.4.222" did not contain any information for window size 2048. However, I was able to find win size of 2048 in p0f.fp file and that did not relate to an operating system either. Win size of 2048 is associated with nmap scan and this entry in the p0f.fp file has a TTL value of 64, which makes the alert to have originated from 18 hops away. No information about an operating system was available for these parameters in p0f.fp file.
RIPE Whois query shows that host 130.240.13.82 belongs to Sweden.

```
inetnum:     130.240.0.0 - 130.240.255.255
netname:     LUTHNET
descr:       Lulea Institute of Technology
country:     SE
admin-c:     BJN1-RIPE
tech-c:      HR2491-RIPE
mnt-by:      SUNET-MNT
status:      ASSIGNED PI Definition
remarks:     ************************************************************
remarks:     *ABUSE CONTACT: abuse@luth.se IN CASE OF HACK ATTACKS,
remarks:     * ILLEGAL ACTIVITY, VIOLATION, SCANS, PROBES, SPAM, ETC.
remarks:     ************************************************************
changed:     ber@sunet.se 19960218
changed:     ripe-dbm@ripe.net 19990706
changed:     ripe-dbm@ripe.net 20000225
changed:     fredrik@sunet.se 20020913
source:      RIPE
```

Attacking host 65.57.64.224

Source IP 65.57.64.224 is suspicious, since numerous internal hosts have either made attempts to connect to or connected to this host on ports 6667, 6668 and 6669. Furthermore, scan logs reveal that this host has performed SYN scans on random ports on campus hosts with IP addresses of 172.16.202.14 and 172.16.226.178. I am unable to perform an OS detection due to lack of information on the alerts and lack of binary logs.

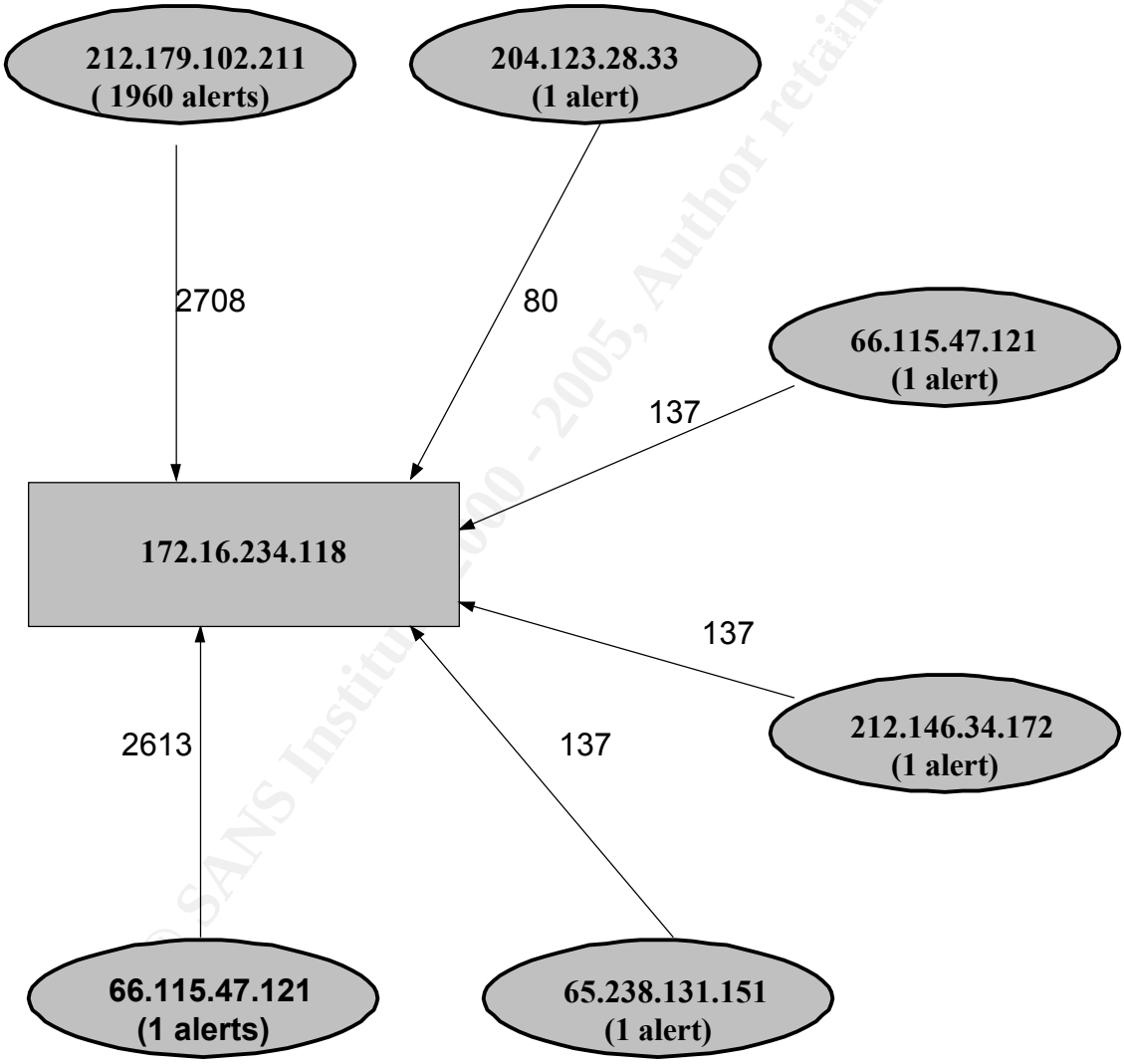
Following is the ARIN whois information for attacking IP 65.57.64.224.

```
OrgName:    Level 3 Communications, Inc.
OrgID:      LVLT
Address:    1025 Eldorado Blvd.
City:       Broomfield
StateProv:  CO
PostalCode: 80021
Country:    US
NetRange:   65.56.0.0 - 65.59.255.255
CIDR:       65.56.0.0/14
NetName:    LC-ORG-ARIN-BLK2
NetHandle:  NET-65-56-0-0-1
Parent:     NET-65-0-0-0-0
NetType:    Direct Allocation
NameServer: NS1.LEVEL3.NET
NameServer: NS2.LEVEL3.NET
Comment:    ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate:    2001-09-21
Updated:    2002-08-08
```

2.5 Link Graph

For easier analysis of the attacks directed at host 172.16.234.18, the following link graph has been provided with attack types, attacking hosts and ports involved. The attacks corresponding to the port numbers are the following.

(1) 2708 - Watchlist 000220 IL-ISDNNET-990517
(2) 80 - CS WEBSERVER - external web traffic
(3) 137 - SMB Name Wildcard
(4) 2613 - EXPLOIT x86 NOOP



Graph 2.1 Link graph displaying attacks directed towards 172.16.234.118

2.6 Possible systems compromised

By reviewing the alerts, several campus hosts need to be scanned for possible virus infection. Host 172.16.82.38 is exhibiting similar behavior of Nimda worm and is actively scanning numerous external hosts. This host should immediately be taken offline and scanned for Nimda infection.

Secondly there has been quite a few Red worm activity. Once a host is infected with Red worm, it opens port 65535 for the remote host to connect. Therefore, any connection attempts made to internal hosts on port 65535 should be suspicious and must be scanned for infection. Hosts with IP address 172.16.88.193, 172.16.202.82, 172.16.25.12, 172.16.20.118, 172.16.235.10, 172.16.236.50, 172.16.244.18 and 172.16.6.7 have exhibited similar behavior as Red worm and should be investigated. While this activity can be an infection, some of the Red worm traffic alerts are false positives. For example, the following activity appears to be part of Kazza file sharing using an 65535 as an ephemeral port.

```
02/08-13:50:44.354116  [**] High port 65535 tcp - possible Red Worm - traffic
[**] 172.16.205.226:1214 -> 80.48.89.20:65535
```

2.7 Recommendation

It appears that IDS is picking up a lot unnecessary traffic. Based on the analysis, few alerts can be avoided by proper tuning and security policy changes on the firewall.

Random SYN-FIN scans directed to campus hosts on port 21 were observed in the logs. Inbound traffic on port 21 should be blocked except the FTP server. In addition, upon further examining the null scan traffic, they have port 0 associated with source and destination. This looks like an nmap scan run against these hosts using –sN switch. With port set to 0, this ICMP traffic can either be an effort to map the network using TTL values or to cause DoS (Denial of Service) attack. Inbound ICMP traffic should be blocked on the perimeter firewall and no ICMP replies should be allowed outbound either.

There has been several Trojan server activity alerts upon detecting port 27374. This port is associated with Remote Access Trojans. Most of the alerts are for inbound scanning looking for an infected host to gain root access to those infected. Such inbound scan on port 27374 should be blocked.

P2P (peer–to-peer) file sharing activity has been one of the biggest threats in a university network due to numerous students involve in file sharing with users on the internet. Kazza by default uses port 1214 for file sharing. However, even if these ports are blocked at the firewall, this application will keep trying to get in through an open port. Since a few ports are open for inbound traffic, such as port 80, it is impossible to stop this activity. The only workaround to stop this

traffic has been to implement strict security policies. The use of file sharing application on the university network should not be allowed. As mentioned in the executive summary, university network is exposed a variety of threats due to file sharing application being in use. File sharing has also led to spread of malicious codes. According to a new study based on the following link, 45 percent of executable files downloaded through Kazaa contain malicious codes.

http://www.wired.com/news/business/0,1367,61852,00.html

Placement of IDS is very important in validating the alerts whether or not the alerts point to an attack or are just being noisy. I do not have a network diagram of the university network. Though the placement of IDS depends on the network layout, ideally, I recommend a minimum of two IDS devices being in place for monitoring a large network, placing the first one outside the firewall and the second one behind the firewall focusing more on the internal traffic. Needless to say, on very large networks, each internal network segments have an IDS dedicated or even HIDS monitoring activities on each host. From a cost effective standpoint, I also recommend having a Managed Security Services Provider (MSSP) manage the IDS. Based on the information from Information Security magazine [20], for in-house management, just the cost for personnel and maintenance is $250,000 the first year and $150,000 thereafter for one FTE (Full Time Equivalent) and this is not even real time monitoring. Whereas the total cost with an MSSP is less than $90,000 per year.

Most of the malicious traffic that are seen by the external IDS device will be blocked at the firewall. However, with a routine review of the logs on the external IDS device, security staff will be more clued in of any pre-attack activities and latest trends such as a new vulnerability. It will assist monitoring activities much easier.

Top three detects
Following is the analysis of the top three most critical alerts.

2.8 Detect 1 – EXPLOIT x86 NOOP

2.8.1 Description of detect

```
02/06-06:19:39.684460 [**] EXPLOIT x86 NOOP [**] 131.118.254.130:2304 ->
172.16.24.8:119

02/06-06:19:39.684707 [**] EXPLOIT x86 NOOP [**] 131.118.254.130:2304 ->
172.16.24.8:119
```

Table 7. Copy of alerts detected

This alert is triggered by snort signature upon detecting contiguous no operation codes. Snort signature looks for the content of “0x90” in transmission. A string of NOOP codes can cause a buffer overflow in x86 architecture machines and eventually resulting in a system compromise. The functionality of machine code 0x90 is to do nothing. Please refer to the link below for more information on 0x90 code.
http://weblogs.asp.net/oldnewthing/archive/2004/11/11/255800.aspx

While the intention of this traffic can be for evil purpose, this alert can very well be triggered on benign traffic, since NOOP codes are used to pad out the TCP options. Based on the analysis performed on this alert, there have been several external hosts targeting University hosts.

2.8.2 Reason this detect was selected

This buffer overflow exploit can result in a system compromise by sending a string of NOOP codes to vulnerable systems. Vulnerability also includes arbitrary code execution on the systems by a remote host. This type of activity should be seriously investigated, since an attacker can gain full control of the system by executing such codes.

2.8.3 Detect was generated by

By examining the pattern of the logs, the alerts were generated by a snort IDS device. I do not have access to the actual snort IDS device to review the snort rule that triggered the alert. Based on reference [17] I have put down a rule that could have very well triggered the alert with some modification to produce the alert above.

```
alert ip $EXTERNAL any -> $HOME any (msg:"EXPLOIT x86 NOOP";
content:"|90 90 90 90 90 90 90 90 90 90 90 90 90|";)
```

Table 8. Possible snort signature that triggered the alert

This signature looks for the content of “90” on traffic directed from external hosts towards 172.16.0.0 network. In such case, an alert will be triggered labeling the alert to be “EXPLOIT x86 NOOP”.

2.8.4 Probability the source IP was spoofed

In an event of TCP handshake requirement between the hosts involved, host IP is not spoofed. In other words, attacking IP cannot receive packets with SYN and ACK flags from the victim and communication cannot be established. Communication between the hosts has been constant in the logs. It is highly unlikely the source IP was spoofed.

2.8.5 Attack mechanism

As you can see from the alerts in section “attack description”, attacker has targeted port 119, which is for NNTP service. NNTP stands for Network News Transfer Protocol and is used for news articles to be stored in a database for people choose and read of their choice with proper authentication. I first verified if this host was still reachable and it was. I performed an nslookup and the telnet attempts on port 119 were successful. It has been verified that NNTP service is provided by this server. Please note that naming convention does not necessarily reveal the actual service.

```
8.24.85.130.in-addr.arpa        name = news.umbc.edu.
```

Attacking host 131.118.254.130 belongs to the university and this host has engaged in a longer communication each time. In the case of an actual attack, an attacker is likely to send in a series of 0x90 codes and engage in further exploit in a successful execution of this attack. There is no indication of any other suspicious activity from this host in any of the logs.

ARIN Whois Lookup for 131.118.254.130

```
OrgName:    University of Maryland
OrgID:      UNIVER-270
Address:    System Administration
Address:    3300 Metzerott Road
City:       Adelphi
StateProv:  MD
PostalCode: 20783
Country:    US
```

This alert could have very well been triggered by snort signature upon detecting 0x90 code in binary data, while transferring articles from the university NNTP server. Hence this alert is a false positive.

2.8.6 Correlations

Upon further examining the activity, based on the link below, it appears that this alert has a high number of false positives, especially on file transfers.

http://www.whitehats.com/info/IDS181

Some parts of the reports done by Vilaiporn Taweelappontong[3] and Terry MacDonald[2] have similarities to the analysis I have done.

2.8.7 Evidence of active targeting

Host 131.118.254.130 does not appear to be part of any other malicious activity in any of the logs provided. This alert more than likely is triggered on file transfer and has been ruled out to be a false positive. Hence, this is not active targeting.

2.8.8 Severity

The following formula was used for the calculation of severity.

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

Criticality (5)
Targeted system is a news server. Therefore, this is a critical target. Therefore, criticality has been rated as 5.

Lethality (5)
If the attack was successful, it will result in system compromise and attacker will gain access to the system with full privileges. Therefore, lethality has been rated as 5.

System countermeasures (2)
I am not able to verify if the systems are updated with latest patches with the information provided. However, by reviewing some of the security policies at the links below, I would assume that university security staff take proactive security measures including applying patching. I do not believe a host based IDS is in place.

http://www.umbc.edu/oit/sans/security/awareness/bestpractice.html
http://www.umbc.edu/oit/sans/security/index.html

Network countermeasures (1)
Access to this host on port 119 cannot be limited with a firewall policy or rule, since that would defeat the purpose of students being able to access the resource from anywhere. Firewall or any kind of perimeter defense at this time did not appear to have blocked any traffic related to this activity, since the firewall has to explicitly allow any activity on port 119 on this host. The only solution to such issue is to monitor network traffic using network intrusion detection system (NIDS) / or intrusion prevention system (IPS), which can monitor traffic and block any suspicious activity based on the rules. Network countermeasures has been rated as 1.

Severity = (5+5) –(2+1) = 7

Detect 2 - NIMDA - Attempt to execute cmd from campus host

2.9.1 Description of detect

NIMDA is a mass-mailing worm, which was first found on September 18, 2001. Vulnerable operating systems to this worm include Windows 95, Windows 98, Windows NT, Windows ME and Windows 2000. Based on the report from F-secure[8], it spreads via the following.

Email – It sends an email attachment with an executable file which, when downloaded install the worm itself to the host. The name of the executable file is README.EXE.

Network Shares – On a LAN environment, this worm can propagate via network shares copying itself to other workstations and servers.

File infection – As the name implies, it infects other files by appending itself to other executable files.

WWW – It also spreads by attempting to perform root traversal on webservers eventually compromising the host.

A snip of detects found in the logs is shown below.

```
02/06-08:45:07.350847 [**] NIMDA - Attempt to execute cmd from campus host
[**] 172.16.82.38:2998 -> 212.88.174.44:80

02/06-08:45:21.304409 [**] spp_http_decode: IIS Unicode attack detected [**]
172.16.82.38:3221 -> 130.223.7.118:80

02/06-08:45:07.535740 [**] NIMDA - Attempt to execute root from campus host
[**] 172.16.82.38:3002 -> 130.63.233.177:80
```

Logs above show that these are attempts to perform root traversal on webservers. Internal hosts 172.16.82.38 and 172.16.233.90 have been detected with the infection of Nimda worm and are actively scanning other webservers for vulnerability. Based on the report found at http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=PE%5FNIMDA%2EA&VSect=T for this vulnerability, these hosts are probably looking to execute commands on target computers and then force the target machines to download a copy of the worm as ADMIN.DLL and copy the file to the root directory. As you might have noticed already, the second alert in the table above has a description of IIS Unicode attack. In order to avoid being detected of directory traversal, unicodes are being used in place of '../'.

Please see http://www.unicode.org/ for more information on unicodes.

2.9.2 Reason this detect was selected

Since these hosts are actively scanning to infect other webservers, hosts involved in this attack are possibly compromised by Nimda worm and can result in a Denial of Service condition. In addition, compromised university hosts are webservers, which they do not seem to be live at the time of writing, any host visiting them will be infected by this worm as well.

2.9.3 Detect was generated by

These logs were generated by Snort intrusion detection appliance possibly snort version 1.8.7. I am not aware of any IDS implementation in this scenario. However based on the logs, I can only deduce a possible signature that could have triggered this alert based on snort signature database [23]. Following signatures have been modified produce the output above.

```
alert tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg:" NIMDA - Attempt to
execute cmd from campus host "; flow:to_server,established; content:"cmd.exe";
nocase; classtype:web-application-attack; sid:1002; rev:6;)

alert tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg:" spp_http_decode: IIS
Unicode attack detected "; flow:to_server,established; content:"/..%255c..";
nocase; reference:bugtraq,1806; reference:cve,2000-0884; classtype:web-
application-attack; reference:nessus,10537; sid:1945; rev:6;)
```

The snort signatures above indicate that any tcp traffic originating from an internal network (HOME_NET assumed to be defined as 130.85.0.0) with the content of cmd.exe and "/..%255c.." to any external networks on port 80 will be triggered with alerts "NIMDA - Attempt to execute cmd from campus host" and "spp_http_decode: IIS Unicode attack detected" respectively.

2.9.4 Probability the source address was spoofed

Source IP is not spoofed. Since two attacking host requires an http communication with the target host a TCP handshake must take place.

2.9.5 Attack Mechanism

Port 80 was the target in order to perform a directory traversal using the URL to the webserver. If such vulnerability exists on the webserver, this worm uses it to infect the host. As the snort alert indicates in the message as "Attempt to execute cmd from campus host", the mechanism can be anything similar to the following based on Trendmicro[9]. Due to space limitation the rest of it has been included in the Appendix A.

http://*ip_address*/c/winnt/system32/cmd.exe
http://*ip_address*/../../cmd.exe
http://*ip_address*/scripts/..%2f../winnt/system32/cmd.exe

All three attempts above are to traverse to system32 directory to run cmd prompt. First example is very straightforward as the directory traversal is obvious. Second and third have a few tweaks to evade the restrictions on the webserver. By default, the root directory in windows machines point to C:/inetpub/wwwroot/. Second example uses ../ to traverse backwards to C:/ and to gain access to command prompt. Third example uses Unicode to evade from being detected in place of "../..". There is not enough details on the logs to discover the mechanism used in this attack. However, there is a high probability that one or more of the above examples were used.

```
02/06-08:46:06.303822 [**] spp_http_decode: IIS Unicode attack detected [**]
172.16.82.38:4017 -> 198.12.19.179:80

02/06-08:46:06.303822 [**] NIMDA - Attempt to execute cmd from campus host
[**] 172.16.82.38:4017 -> 198.12.19.179:80
```

Alerts above indicate that they were both triggered at the very same time, which means that an attempt has been made on the target machine 198.12.19.179 to perform directory traversal using unicodes and to execute cmd.exe. While the logs indicate the campus hosts are compromised, this alert can be triggered on legitimate activity due to poor html coding. In other words, using ../ in the coding to access folder in other directories can cause the snort signature to trigger the alert. One of the best ways to validate the attack is to apply the exact string on the target host to see if the vulnerability exists. Due to lack of information in the logs, I cannot do this test.

If the target hosts were vulnerable IIS servers without the appropriate patches, the attack will indeed succeed. Links below provide the patches to secure the webservers from this attack.

Microsoft IIS 4.0:
http://www.microsoft.com/ntserver/nts/downloads/critical/q269862/default.asp

Microsoft IIS 5.0:
http://www.microsoft.com/windows2000/downloads/critical/q269862/default.asp

2.9.6 Correlations

Many write-ups have been done on Nimda worm activity and this worm is related to Code Red II worm as well, since Nimda uses the backdoor port left open by Code Red II to infect the webservers. CVE-2000-0884 also talks about

web server folder traversal vulnerability. Michael Meacle[16] has done a similar report.

```
Feb  6 08:46:08 172.16.82.38:3983 -> 130.106.27.33:80 SYN ******S*
Feb  6 08:46:08 172.16.82.38:3985 -> 130.6.26.53:80 SYN ******S*
Feb  6 08:46:08 172.16.82.38:3986 -> 130.90.165.240:80 SYN ******S*
```

SYN traffic shown above was found under scan logs.

This detect has been documented on CA-2001-26 [4], CVE-2001-0333 and CVE-2000-0884. Similar reports are available by Symantec [7], F-secure [8] and Trendmicro [9] as well.

2.9.7 Evidence of active targeting

As mentioned above, Nimda worm actively scans for other vulnerable hosts to infect. This activity is a scan and there is no evidence of any active targeting.

2.9.8 Severity

Criticality (3)
Due to lack of information about the host, it is hard to determine what services are running on this host and how critical the information on this host is.

Lethality (5)
This host is compromised and actively scanning other hosts for vulnerabilities. Once compromised, this worm is capable of creating open shares and guest accounts with administrator privileges.

System Countermeasures (3)
There is no evidence in the logs how this system was compromised. It could very well have gotten the worm via email. This activity was in progress on almost the entire day of February 6 2003. I believe there was no firewall appliance in place that defends against Nimda. However based on the following link, it appears university security staff take proactive security measures to avoid such infection by keeping up to date with patches and routinely updating virus signatures. Based on this information I am assuming this host was infected via email.

http://www.umbc.edu/oit/sans/security/index.html

Network countermeasures (1)
At the time of infection of this host, there were security products, which can defend against this activity. It does not appear such appliance was in place. Since I believe this infection took place via email, port 25 is allowed on the perimeter firewall for SMTP traffic.

Severity = (3+5) - (3+1) = 4

2.10 Detect 3 - External RPC call

2.10.1 Description of detect

External RPC call alerts are triggered, when external hosts perform a scan on the university hosts on port 111. Port 111 is associated with portmapper and is responsible for RPC services on a host. Inetd service runs on a standard port such as webserver running on port 80. However, RPC service runs on various ports. When an RPC service is started, portmapper converts the RPC service to a port number. Hence, if a client wishes to communicate to the server on a certain RPC service, it contacts the portmapper to determine the port number, so that packets can be directed to that port on the server.

```
02/08-09:32:20.241478 [**] External RPC call [**] 62.118.18.203:1808 ->
172.16.2.59:111
02/08-09:32:20.243695 [**] External RPC call [**] 62.118.18.203:2169 ->
172.16.3.22:111
02/08-09:32:20.914880 [**] External RPC call [**] 62.118.18.203:2334 ->
172.16.3.36:111
```

Alerts above show hosts 62.118.18.203 and 64.81.138.68 attempting to query university hosts to list RPC services being run on them. Once the attacker identifies services and corresponding target ports, an exploit on a vulnerable service can very well be underway.

2.10.2 Reason this detect was selected

There is a list of RPC services running on UNIX machines. Upon determining services running on the machine, a probe of this nature offers leads to various RPC services and hence various possible exploits on a target machine. For example, according to CERT Advisories, ToolTalk was actively being targeted by attackers, which is a vulnerable to buffer overflow attack and upon successful execution, attackers can gain root access to the host.

2.10.3 Detect was generated by

Based on the rpc.rules on Snort version 2.2.0 (Build 30), there is no signature implemented to be triggering an alert as shown above. Rather new developments seem to have a signature of its own for each service [22]. Reviewing the alert, I have come up with the following signature that triggered the alert.

```
Alert tcp $External any -> $Home 111 (msg:“External RPC call”;)
```

2.10.4 Probability the source IP was spoofed

It is unlikely that source IPs were spoofed. Since these alerts are for service reconnaissance, any replies from university hosts will not reach the attacker, if the scanning originated from a spoofed IP. A snip of SYN traffic from the scan logs below show further evidence of this activity.

```
Feb  7 06:36:42 64.81.138.68:4011 -> 172.16.240.20:111 SYN ******S*
Feb  7 06:36:42 64.81.138.68:4248 -> 172.16.240.70:111 SYN ******S*
Feb  8 09:34:29 62.118.18.203:3276 -> 172.16.169.217:111 SYN ******S*
```

2.10.5 Attack Mechanism

This is a pre-attack activity with the intention of gathering enough information to possibly launch an attack on the university network. Basically the attacker is sending SYN packets to the hosts on port 111 as an attempt to find out if port 111 is open on those hosts. Nmap command with –sR switch can also be used to figure out RPC ports and, their program numbers and version numbers.

2.10.6 Correlations

There is a list of CVE entries for vulnerability on RPC services. CVE-1999-0003, CVE-19999-0008 and CVE 1999-0969 entries are a few to note. CA-98.11 [11] and CA-2002-20 [12] have detailed information on Tooltalk, a well- known RPC service. Vilaiporn Taweelappontong [2] and SaiPrasad Kesavamatham [15] have worked on the similar detect and did not find any correlations. David P. Reece has a write-up on this activity and can be viewed at the following link.

http://www.sans.org/resources/idfaq/blocking.php

2.10.7 Evidence of active targeting

The table below lists a summary of the activity.

Source	No. of Alerts	No. of Alerts	# Dsts (sig)	# Dsts (total)
62.118.18.203	709	709	706	706
64.81.138.68	280	280	280	280
209.184.71.136	62	62	2	2
212.185.251.133	1	1	1	1

212.185.251.134	1	1	1	1

Table 2.6 Sources and destinations involved in the activity

Activity from 62.118.18.203, 64.81.138.68 and 209.184.71.136 do not appear to be active targeting. This activity is a reconnaissance scanning in an attempt to find hosts with port 111 open. No other alerts found in alert, scans and OOS logs contributing to active targeting by any other activity on university network from the these IPs.

Host 212.185.251.133 and 212.185.251.134 have targeted only host 172.16.100.158 with one attempt and the activities are about 28 minutes apart. Obviously, both hosts belong to Germany and possibly from the same attacker. Hence there is reasonable evidence to believe that this is active targeting. Port 111 should be blocked at the firewall for both udp and tcp traffic.

RIPE whois information for host 212.185.251.133 and 212.185.251.134

```
inetnum:      212.185.208.0 - 212.185.255.255
netname:      DTAG-DIAL9
descr:        Deutsche Telekom AG
country:      DE
admin-c:      DTIP
tech-c:       DTST
status:       ASSIGNED PA Definition
remarks:      ************************************************************
remarks:      * Abuse Contact: http://www.t-com.de/ip-abuse in case of Spam,   *
remarks:      * Hack Attacks, Illegal Activity, Violation, Scans, Probes, etc. *
remarks:      ************************************************************
mnt-by:       DTAG-NIC
changed:      ripe.dtip@telekom.de 19990305
changed:      ripe.dtip@telekom.de 20030211
changed:      ripe.dtip@telekom.de 20040709
source:       RIPE
```

2.10.8 Severity

Criticality (3)

External RPC call scans are mere probe. However, one of the ways to determine criticality on these systems is to detect if these hosts are UNIX based. We can rule out all the windows hosts being critical of this probe. Since none of the logs is in binary format, passive OS fingerprinting cannot be done to detect the operating system. Scan appears absolutely random and no evidence of any fingerprinting beforehand to be running the scan on UNIX machines.

Lethality (4)

If this probe succeeded, then the attacker will possibly be aware of the RPC services on the workstations, leading to a possible attack on the hosts depending on the RPC service and what it is vulnerable to.

System countermeasures (3)
With the information provided, I cannot determine any patches being in place on the target hosts. However, I assume system patches are up to date from the information available at the following link and I do not believe a host based IDS is in place.

http://www.umbc.edu/oit/sans/security/index.html

Network countermeasures (1)
Reviewing the OOS logs and alert logs, It appears that traffic on port 111 was allowed through the firewall.

Severity = (3+4) – (3+1) = 3

Part 3. Analysis Process

The logs were obtained from http://isc.sans.org/logs. The operating system used was a Redhat Linux 9.0 on Dell 1550 PowerEdge. Logs were processed using "SnortSnarf version 021111.1" to break down logs to an html file, which categorized the alerts based on the signature section, top 20 source IPs and top 20 destination IPs. I combined all five days worth of log using 'cat' command to avoid having to process them separately. Data on the files had a MY.NET variable, which must have been set on the snort.conf file to define the primary university network. Since SnortSnarf requires numbers on all four octets of the host address, MY.NET variable was replaced with 172.16 using command 'sed'. Syntax of the commands used are shown in Appendix A. Following is the command used for processing the logs using Snortsnarf.

./snortsnarf.pl –rs /tmp/GCIA/alert_files –d /tmp/GIAC_alert/

-rs switch puts the alarms in the order of most interesting first
–d defines the destination path for the output to be written into

The alert logs of five days are 16MB in size and processing them was very time consuming due to high RAM usage by SnortSnarf. SnortSnarf generated 157201 alerts from alert logs. 'spp_portscan' alerts were found in both alert and scan logs. Snortsnarf did not process the spp_portscan logs due to bad alert format and were not used in analysis from the logs, however were processed and used for analysis from scan logs.

Scan and OOS logs were relatively smaller in size and they were 6.11MB and

3.74 MB respectively and therefore processing was much easier than alert logs. OOS logs were modified using the perl script written by Ricky Smith[15] so that they can be processed using Snortsnarf and the script is attached in Appendix A. 130.85 netblock found in the scan logs were replaced with 172.16 for consistency throughout the logs. Analyzing the alert logs for the top most targeted ports was difficult, since alerts are not aligned well due to various event names. I have listed below the process and obstacles I had in this part of the analysis.

(1) Alerts with the destination IPs in the range of 172.16.0.0 were separated using "**cat <alert_file> | grep "-> 172.16" > <alert_port_filename>** "

(2) I used the following alert as a standard for 'awk' command.
02/06-00:15:31.001599 [**] SMB Name Wildcard [**] 68.113.37.91:1027 -> 172.16.82.119:137
more <file_name> | awk '{print $1 " " $3" " $4 " " $5 " " $7 " "$9}'
However, alert file contains various event names, which do not work with the
variables set above. For example, if the alert name was 'CGI Null Byte Attack Detected', command used above will not provide the output we need.

(3) Therefore I separated the portion with destination IP and port number from each alert. The only variable to work with was '>' and I used a simple PERL script to separate them, which is attacked in Appendix A.

(4) Once they were separated, they were in the format of **<ip_address>:<port_number>.** I used 'sed' with the following parameters '**sed s/:/" "/g alert_port_filename > new_alert_port_file'** to replace the colon with a space. Then I used the awk to get the port count for each port.

'cat <file_name> | awk '{print $2}' | sort | uniq -c > <port-count>'

Reference

(1) Zetter, Kim. “Kazaa Delivers More Than Tunes.” URL: http://www.wired.com/news/business/0,1367,61852,00.html (January 2005).

(2) Taweelappontong, Vilaiporn. “SANS GCIA Practical Assignment.” URL: http://www.giac.org/practical/GCIA/Vilaiporn_Taweelappontong_GCIA.pdf (January 2005)

(3) MacDonald, Terry. “Intrusion Detection and Analysis: An Investigation.” URL http://www.giac.org/practical/GCIA/Terry_MacDonald_GCIA.pdf (January 2005):52-53.

(4) CERT coordination center. “CERT Advisory CA-2001-26 Nimda Worm.” URL: http://www.cert.org/advisories/CA-2001-26.html (December 2004).

(5) Internet Assigned Numbers Authority. “Root-Zone Whois Information.” URL:http://www.iana.org/cctld/cctld-whois.htm (January 2005).

(6) “Draft - Sun Remote Procedure Call.” URL: http://probing.csx.cam.ac.uk/about/sunrpc.html (January 2005).

(7) Symantec. “W32.Nimda.A@mm.” URL: http://securityresponse.symantec.com/avcenter/venc/data/w32.nimda.a@mm.html (January 2005).

(8) F-secure. “F-Secure Virus Descriptions – Nimda.” URL:http://www.f-secure.com/v-descs/nimda.shtml (January 2005).

(9) Trend Micro. “PE_NIMDA.A.” URL: http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=PE%5FNIMDA%2EA&VSect=T (January 2005)

(11) CERT coordination center. “CERT Advisory CA-98.11.” URL: http://www.cert.org/advisories/CA-98.11.tooltalk.html (January 2005).

(12) CERT coordination center. “CERT Advisory CA-2002-20 multiple Vulnerabilities in CDE Tooltalk.” URL:http://www.cert.org/advisories/CA-2002-20.html (January 2005).

(13) Kesavamatham, SaiPrasad. “External RPC call.” URL: http://www.giac.org/practical/GCIA/SaiPrasad_Kesavamatham_GCIA.pdf (January 2005): 54-55.

(14) Spitzner, Lance. “Lists of fingerprints for passive fingerprint monitoring.” URL:http://www.honeynet.org/papers/finger/traces.txt (January 2005).

(15) Smith, Ricky. “Perl script: Parse-oos.pl.” URL:http://www.giac.org/practical/GCIA/Ricky_Smith_GCIA.pdf (January 2005): 68.

(16) Michael, Meacle. “ Detect #2 nimda.” URL: http://www.giac.org/practical/GCIA/Michael_Meacle_GCIA.pdf (January 2005): 33-36.

(17) Snort. “Snort signature database.” URL: http://www.snort.org/snort-db/sid.html?sid=648 (January 2005).

(18) Link Logger. “TCP Port 27374.” URL:http://www.linklogger.com/TCP27374.htm (January 2005).

(19) Kantor Brian, Lapsley Phil. “Network News Transfer Protocol.” URL:http://www.w3.org/Protocols/rfc977/rfc977

(20) Stone, Adam. “In MSSP We Trust.” Information Security. (February 2005): 40-47.

(21) Internet Assigned Numbers Authority. “Port Numbers.” URL: http://www.iana.org/assignments/port-numbers (February 2005).

(22) Snort. “Snort signature database.” URL: http://www.snort.org/cgi-bin/sigs-search.cgi?sid=RPC (January 2005)

(23) Snort. “ Snort signature database.” URL: http://www.snort.org/snort-db/sid.html?sid=1002 (January 2005)

Appendix A

Syntax of the commands used :

sed s/MY.NET/172.16/g alert > alert_file

cat alert alert_day_1 alert_day_2 alert_day_3 alert_day_4 alert_day_5 > alert_input

```
#  Lists of fingerprints for passive fingerprint monitoring
#  Updated 23 May, 2000
#  Mail your signatures to Lance Spitzner <lance@spitzner.net>

# OS            VERSION PLATFORM        TTL     WINDOW          DF      TOS
#---            ------- --------        ---     ------------    --      ---

DC-OSx           1.1-95  Pyramid/NILE    30      8192            n
0

Windows         9x/NT   Intel           32      5000-9000       y       0

NetApp          OnTap   5.1.2-5.2.2     54      8760            y
0

HPJetDirect      ?       HP_Printer      59      2100-2150       n
0

AIX             4.3.x   IBM/RS6000      60      16000-16100     y
0
AIX             4.2.x   IBM/RS6000      60      16000-16100     n
0
Cisco            11.2    7507            60      65535           y
0
DigitalUnix     4.0     Alpha           60      33580           y       16
IRIX            6.x     SGI             60      61320           y
16
OS390            2.6     IBM/S390        60      32756           n
0
Reliant          5.43    Pyramid/RM1000  60      65534           n
0

FreeBSD         3.x     Intel           64      17520           y       16
JetDirect       G.07.x  J3113A          64      5804-5840       n       0
Linux           2.2.x   Intel           64      32120           y       0
OpenBSD         2.x     Intel           64      17520           n       16
OS/400          R4.4    AS/400          64      8192            y       0
SCO             R5      Compaq          64      24820           n       0
Solaris         8       Intel/Sparc     64      24820           y
0
FTX(UNIX)        3.3     STRATUS         64      32768           n
0
Unisys          x       Mainframe       64      32768           n       0

Netware         4.11    Intel           128     32000-32768     y       0
Windows         9x/NT   Intel           128     5000-9000       y       0
Windows         2000    Intel           128     17000-18000     y
0

Cisco           12.0    2514            255     3800-5000       n       192
Solaris         2.x     Intel/Sparc     255     8760            y       0
```

Following is the perl script used for parsing OOS logs, which was written by Ricky Smith. I give him full credit for it.

```
#! /usr/bin/perl

# use strict;

##########
#
#       Author:  Rick Smith
#         Date:  13 Dec 2002
#      Version:  1.0
#
#####
#
#      Purpose:  Parse the OOS log files and connvert to tab-delimited
#                for import into Excel spreadsheets.
#
#  Description:  1. Uses the specified oos log file to creates a parsed-<oos-
file>.txt
#                  in the current directory which is ready for use with SnortSnarf.
#
#        Usage:  parse-oos.pl <path_to_log>/<oos file>
#
#   Parameters:  1.  the oos file to parse including the full path
#
##########

#############
## MAIN
#############

##Initialize variables.
$logfile = 0;
$resultsfilename = 0;
$linecount = 0;

# Check for null input
if (! $ARGV[0])  {
          die "Usage: parse-oos.pl <path_to_logs>/<oos file>\n\n";
};  #if

## Get the oos file name and path to the log files from command line.
$logfile = @ARGV[0];

open (LOG, $logfile) || warn "Cannot open $logfile : $!";

$resultsfile = ">parsed-". $logfile ."oos.txt";
open RESULTS, $resultsfile || die "Cannot open $resultsfile : $!";



## Parse through the log files.
## Sequentially read in the lines from      the log file and print to the RESULTS
## file if the line in the correct format   for SnortSnarf
while (<LOG>) {
```

One of the mostly used commands was ‘grep’ with the following options. Some of the commands used were the following to further dig down on a certain alert. Commands “more” or “less” can be used instead of “cat” for the following.

To separate only EXPLOIT x86 NOOP alerts:
cat alert_input | grep " EXPLOIT x86 NOOP " > EXPLOIT_x86_NOOP

For alerts without EXPLOIT x86 NOOP
cat alert_input | grep –v " EXPLOIT –v x86 NOOP " > Rest_of_the_Alert

Script used for separating the destination port numbers from alert file.

```
#dest_port_script.pl
#Command usage : perl dest_port_script.pl input_file > output_file

#!/usr/bin/perl

while (<>) { #01
        @dest_port=split(/\-\>/);;

        print $dest_port[1];
}
```

Nimda

Following are some the strings used with URL by Nimda worm.

```
/scripts
/MSADC
/c
/d
/scripts/..%255c..
/_vti_bin/..%255c../..%255c../..%255c..
/_mem_bin/..%255c../..%255c../..%255c..
/msadc/..%255c../..%255c../..%255c/..%c1%1c../
..%c1%1c../..%c1%1c..
/scripts/..%c1%1c.
/scripts/..%c0%2f.
/scripts/..%c1%9c..
/scripts/..%%35%63..
/scripts/..%%35c..
/scripts/..%25%35%63..
/scripts/..%252f..
/winnt/system32/cmd.exe?/c+
/root.exe?/c+
```