Global Information Assurance Certification Paper

SANS GCIA Practical Assignment

Michael Holstein

Version 3.1

Table of Contents

Assignment 1 – Use of Fragroute to evade NIDS detection

Introduction

Network based Intrusion Detection Systems (NIDS) are typically configured to passively monitor network traffic on a segment by way of a hardware tap or other tactic such as use of the switchport-monitor command (Cisco IOS) allowing the NIDS to monitor, and in some cases, inject traffic for all hosts and destinations passing through the segment.

Most NIDS systems are pattern based, requiring a large set (typically ~1500+) signatures to alert based on a specific combination of TCP flags in the header, or a set pattern in the payload. The accuracy of this approach depends, of course, on the skill of the administrator writing the signature, but in most cases this provides for very accurate detection of a specific attack, and will not catch new or modified attacks.

Statistically based NIDS systems, which are usually used in conjunction with pattern matching, tries to establish a baseline of activity and alert when packets are "statistically significant" in their deviation from the norm – a mathematical way of saying "weird packet". Unlike pattern matching, this tactic can catch new (and only occasionally, more creative) attacks at the cost of being rather noisy and requiring human analysis of all alerts.

Because most NIDS systems operate in layer 2 (OSI), they simply feed raw traffic into a detection engine and rely on the pattern matching and/or statistical analysis to determine what is malicious. Packets are not processed by the host's TCP/IP stack – allowing the NIDS to analyze traffic the host would otherwise discard. This approach also has the disadvantage that packets can be intentionally crafted in such a way as to confuse pattern-matching NIDS systems, while still being correctly assembled by the host TCP/IP stack to render the attack payload.

Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection, written by Ptacek & Newsham (1998), details a number of these attack methods, which are summarized below. The techniques described in Ptacek & Newsham were used by programmer Dug Song to create Fragroute.

Fragroute, by its own assertion [man(8) page], "…intercepts, modifies, and rewrites egress traffic destined for the specified host, implementing most of the attacks described in the Secure Networks "Insertion, Evasion, and Denial of Service 'Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection' paper of January 1998."

Terms and Conventions used in this document

Software :

Snort : Network Intrusion Detection (NIDS). www.snort.org/dl/snort-1.8.6.tar.gz
Tcpdump : Packet capture utility.
www.tcpdump.org/release/tcpdump-3.7.1.tar.gz
Ethereal : Packet analysis utility.
www.ethereal.com/distribution/ethereal-0.9.3.tar.gz
Fragroute : Packet shaper.
www.monkey.org/~dugsong/fragroute/fragroute-1.2.tar.gz

Obfuscation : source and destination hosts/networks are aliased as follows :

Attack.source : host initiating the attack
Attach.target : host running the daemon under attack.

Session logs : mathematical operands are used to indicate direction of communication :

'>' : commands issued from the attack.source
'<' : command results returned from attack.server

How It Works

To determine the effectiveness of Fragroute in obscuring a potential attack, three hosts were used : one running fragroute as the source, a second running wu-ftpd as the target, and a third running Tcpdump, Snort, and Ethereal for capture and analysis. All three hosts were connected to an isolated network segment.

Because the purpose of this analysis was the evasion technique and not the attack itself, I chose a common FTP exploit – attempting to "cd ~root" while authenticated as an unprivileged user. This exploit is well documented [CVE-1999-0082] and reliably detected by most NIDS systems.

It involves the following commands (comments indicate where packet logging started and stopped for all examples which follow) :

```
Attack.source> ftp attack.target
               < 220 attack.target FTP server ready
               > user unprivileged
               < 331 password required for unprivileged
               > pass mypassword
               < 230 user unprivileged logged in
               > cd ~root                                  #network trace begins
               < 250 CWD command successful                #network trace ends
```

For a baseline, the above sequence (logged where indicated) was executed without the use of Fragroute using Tcpdump for capture and Ethereal for analysis :

```
# Time        Source        Destination   Protocol Info
1 0.000000    attack.source attack.target FTP      Request: CWD ~root
2 0.000000    attack.target attack.source FTP      Response: 250 CWD command
                                                   successful.
3 0.000000    attack.source attack.target TCP      42579 > ftp [ACK] Seq=1530339426
                                                   Ack=148953486 Win=5840 Len=0
```

Snort immediately complained :

```
#(4 - 164) [2002-05-02 18:48:47] [CVE/CVE-1999-0082] [arachNIDS/318]  FTP CWD ~root
attempt
IPv4: attack.source -> attack.target
      hlen=5 TOS=0 dlen=51 ID=9657 flags=0 offset=0 TTL=127 chksum=4493
TCP:  port=1406 -> dport: 21  flags=***AP*** seq=82430654
      ack=3320914616 off=5 res=0 win=16436 urp=0 chksum=63938
Payload:  length = 11

000 : 43 57 44 20 7E 72 6F 6F 74 0D 0A                CWD ~root..
```

The attack was then repeated using Fragroute to obscure the attempt. The standard ruleset (provided when Fragroute is compiled) was used for testing. The function of each rule is explained as comments :

```
Tcp_seg 1 new      #break each TCP data segment into 1 byte pieces, favor new data vs
                   old.
Ip_frag 24         #break each IP packet into 24 byte fragments, preserving original
                   header.
Ip_chaf dup        #interleave duplicate packets with bogus payloads or invalid IP
                   options.
Order random       #reorder packets in queue randomly for transmission.
Print              #log to STDOUT as we go.
```

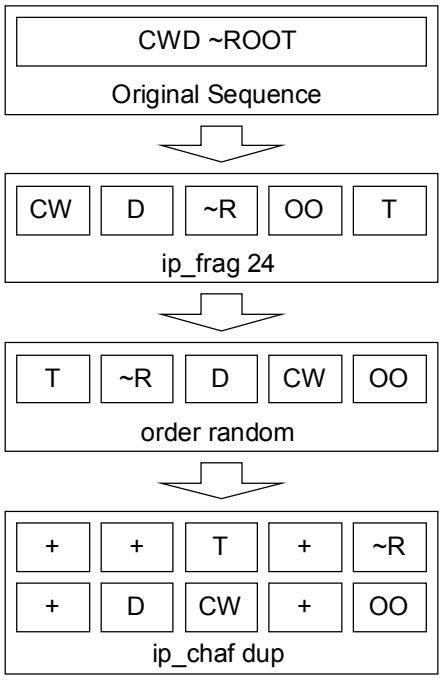
The session was again logged with Tcpdump and analyzed with Ethereal :

```
#  Time        SRC    DST    Pro     DATA
1  0.000000    SOURCE TARGET FTP     Request: \000
2  0.000000    SOURCE TARGET FTP     Request:
3  0.000000    SOURCE TARGET FTP     Request: \000
4  0.000000    SOURCE TARGET FTP     Request: \000
5  0.000000    SOURCE TARGET FTP     Request: \000
6  0.000000    SOURCE TARGET FTP     Request: \000
7  0.000000    SOURCE TARGET FTP     Request: t
8  0.000000    SOURCE TARGET TCP     27244 > 12138 [FIN, ACK, URG, ECN] Seq=1249539147
                                     Ack=1481190196 Win=27989, bogus TCP header lngth
                                     (16,
                                     must be at least 20)
9  0.000000    SOURCE TARGET TCP     17527 > 22646 [RST, URG, ECN] Seq=843544945
                                     Ack=2003986504 Win=26704 Urg=21332
                                     Len=4294967289[Malformed Packet]
10 0.000000    SOURCE TARGET TCP     1829 > 14678 [FIN, SYN, RST, PSH, ECN]
                                     Seq=1198864246
                                     Ack=2004438608 Win=29296 Len=2
11 0.000000    SOURCE TARGET FTP     Request: ~r
12 0.000000    SOURCE TARGET TCP     19817 > 20814 [PSH, ACK, URG, ECN] Seq=2054768737
                                     Ack=1163407946 Win=19030, bogus TCP header
                                     length (16, must be at least 20)
13 0.000000    SOURCE TARGET TCP     21862 > 12139 [FIN, SYN, PSH, URG, ECN]
                                     Seq=1919833172
                                     Ack=1432513857 Win=18227 Urg=22614 Len=2
14 0.000000    SOURCE TARGET TCP     14414 > 18520 [SYN, RST, ECN] Seq=1867985495
```

```
                                        Ack=1450783842 Win=30830, bogus TCP header
                                        length (16, must be at least 20)
15 0.000000    SOURCE TARGET TCP        28484 > 12615 [ACK, URG] Seq=1934651767
                                        Ack=1198740550 Win=16715 Urg=23143
                                        Len=4294967293 [Unreassembled Packet]
16 0.000000    SOURCE TARGET TCP        28786 > 28529 [FIN, SYN, PSH, ECN] Seq=1431253841
                                        Ack=1114855754 Win=19058 Len=4294967293
                                        [Unreassembled Packet]
17 0.000000    SOURCE TARGET TCP        17775 > 30535 [FIN, ACK, URG, ECN] Seq=1364159342
                                        Ack=846033235 Win=20549, bogus TCP header length
                                        (16, must be at least 20)
18 0.000000    SOURCE TARGET FTP        Request: D
19 0.000000    SOURCE TARGET FTP        Request: CW
20 0.000000    SOURCE TARGET TCP        16722 > 12151 [PSH, ACK, ECN] Seq=1447833401
                                        Ack=1331901540 Win=25683 Len=4294967290
                                        [Unreassembled Packet]
21 0.000000    SOURCE TARGET TCP        6423 > 17498 [FIN, SYN, RST, ECN] Seq=1953068616
                                        Ack=2004244804 Win=17779 Len=1
22 0.000000    SOURCE TARGET FTP        Request: oo
23 0.000000    TARGET SOURCE TCP        ftp > 41509 [ACK] Seq=917209621 Ack=2334649304
                                        Win=16560 Len=0
24 0.000000    TARGET SOURCE TCP        ftp > 41509 [ACK] Seq=917209621 Ack=2334649304
                                        Win=16560 Len=0
25 0.000000    TARGET SOURCE TCP        ftp > 41509 [ACK] Seq=917209621 Ack=2334649304
                                        Win=16560 Len=0
26 0.000000    TARGET SOURCE TCP        ftp > 41509 [ACK] Seq=917209621 Ack=2334649304
                                        Win=16560 Len=0
27 0.000000    TARGET SOURCE TCP        ftp > 41509 [ACK] Seq=917209621 Ack=2334649304
                                        Win=16560 Len=0
28 0.000000    TARGET SOURCE TCP        12139 > 21862 [RST, ACK] Seq=1432513857
                                        Ack=1919833173 Win=18227 Len=2
29 0.000000    TARGET SOURCE TCP        ftp > 41509 [ACK] Seq=917209621 Ack=2334649304
                                        Win=16560 Len=0
30 0.000000    TARGET SOURCE TCP        ftp > 41509 [ACK] Seq=917209621 Ack=2334649304
                                        Win=16560 Len=0
31 0.000000    TARGET SOURCE TCP        ftp > 41509 [ACK] Seq=917209621 Ack=2334649304
                                        Win=16560 Len=0
32 0.000000    TARGET SOURCE TCP        ftp > 41509 [ACK] Seq=917209621 Ack=2334649304
                                        Win=16560 Len=0
33 0.000000    TARGET SOURCE TCP        ftp > 41509 [ACK] Seq=917209621 Ack=2334649310
                                        Win=16554 Len=0
34 0.000000    TARGET SOURCE TCP        ftp > 41509 [ACK] Seq=917209621 Ack=2334649315
                                        Win=16555 Len=0
35 0.000000    TARGET SOURCE FTP        Response: 250 CWD command successful.
36 0.010000    SOURCE TARGET TCP        41509 > ftp [ACK] Seq=2334649315 Ack=917209650
                                        Win=5840 Len=0
37 0.010000    SOURCE TARGET TCP        26951 > 13398 [SYN, ECN] Seq=1750290291
                                        Ack=1413969516 Win=13902 Len=0
38 0.010000    TARGET SOURCE TCP        13398 > 26951 [RST, ACK] Seq=1413969516
                                        Ack=1750290292 Win=13902 Len=0
```

A request/response which would typically require only 3 packets now uses 38. Our original request of "cd ~root" is sent out of order in packets 7, 11, 18, 19 and 22 with 1 or 2 byte payloads. Packets 1, 2, 3, 4, 5, 6, 7 are duplicate "chaf" packets issued as part of the FTP session.

The remaining packets from the attack.source are"chaf" packets with a variety of problems, including short headers, invalid checksums, or are duplicates. Packets from the attack.target returned are ACKs for the chaf packets which correctly checksumed by the remote IP stack.



Our original attack sequence fits within one packet.

Using the "ip_frag 24" option splits the sequence into packets with payloads of 24 bytes or less

Using the "order random" option puts the fragments in random order.

Using the "ip chaf dup" option inserts duplicate packets copying the header from a valid portion of the stream, but with invalid TCP options, garbage payloads, or invalid checksums.

The fragmented stream was correctly reassembled by the target's IP stack, resulting in the "250" success command in packet 35. Fragroute does not manipulate reverse traffic.

Snort –1.8.6 failed to detect any elements of the attempt.

Nature of the threat

The thought of a potential attacker being able to download an 83k of software and make themselves invisible to a well-laid and meticulously maintained network of security hardware and software would agitate even the most sedate of security staff. Intrusion detection systems provide valuable warning as potential threats test your network, and (usually) provide the evidence to figure out what happened if they beat you at finding something of interest.

According to Marty Roesch, snort 1.9 (currently under development) "…deals with some of the more interesting attacks from fragroute…" (Roesch, 1). Testing this theory involved compiling snort-current from CVS and replaying the same tcpdump file used previously through it using snortrules-current, also from CVS. Snort detected some of the "chaf" fragments as a portscan, and the responses from garbage packets as "Evasive RST" – neither of which identifies the original attack. Tracking snort-current will address the issue eventually, but at present it appears that NIDS systems are still unable to cope with an attack wrapped by Fragroute.

```
[**] [100:1:1] spp_portscan: PORTSCAN DETECTED to port 22646 from attack.source
(STEALTH) [**] 05/06-11:30:43.912934

[**] [111:2:1] spp_stream4: possible EVASIVE RST detection [**]
05/02-20:58:16.589253 attack.target:12139 -> attack.source:21862
```

```
TCP TTL:59 TOS:0x10 ID:47366 IpLen:20 DgmLen:42 DF
***A*R** Seq: 0x55626D41  Ack: 0x726E5455  Win: 0x4733  TcpLen: 20

[**] [111:2:1] spp_stream4: possible EVASIVE RST detection [**]
05/02-20:58:16.599253 attack.target:13398 -> attack.source:26951
TCP TTL:59 TOS:0x10 ID:49947 IpLen:20 DgmLen:40 DF
***A*R** Seq: 0x5447766C  Ack: 0x68534F74  Win: 0x364E  TcpLen: 20
```

Possible Solutions To The Vulnerability

- Use a host-based IDS system on exposed systems. Host based IDS systems are able to detect malicious activity by monitoring at the application layer, and are able to report on entries created in the system or access logs. Logsnorter is one such example [www.snort.org/dl/contrib./logsnorter-0.2.tar.gz].
- Upgrade your NIDS software. Vendors are presently scrambling to address the issues created by Fragroute and will figure it out eventually.

References

Lemos, Robert. New tool camouflages hacker programs. ZdNet Australia. 22 April 2002. http://www.zdnet.com.au/newstech/security/story/0,2000024985,20264745,00.htm

Mitre. Common Vulnerabilities and Exposures. 27 August 1999. http://cve.mitre.org/cgi-bin/cvename.cgi?name=1999-0082

Ptacek, Thomas & Newsham, Timothy. "Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection". Secure Networks, January 1998. http://www.insecure.org/stf/secnet_ids/secnet_ids.html

Roesch, Marty. News. 7 May 2002. www.snort.org/index.html

Song, Dug. "Fragroute(8)" http://www.monkey.org/~dugsong/fragroute/fragroute.8.txt

Timm, Kevin.IDS Evasion Techniques and Tactics. SecurityFocus (Infocus). 7 May, 2002 http://online.securityfocus.com/infocus/1577

Assignment 2 – Network Detects

Detect 1 – Formmail CGI attempts

```
---------------------------------------------------------------------------
#(25 - 182) [2002-04-16 18:05:35] [Bugtraq/1187] [CVE/CVE-1999-0172]
[arachNIDS/226]  WEB-CGI formmail access
IPv4: 206.133.210.27 -> MY.NET.200.90
      hlen=5 TOS=0 dlen=440 ID=64201 flags=0 offset=0 TTL=116 chksum=55921
TCP:  port=2268 -> dport: 80  flags=***AP*** seq=208899197
      ack=1247658862 off=5 res=0 win=5840 urp=0 chksum=28989
Payload:  length = 400

000 : 47 45 54 20 2F 63 67 69 2D 62 69 6E 2F 66 6F 72   GET /cgi-bin/for
010 : 6D 6D 61 69 6C 2E 70 6C 3F 72 65 63 69 70 69 65   mmail.pl?recipie
020 : 6E 74 3D 41 6E 74 69 41 72 61 62 4C 65 61 67 75   nt=AntiArabLeagu
030 : 65 40 61 6F 6C 2E 63 6F 6D 26 73 75 62 6A 65 63   e@aol.com&subjec
040 : 74 3D 68 74 74 70 3A 2F 2F 78 78 78 78 78 78 78   t=http:xxxxxxxxx
050 : 78 78 78 78 78 78 78 78 78 78 78 78 78 78 78 2F   xxxxxxxxxxxxxx/
060 : 63 67 69 2D 62 69 6E 2F 66 6F 72 6D 6D 61 69 6C   cgi-bin/formmail
070 : 2E 70 6C 26 62 6F 64 79 3D 4A 75 70 5A 26 65 6D   .pl&body=JupZ&em
080 : 61 69 6C 3D 63 61 66 40 61 6F 6C 2E 63 6F 6D 20   ail=caf@aol.com
090 : 48 54 54 50 2F 31 2E 31 0D 0A 41 63 63 65 70 74   HTTP/1.1..Accept
0a0 : 3A 20 69 6D 61 67 65 2F 67 69 66 2C 20 69 6D 61   : image/gif, ima
0b0 : 67 65 2F 78 2D 78 62 69 74 6D 61 70 2C 20 69 6D   ge/x-xbitmap, im
0c0 : 61 67 65 2F 6A 70 65 67 2C 20 69 6D 61 67 65 2F   age/jpeg, image/
0d0 : 70 6A 70 65 67 2C 20 2A 2F 2A 0D 0A 41 63 63 65   pjpeg, */*..Acce
0e0 : 70 74 2D 4C 61 6E 67 75 61 67 65 3A 20 65 6E 2D   pt-Language: en-
0f0 : 75 73 0D 0A 41 63 63 65 70 74 2D 45 6E 63 6F 64   us..Accept-Encod
100 : 69 6E 67 3A 20 67 7A 69 70 2C 20 64 65 66 6C 61   ing: gzip, defla
110 : 74 65 0D 0A 55 73 65 72 2D 41 67 65 6E 74 3A 20   te..User-Agent:
120 : 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D   Mozilla/4.0 (com
130 : 70 61 74 69 62 6C 65 3B 20 4D 53 49 45 20 35 2E   patible; MSIE 5.
140 : 30 3B 20 57 69 6E 64 6F 77 73 20 39 38 3B 20 44   0; Windows 98; D
150 : 69 67 45 78 74 29 0D 0A 48 6F 73 74 3A 20 78 78   igExt)..Host: xx
160 : 78 78 78 78 78 78 78 78 78 78 78 78 78 78 78 78   xxxxxxxxxxxxxxxx
170 : 6E 2E 75 73 0D 0A 43 6F 6E 6E 65 63 74 69 6F 6E   xxxx..Connection
180 : 78 78 78 78 65 70 2D 41 6C 69 76 65 0D 0A 0D 0A   : Keep-Alive....
```

Identity of offending source

```
[user@server user]$ whois 206.133.210.27@whois.arin.net
[whois.arin.net]
Sprint (NETBLK-NETBLK-SPRINTSOI-BLKA)
   13221 Woodland Park Road
   Herndon, VA 22071
   US

   Netname: NETBLK-SPRINTSOI-BLKA
   Netblock: 206.133.0.0 - 206.133.255.255
   Maintainer: SPRN

   Coordinator:
      Sprintlink (Sprint)  (SPRINT-NOC-ARIN)  NOC@SPRINT.NET
      800-232-6895

   Domain System inverse mapping provided by:

   NS1.DIALSPRINT.NET           206.134.151.45
   NS2.DIALSPRINT.NET           206.134.79.44
   NS3.DIALSPRINT.NET           205.149.192.145

   ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

   Record last updated on 14-Jun-1998.
   Database last updated on  12-May-2002 19:57:36 EDT.
```

```
[user@server user]$ nslookup 206.133.210.27 ns1.dialsprint.net
Server:         ns1.dialsprint.net
Address:        206.134.151.45#53

27.210.133.206.in-addr.arpa     name = sdn-ar-005nvlvegP289.dialsprint.net.
```

Source Of Trace

Author's Network (class B netblock via DS-3).

Detect Was Generated By

Snort NIDS (version 1.9)

Log Format :

```
-----------------------------------------------------------------------------
#(25 - 182) [2002-04-16 18:05:35] [Bugtraq/1187] [CVE/CVE-1999-0172]
[arachNIDS/226]  WEB-CGI formmail access
IPv4: 206.133.210.27 -> MY.NET.200.90
      hlen=5 TOS=0 dlen=440 ID=64201 flags=0 offset=0 TTL=116 chksum=55921
TCP:  port=2268 -> dport: 80  flags=***AP*** seq=208899197
      ack=1247658862 off=5 res=0 win=5840 urp=0 chksum=28989
Payload:  length = 400

000 : 47 45 54 20 2F 63 67 69 2D 62 69 6E 2F 66 6F 72   GET /cgi-bin/for
```

Time and Date: April 16th, 2002 @ 18:05:35 EST (GMT – 05:00)
Alert Name: WEB-CGI formmail access
Source IP Address: 206.133.210.27
Destination IP Address: MY.NET.200.90
Source Port: 2268 **Destination Port:** 80
TCP flags: (ack),(psh)
(payload in hex/ascii follows – only first line shown)

The exact rule that triggered this alert was copied from the management console (Demarc-1.05) and was in use by the Snort NIDS engine (version 1.9 from CVS).

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"WEB-CGI formmail
access";flow:to_server; flags:A+; uricontent:"/formmail"; nocase;
reference:bugtraq,1187; reference:cve,CVE-1999-0172; reference:arachnids,226;
classtype:web-application-activity; sid:884; rev:5;)
```

Signatures are automatically updated and come from :
http://www.snort.org/dl/signatures/snortrules-current.tar.gz

Probability The Source Address Was Spoofed

Zero. The session requires the three-way handshake, and the nature of the attack (below) requires some degree of interaction with the host.

Description Of Attack

The attacker is testing for the existence of the "formmail" CGI. An input validation vulnerability exists in the Matt Wright Formmail CGI script which permits an arbitrary address be used in the submission – effectively using the webserver as an open SMTP relay. This trick is widely used to send unsolicited commercial email (eg: SPAM). Scripts which automate the systematic testing and exploitation of groups of servers in order to deliver SPAM in this manner have been incorporated into several commercial bulk email products.

Some references to the vunerability :

http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0357
http://online.securityfocus.com/archive/75/250520
http://online.securityfocus.com/bid/2080

Attack Mechanism

```
GET /cgi-
bin/formmail.pl?recipient=AntiArabLeague@aol.com&subject=http://MY.SERVER/cgi-
bin/formmail.pl&body=JupZ&email=caf@aol.com HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 5.0; Windows 98; DigExt)
Host: MY.SERVER
Connection: Keep-Alive
```

The CGI script fails to validate that the destination address is internal as intended (this CGI would typically be used in a "comments" or a "contact us" section of a web page, and it prepares a email which would be typically sent from the webserver to some generic address like "webmaster"). The would-be SPAMmer just forges the required fields, and the webserver now acts as a convenient SMTP relay – usually with a valid reverse DNS entry, and not all that commonly used – meaning the resulting SPAM tends to avoid sites using RBL, etc.

There are numerous advisories on this particular exploit. I have listed a few of them below for reference:

CVE: CAN-2000-0574, CAN-2000-0573, CAN-2000-0917
Bugtraq: 1387, 1711
advICE: 2001322

Correlations

The attack source is a Sprint dialup account, and was likely a "throwaway" account created for the sole purpose of sending out SPAM. Drawing conclusions based on the IP address of the source are not possible.

Also on Aug. 12, 2001 there was another probe by the same IP address to the same network the attacker scanned before. The website the following information came from is: http://www.incidents.org/archives/intrusions/msg01425.html.

Evidence of Active Targeting

Clear – the attack contained actual SPAM the attacker was sending, not just a probe to see if it would work.

Severity :

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

(3+3) – (1+3) = 2

criticality : **(3)** : This is a webserver used for research, not production.
lethality : **(3)** : people will hate us for being the source of SPAM, but that won't kill us.
system countermeasures : **(1)** : obviously not very good since it succeeded.
network countermeasures : **(3)** : extensive IDS logging, but server is on DMZ and exposed.

Defensive Recommendations

Determine if "formmail" functionality is required for content presented by this server or simply part of a default installation of something.

If the "formmail" functionality is not required, delete all copies of the script or move them out of directories visible/executable to the webserver.

If the "formmail" functional is required, the short-term solution is to find and replace all versions of "formmail.pl" with a more secure version. The long-term solution is to find a better (and more secure) way to send email from a webform. There are several methods which are better than Mr. Wright's.

Multiple Choice Question

What's going on here?

```
-------------------------------------------------------------------------------
#(25 - 182) [2002-04-16 18:05:35]
IPv4: 206.133.210.27 -> MY.NET.200.90
      hlen=5 TOS=0 dlen=440 ID=64201 flags=0 offset=0 TTL=116 chksum=55921
TCP:  port=2268 -> dport: 80  flags=***AP*** seq=208899197
      ack=1247658862 off=5 res=0 win=5840 urp=0 chksum=28989
Payload:  length = 400

000 : 47 45 54 20 2F 63 67 69 2D 62 69 6E 2F 66 6F 72   GET /cgi-bin/for
010 : 6D 6D 61 69 6C 2E 70 6C 3F 72 65 63 69 70 69 65   mmail.pl?recipie
020 : 6E 74 3D 41 6E 74 69 41 72 61 62 4C 65 61 67 75   nt=AntiArabLeagu
030 : 65 40 61 6F 6C 2E 63 6F 6D 26 73 75 62 6A 65 63   e@aol.com&subjec
040 : 74 3D 68 74 74 70 3A 2F 2F 78 78 78 78 78 78 78   t=http:xxxxxxxxx
050 : 78 78 78 78 78 78 78 78 78 78 78 78 78 78 78 2F   xxxxxxxxxxxxxx/
060 : 63 67 69 2D 62 69 6E 2F 66 6F 72 6D 6D 61 69 6C   cgi-bin/formmail
070 : 2E 70 6C 26 62 6F 64 79 3D 4A 75 70 5A 26 65 6D   .pl&body=JupZ&em
080 : 61 69 6C 3D 63 61 66 40 61 6F 6C 2E 63 6F 6D 20   ail=caf@aol.com
```

a) An attacker is attempting a buffer overflow on the webserver at MY.NET.200.90
b) An attacker is attempting an exploit to view source code of CGI scripts
c) An attacker is attempting to deface and/or "tag" webpages on MY.NET.200.90
d) An attacker is attempting to exploit CGI scripts to relay mail traffic

Answer : **(D)** : A vulnerable version of the "formmail" CGI is being used as a SPAM relay

Detect 2 – FTP format string attempt

```
-----------------------------------------------------------------------------
#(10 - 536460) [2002-04-09 04:41:43]  EXPERIMENTAL FTP format string attempt
IPv4: 217.228.229.183 -> MY.NET.63.93
      hlen=5 TOS=0 dlen=55 ID=22929 flags=0 offset=0 TTL=117 chksum=17552
TCP:  port=3245 -> dport: 21  flags=***AP*** seq=525222188
      ack=25200814 off=5 res=0 win=32767 urp=0 chksum=50450
Payload:  length = 15

000 : 66 74 70 3A 2F 2F 25 61 3A 25 70 2F 2C 0D 0A      ftp://%a:%p/,..

Apr  9 04:41:57 217.228.229.183:3105 -> MY.NET.63.1:21 SYN ******S*
Apr  9 04:41:58 217.228.229.183:3141 -> MY.NET.63.82:21 SYN ******S*
Apr  9 04:41:59 217.228.229.183:3179 -> MY.NET.63.85:21 SYN ******S*
Apr  9 04:42:00 217.228.229.183:3212 -> MY.NET.63.86:21 SYN ******S*
Apr  9 04:42:01 217.228.229.183:3245 -> MY.NET.63.93:21 SYN ******S*
Apr  9 04:42:03 217.228.229.183:3296 -> MY.NET.63.94:21 SYN ******S*
Apr  9 04:42:04 217.228.229.183:3343 -> MY.NET.63.101:21 SYN ******S*
Apr  9 04:42:05 217.228.229.183:3377 -> MY.NET.63.110:21 SYN ******S*
Apr  9 04:42:06 217.228.229.183:3419 -> MY.NET.63.140:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:4651 -> MY.NET.63.8:21 SYN ******S*
Apr  9 11:01:01 217.228.229.183:4660 -> MY.NET.63.12:21 SYN ******S*
Apr  9 11:01:01 217.228.229.183:4682 -> MY.NET.63.19:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:4683 -> MY.NET.63.20:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:4707 -> MY.NET.63.26:21 SYN ******S*
Apr  9 11:01:01 217.228.229.183:4718 -> MY.NET.63.30:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:4831 -> MY.NET.63.63:21 SYN ******S*
Apr  9 11:01:02 217.228.229.183:4838 -> MY.NET.63.64:21 SYN ******S*
Apr  9 11:01:02 217.228.229.183:4840 -> MY.NET.63.66:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:4847 -> MY.NET.63.67:21 SYN ******S*
Apr  9 11:01:02 217.228.229.183:4848 -> MY.NET.63.68:21 SYN ******S*
Apr  9 11:01:02 217.228.229.183:4856 -> MY.NET.63.69:21 SYN ******S*
Apr  9 11:01:02 217.228.229.183:4864 -> MY.NET.63.72:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:4870 -> MY.NET.63.73:21 SYN ******S*
Apr  9 11:01:02 217.228.229.183:4871 -> MY.NET.63.74:21 SYN ******S*
Apr  9 11:01:02 217.228.229.183:4924 -> MY.NET.63.75:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:4961 -> MY.NET.63.82:21 SYN ******S*
Apr  9 11:01:02 217.228.229.183:4967 -> MY.NET.63.83:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:4970 -> MY.NET.63.84:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:4997 -> MY.NET.63.90:21 SYN ******S*
Apr  9 11:01:02 217.228.229.183:4999 -> MY.NET.63.91:21 SYN ******S*
Apr  9 11:01:02 217.228.229.183:3064 -> MY.NET.63.107:21 SYN ******S*
Apr  9 11:01:02 217.228.229.183:3072 -> MY.NET.63.110:21 SYN ******S*
Apr  9 11:01:02 217.228.229.183:3169 -> MY.NET.63.141:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:4978 -> MY.NET.63.86:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:4948 -> MY.NET.63.78:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:3065 -> MY.NET.63.108:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:3043 -> MY.NET.63.102:21 SYN ******S*
Apr  9 11:01:04 217.228.229.183:3164 -> MY.NET.63.140:21 SYN ******S*
Apr  9 11:01:10 217.228.229.183:4651 -> MY.NET.63.8:21 SYN ******S*
Apr  9 11:01:10 217.228.229.183:4707 -> MY.NET.63.26:21 SYN ******S*
Apr  9 11:01:10 217.228.229.183:4683 -> MY.NET.63.20:21 SYN ******S*
Apr  9 11:01:10 217.228.229.183:4847 -> MY.NET.63.67:21 SYN ******S*
Apr  9 11:01:10 217.228.229.183:4831 -> MY.NET.63.63:21 SYN ******S*
Apr  9 11:01:10 217.228.229.183:4961 -> MY.NET.63.82:21 SYN ******S*
Apr  9 11:01:10 217.228.229.183:4970 -> MY.NET.63.84:21 SYN ******S*
Apr  9 11:01:10 217.228.229.183:4870 -> MY.NET.63.73:21 SYN ******S*
Apr  9 11:01:10 217.228.229.183:4978 -> MY.NET.63.86:21 SYN ******S*
Apr  9 11:01:10 217.228.229.183:4948 -> MY.NET.63.78:21 SYN ******S*
Apr  9 11:01:10 217.228.229.183:3065 -> MY.NET.63.108:21 SYN ******S*
Apr  9 11:01:10 217.228.229.183:3043 -> MY.NET.63.102:21 SYN ******S*
Apr  9 11:01:10 217.228.229.183:3164 -> MY.NET.63.140:21 SYN ******S*
```

Identity of offending source

```
Server# whois 217.228.229.183@whois.ripe.net
[whois.ripe.net]
% This is the RIPE Whois server.
% The objects are in RPSL format.
% Please visit http://www.ripe.net/rpsl for more information.
% Rights restricted by copyright.
% See http://www.ripe.net/ripencc/pub-services/db/copyright.html

inetnum:      217.224.0.0 - 217.237.161.47
netname:      DTAG-DIAL15
descr:        Deutsche Telekom AG
country:      DE
admin-c:      DTIP-RIPE
tech-c:       ST5359-RIPE
status:       ASSIGNED PA
remarks:      ************************************************************
remarks:      * ABUSE CONTACT: abuse@t-ipnet.de IN CASE OF HACK ATTACKS, *
remarks:      * ILLEGAL ACTIVITY, VIOLATION, SCANS, PROBES, SPAM, ETC.   *
remarks:      ************************************************************
notify:       auftrag@nic.telekom.de
notify:       dbd@nic.dtag.de
mnt-by:       DTAG-NIC
changed:      auftrag@nic.telekom.de 20020108
source:       RIPE

person:       Security Team
address:      Deutsche Telekom AG
address:      Technikniederlassung Schwaebisch Hall
address:      D-89070 Ulm
address:      Germany
phone:        +49 731 100 84055
fax-no:       +49 731 100 84150
e-mail:       abuse@t-ipnet.de
nic-hdl:      ST5359-RIPE
notify:       auftrag@nic.telekom.de
notify:       dbd@nic.dtag.de
mnt-by:       DTAG-NIC
changed:      auftrag@nic.telekom.de 20010321
source:       RIPE
```

Although 217.228.229.183 has no reverse record, 217.228.229.1 (logically assumed to be part of the same netblock), is part of the dial-up pool for "t-dialin.net" – an ISP in Germany.

Source Of Trace

Author's Network (class B netblock via DS-3).

Detect Was Generated By

Snort NIDS (version 1.9)

Log Format :

```
--------------------------------------------------------------------------------
#(10 - 536460) [2002-04-09 04:41:43]  EXPERIMENTAL FTP format string attempt
IPv4: 217.228.229.183 -> MY.NET.157.11
      hlen=5 TOS=0 dlen=55 ID=22929 flags=0 offset=0 TTL=117 chksum=17552
TCP:  port=3245 -> dport: 21  flags=***AP*** seq=525222188
      ack=25200814 off=5 res=0 win=32767 urp=0 chksum=50450
Payload:  length = 15
```

```
000 : 66 74 70 3A 2F 2F 25 61 3A 25 70 2F 2C 0D 0A      ftp://%a:%p/,..
```

Time and Date: April 9th, 2002 @ 04:41:43 EST (GMT – 05:00)
Alert Name: EXPERIMENTAL FTP format string attempt
Source IP Address: 217.228.229.183
Destination IP Address: MY.NET.157.11
Source Port: 3245 **Destination Port:** 21
TCP flags: (ack),(psh)
(payload in hex/ascii follows – only first line shown)

The exact rule that triggered this alert was copied from the management console (Demarc-1.05) and was in use by the Snort NIDS engine (version 1.9 from CVS).

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"EXPERIMENTAL FTP format string
 attempt"; flags:A+; flow:to_server; content:"%p"; nocase; classtype:attempted-a
dmin; sid:1530; rev:2;)
```

Signatures are automatically updated and come from :
http://www.snort.org/dl/signatures/snortrules-current.tar.gz

Probability The Source Address Was Spoofed

Zero. The session requires the three-way handshake, and the nature of the attack (below) requires some degree of interaction with the host. Also, the massive network scan would have little effect if the results couldn't make it back to the sender.

Description Of Attack

Versions of wu-ftpd 2.6 and later (and any ftpd derived from wu-ftpd 2.0 or later), as well as systems running ftpd derived from BDS ftpd 5.51 or 5.60 are vunerable to attacks where a malicious user can pass character format strings consisting of carefully constructed *printf() conversion characters (%f, %p, %n, etc) while executing a "site exec" command, the ftp daemon may be tricked into executing arbitrary code as root.[1] [2] [3]

The SANS reading room contains an excellent write-up on this (and another) type of root compromise against wu-ftpd. It is available at http://rr.sans.org/threats/wu-ftp.php

Some references to the vulnerability :

ftp://ftp.auscert.org.au/pub/auscert/advisort/AA-2000.02
http://www.securitfocus.com/vdb/bottom.html?secrion=discussion&vid=1387
http://www.securitfocus.com/vdb/bottom.html?secrion=discussion&vid=1438
http://ciac.llnl.gov/ciac/bulletins/k-054.shtml

[1] http://packetstormsecurity.org/advisories/cert/CA-2000-13.ftpd
[2] http://rhn.redhat.com/errata/RHSA-2001-053.html
[3] http://packetstormsecurity.org/advisories/freebsd/FreeBSD-SA-00:35.proftpd

Attack Mechanism

An excellent demonstration of this in action was posted to BugTRAQ by "zargon"[4]:

```
$ ftp localhost
Connected to localhost.
220 localhost FTP server (Version 1.1.214.6 Wed Feb  9 08:03:34 GMT 2000) ready.
Name (localhost:zorgon):zorgon
331 Password required for zorgon.
Password:
230 User zorgon logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> site exec %p %p %p %p
200-40008f10 00000003 00000002 00000001
200  (end of '40008f10 00000003 00000002 00000001')
ftp> site exec %n %n %n %n
Bus error(coredump)
```

Due to improper implementation of the "site exec" command in wu-ftpd, remote users (including those authenticated as "anonymous") can potentially execute code on the server with root privildges.

Correlations

The attack source is a dialup account, therefore drawing conclusions based on the IP address of the source is difficult. 217.228.229.183 triggered a number of other alerts on this particular day on MY.NET.63.0/24 and on two other discontinuous segments which I monitor, MY.NET.200.0/24 and MY.NET.139.0/24.

Searches at www.incidents.org for other malicious activity from this and other netblocks belonging to the "t-dialin.net" branch of Deutch Telecom AG indicate a FTP scans frequently originate from them, although such could be said of most any ISP.

http://www.incidents.org/archives/intrusions/msg04128.html
http://www.incidents.org/archives/intrusions/msg01899.html
http://www.incidents.org/archives/intrusions/msg03529.html

Evidence of Active Targeting

Clear -- the attacker ran a comprehensive scan against MY.NET.163.0/24 and then returned some 5 hours later to attempt the exploit captured in the logs.

Severity

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

(5+5) – (5+2) = 3

[4] http://online.securityfocus.com/archive/1/155006

criticality : **(5)** : among other things, this is our main DNS server
lethality : **(5)** : successful compromise can give root access
system countermeasures : **(5)** : server does not permit use of any "SITE" command
network countermeasures : **(2)** : extensive IDS logging, but server is on DMZ and exposed

Defensive Recommendation

As general best-practice, check with http://www.wu-ftpd.org/ (or your vendor) for the latest security updates for any daemons you run.

There are no specific recommendations that relate to this incident.

Multiple Choice Question

What's going on here?

```
---------------------------------------------------------------------------
Apr  9 04:41:57 217.228.229.183:3105 -> MY.NET.63.1:21 SYN ******S*
Apr  9 04:41:58 217.228.229.183:3141 -> MY.NET.63.82:21 SYN ******S*
Apr  9 04:41:59 217.228.229.183:3179 -> MY.NET.63.85:21 SYN ******S*
Apr  9 04:42:00 217.228.229.183:3212 -> MY.NET.63.86:21 SYN ******S*
Apr  9 04:42:01 217.228.229.183:3245 -> MY.NET.63.93:21 SYN ******S*
Apr  9 04:42:03 217.228.229.183:3296 -> MY.NET.63.94:21 SYN ******S*
---------------------------------------------------------------------------
#(10 - 536460) [2002-04-09 04:41:43]
IPv4: 217.228.229.183 -> MY.NET.63.93
      hlen=5 TOS=0 dlen=55 ID=22929 flags=0 offset=0 TTL=117 chksum=17552
TCP:  port=3245 -> dport: 21  flags=***AP*** seq=525222188
      ack=25200814 off=5 res=0 win=32767 urp=0 chksum=50450
Payload:  length = 15

000 : 66 74 70 3A 2F 2F 25 61 3A 25 70 2F 2C 0D 0A      ftp://%a:%p/,..
---------------------------------------------------------------------------
```

e) Someone was looking for an anonymous FTP server and found one on MY.NET
f) Someone was looking for an exploitable FTP server and found one on MY.NET
g) Someone was testing servers on MY.NET for vulnerable FTP servers
h) Servers on MY.NET have been compromised by 217.228.229.183

Answer : **(C)** : absent any additional code, the %p is a probe for the "format string" vulnerability

Detect 3 – MS-SQL 'sa' login attempts

```
-------------------------------------------------------------------------------
#(15 - 7942) [2002-05-28 04:29:26]  MS-SQL sa login failed
IPv4: 203.154.131.184 -> MY.NET.63.8
      hlen=5 TOS=0 dlen=99 ID=65180 flags=0 offset=0 TTL=127 chksum=13259
TCP:  port=1433 -> dport: 4878  flags=***AP*** seq=172373388
      ack=1281352040 off=5 res=0 win=7697 urp=0 chksum=44908
Payload:  length = 59

000 : 04 01 00 3B 00 00 00 00 AA 27 00 18 48 00 00 01   ...;.....'..H...
010 : 0E 1B 00 4C 6F 67 69 6E 20 66 61 69 6C 65 64 20   ...Login failed
020 : 66 6F 72 20 75 73 65 72 20 27 73 61 27 2E 00 00   for user 'sa'...
030 : 00 00 FD 02 00 00 00 00 00 00 00                  ...........


May 28 04:29:19 203.154.131.184:4871 -> MY.NET.63.1:1433 SYN ******S*
May 28 04:29:20 203.154.131.184:4878 -> MY.NET.63.8:1433 SYN ******S*
May 28 04:29:20 203.154.131.184:4882 -> MY.NET.63.12:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4889 -> MY.NET.63.19:1433 SYN ******S*
May 28 04:29:22 203.154.131.184:4890 -> MY.NET.63.20:1433 SYN ******S*
May 28 04:29:22 203.154.131.184:4896 -> MY.NET.63.26:1433 SYN ******S*
May 28 04:29:22 203.154.131.184:4900 -> MY.NET.63.30:1433 SYN ******S*
May 28 04:29:20 203.154.131.184:4929 -> MY.NET.63.59:1433 SYN ******S*
May 28 04:29:20 203.154.131.184:4932 -> MY.NET.63.62:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4937 -> MY.NET.63.67:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4934 -> MY.NET.63.64:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4936 -> MY.NET.63.66:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4933 -> MY.NET.63.63:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4938 -> MY.NET.63.68:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4939 -> MY.NET.63.69:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4941 -> MY.NET.63.71:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4940 -> MY.NET.63.70:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4942 -> MY.NET.63.72:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4943 -> MY.NET.63.73:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4945 -> MY.NET.63.75:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4944 -> MY.NET.63.74:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4948 -> MY.NET.63.78:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4952 -> MY.NET.63.82:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4954 -> MY.NET.63.84:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4955 -> MY.NET.63.85:1433 SYN ******S*
```

Identity of offending source

```
Server# host 203.154.131.184
184.131.154.203.in-addr.arpa. domain name pointer TruPPPS-NE6054.inet.co.th.


Server# whois 203.154.131.184@whois.apnic.net
[whois.apnic.net]

% Rights restricted by copyright. See http://www.apnic.net/db/dbcopyright.html
% (whois7.apnic.net)

inetnum:     203.154.0.0 - 203.154.255.255
netname:     INET-TH
descr:       Internet Thailand Company Limited
country:     TH
admin-c:     BS16-AP
tech-c:      CN2-TH
tech-c:      SK13-AP
tech-c:      SK26-TH
mnt-by:      APNIC-HM
mnt-lower:   MAINT-TH-INET
changed:     hostmaster@apnic.net 20010124
source:      APNIC

person:      Buncha Srisamanuwat
```

```
address:     Internet Thailand Company Limited
address:     108 Bangkok Thai Tower, 12th Floor,
address:     Rangnam Road, Rajdhevee,
address:     Bangkok 10400
country:     TH
phone:       +66-2-640-0345
fax-no:      +66-2-640-0456
e-mail:      athicha@inet.co.th
nic-hdl:     BS16-AP
mnt-by:      MAINT-TH-INET
changed:     snakk@inet.co.th 20010118
source:      APNIC
```

Source Of Trace

Author's Network (class B netblock via DS-3).

Detect Was Generated By

Snort NIDS (version 1.9)

Log Format :

```
-----------------------------------------------------------------------------
#(15 - 7942) [2002-05-28 04:29:26]  MS-SQL sa login failed
IPv4: 203.154.131.184 -> MY.NET.63.8
      hlen=5 TOS=0 dlen=99 ID=65180 flags=0 offset=0 TTL=127 chksum=13259
TCP:  port=1433 -> dport: 4878  flags=***AP*** seq=172373388
      ack=1281352040 off=5 res=0 win=7697 urp=0 chksum=44908
Payload:  length = 59

000 : 04 01 00 3B 00 00 00 00 AA 27 00 18 48 00 00 01   ...;.....'..H...
```

Time and Date: May 28th, 2002 @ 04:29:26 EST (GMT – 05:00)
Alert Name: MS-SQL sa login failed
Source IP Address: 203.154.131.184
Destination IP Address: MY.NET.63.8
Source Port: 1433 **Destination Port:** 4878
TCP flags: (ack),(psh)
(payload in hex/ascii follows – only first line shown)

The exact rule that triggered this alert was copied from the management console (Demarc-1.05) and was in use by the Snort NIDS engine (version 1.9 from CVS).

```
alert tcp $SQL_SERVERS 1433 -> $EXTERNAL_NET any (msg:"MS-SQL sa login failed";
content: "Login failed for user |27|sa|27|"; flags:A+; classtype:unsuccessful-
user; sid:688; rev:3;)
```

Signatures are automatically updated and come from :
http://www.snort.org/dl/signatures/snortrules-current.tar.gz

Probability The Source Address Was Spoofed

Possible, but unlikely. The traffic observed is a result of a worm which causes servers to scan the internet for other hosts to infect. The worm does not cause the compromised server to craft packets in such a way as to conceal the host's address, so we can be reasonably certain that 203.154.131.184 is the actual source.

Description Of Attack

Running any server or service with default or null passwords is a grave (but common) security mistake[5]. A recently released worm dubbed "SQL Snake" attempts to exploit this mistake to deliver trojan code to a SQL server, causing it to search the internet for other SQL servers to infect. This is very similar to the Nimda worm, except for SQL servers instead of IIS servers.

Some references to the vulnerability :

http://www.cert.org/incident_notes/IN-2002-04.html
http://support.microsoft.com/default.aspx?scid=kb;EN-US;Q313418
http://www.incidents.org/diary/diary.php?id=156
http://www.incidents.org/diary/diary.php?id=157
http://vil.nai.com/vil/content/v_99499.htm

Attack Mechanism

Once connected, the worm attempts to use the xp_cmdshell utility to enable and set a password for the guest user. If successful, the worm will :

1. assigns the guest user to the local Administrator and Domain Admins groups
2. copies itself to the victim system
3. disables the guest account
4. sets the *sa* password to the same password as the guest account
5. executes the copy on the victim system

Once the local copy is executing on the victim system, the worm begins scanning for other systems to infect. It also attempts to send a copy of the local password (SAM) database, network configuration information, and other SQL server configuration information to a fixed email address (ixtld@postone.com) via email.[6]

An excellent analysis of this worm is available at
http://www.incidents.org/diary/diary.php?id=156

[5] http://www.kb.cert.org/vuls/id/635463
[6] http://www.cert.org/incident_notes/IN-2002-04.html

Correlations

203.154.131.184 triggered the same alert on other hosts within MY.NET.63.0/24 and on two other discontinuous segments which I monitor, MY.NET.200.0/24 and MY.NET.139.0/24.

TCP/1433 is the top port on the Incidents.org "Internet Storm Center".[7], and based on statistical data from my own segments, is a close contender with Nimda/CodeRed noise on TCP/80 as to number of probes.

Evidence of Active Targeting

None. This is worm activity which systematically scans in numerical order for other servers to infect.

Severity :

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

(4+3) – (5+1) = 1

criticality : **(4)** : host is a production SQL server, but only for extranet applications
lethality : **(3)** : compromise is relatively easy to recover from
system countermeasures : **(5)** : server does not use a blank "sa" password
network countermeasures : **(1)** : we discovered that plug-gw allowed proxied connections

Defensive Recommendation

Ensure all SQL servers are firewalled from the internet.

Ensure all security related patches are applied (http://www.microsoft.com/security)

Follow "best practice" on default and admin accounts, using strong passwords. A good guide on SQL server security is available from the SANS reading room at : http://rr.sans.org/win/SQL_sec.php

[7] http://isc.incidents.org/top10.html

Multiple Choice Question

What's going on here?

```
May 28 04:29:19 203.154.131.184:4871 -> MY.NET.63.1:1433 SYN ******S*
May 28 04:29:20 203.154.131.184:4878 -> MY.NET.63.8:1433 SYN ******S*
May 28 04:29:20 203.154.131.184:4882 -> MY.NET.63.12:1433 SYN ******S*
May 28 04:29:21 203.154.131.184:4889 -> MY.NET.63.19:1433 SYN ******S*
May 28 04:29:22 203.154.131.184:4890 -> MY.NET.63.20:1433 SYN ******S*
May 28 04:29:22 203.154.131.184:4896 -> MY.NET.63.26:1433 SYN ******S*
May 28 04:29:22 203.154.131.184:4900 -> MY.NET.63.30:1433 SYN ******S*
May 28 04:29:20 203.154.131.184:4929 -> MY.NET.63.59:1433 SYN ******S*

-----------------------------------------------------------------------------
#(15 - 7942) [2002-05-28 04:29:26]  MS-SQL sa login failed
IPv4: 203.154.131.184 -> MY.NET.63.8
      hlen=5 TOS=0 dlen=99 ID=65180 flags=0 offset=0 TTL=127 chksum=13259
TCP:  port=1433 -> dport: 4878  flags=***AP*** seq=172373388
      ack=1281352040 off=5 res=0 win=7697 urp=0 chksum=44908
Payload:  length = 59

000 : 04 01 00 3B 00 00 00 00 AA 27 00 18 48 00 00 01   ...;.....'..H...
010 : 0E 1B 00 4C 6F 67 69 6E 20 66 61 69 6C 65 64 20   ...Login failed
020 : 66 6F 72 20 75 73 65 72 20 27 73 61 27 2E 00 00   for user 'sa'...
030 : 00 00 FD 02 00 00 00 00 00 00 00                  ...........
```

a) Someone is looking for exploitable SQL servers and found one on MY.NET
b) A misconfigured web server cannot connect to the database server at MY.NET.63.8
c) Automated methods to test SQL servers on MY.NET are underway and not succeeding
d) MY.NET contains hosts infected with the "SQLsnake" worm.

Answer : **(C)** : the host is being tested by a remote host, but was not vulnerable to *this* attack

Assignment 3 – Analyze This!

List of Files Used For Dataset

Filename	MD5 Checksum
alert.020401.gz	2d02c20466887012e92a8ccd101ac590
alert.020402.gz	a9bcc5e15836b24560f93adfb2b34212
alert.020403.gz	0484a7cef6f52d39273ff3821d838eb2
alert.020404.gz	f4bc71d66c923282a1ff0ffc7a9d2d36
alert.020405.gz	efecde791fc47926bc6282d480e09e27
oos_Apr.1.2002.gz	ddfed6dfcca2411d6a157919434b2435
oos_Apr.2.2002.gz	a4f0691d6f4ae7480886849c7358b323
oos_Apr.3.2002.gz	d9657cfccb74caab80f23b83d34c90d2
oos_Apr.4.2002.gz	96c4b3fed84a44a84117594aac747326
oos_Apr.5.2002.gz	18bb032562d49708362b79a3ab0e3c18
scans.020401.gz	3ab4be4eb741a202acaac99103d63963
scans.020402.gz	34682fe89cf9877c729444d3d10dcdf0
scans.020403.gz	c94c46774546d008b1c90fd5a31bd1a0
scans.020404.gz	70b7e64762d559548bf119045c33bf80
scans.020405.gz	a8d1c7f0374f949c10454dcfc7165bd4

Analysis Process

There is *no way* I can analyze all this data by hand -- this is exactly what ACID was designed to do – but the problem becomes *how do I get all this data into mySQL in the snortDB format*?

I prepared a test server (neptune.MY.NET) with the following configuration :

- Dell PowerEdge 2550 : Dual Pentium III, 1Ghz, 2Gb RAM
- FreeBSD 4.5 SMP
- Apache 1.3.24 (with mod_ssl and mod_perl)
- ACID 0.9.6b21
- Perl 5.005 (with DBD::MySQL)
- PHP 4.2.0
- GD 1.8.4 (with JPEG and PNG libraries)

…and a second test server (pluto.MY.NET) with the following configuration :

- Dell PowerEdge 2550 : Dual Pentium III, 1Ghz, 2Gb RAM
- FreeBSD 4.5 SMP
- MySQL 3.23.48

Format of logs :

Fortunately, the format of the logfiles is the Snort FAST format (eg: ./snort –a fast –c snort.conf). Each row of data is formatted as follows :

[timestamp] [alert] [IP source and destination (if any)]

These three fields are delimited by a pair of asterisks enclosed in brackets '[**]'

The IP information is delimited between source and destination by an arrow '->'
The IP address is delimited from the port number by a colon ':'

Perl can make quick work of fields like this using the SPLIT command. See [Appendix A] for the code I used to do just that. From the original alert line, we obtain the following individual variables :

$month	numerical month
$day	numerical day
$hour	numerical hours (24hr clock)
$minute	numerical minutes
$second	numerical seconds
$millisecond	numerical miliseconds
$alert	text alert (snort signature triggered)
$srcIP	dotted quad IP of source
$srcPT	numerical TCP/UDP source port (if any)
$dstIP	dotted quad IP destination (if any)
$dstPT	numerical TCP/UDP destination port (if any)

In the event of a TCP or UDP signature based alert, all 11 fields are present. The ALERT files also contain all activity from the portscan preprocessor, including both the initial alert, the status updates, and the final totals. We need to combine the alert.(date) files into two separate files – one with ALERT data, the other with portscan alerts.

```
zgrep –v alert.020401.gz spp_portscan >alert.020401.tcp [repeated for each file]
cat alert.02040*.tcp >>alert.tcp [final file used for ALERT analysis]
zcat scans.02040*.gz >>alert.scans [final file used for SCANS analysis]
```

When importing a portscan file, the fields are logged into the database as far as the srcIP – the remaining three fields are imported as NULL values. This conveniently allows us to use the statistics built into ACID which determine the frequency of any given alert, and allows analysis based on the source IP of the problem, whatever the alert type may be. The script echos to STDOUT the details of each line as the import is performed, including the CID (sequence number) and timestamp (used as a progress gauge on the file).

Getting the data into MySQL in a format compliant with the Snort database format involves a few special considerations :

1. The signatures are keyed by 'sig_id' onto the events. The script must determine if the signature already exists in the database, and use that sig_id value for all future signatures of the same name. If the signature is new, we must increment the sig_id value by 1 and use that value both now and for all future events of the same name
2. The events are keyed by CID. The CID is a unique serial value for all events recorded by a sensor. This essentially starts at 1 and increases by 1 as each line is imported.
3. Since all events will be imported using a single sensor ID, we need to manually put this data in the database (only once) by using the following SQL commands :

```
use giac;
insert into sensor (sid, hostname, interface, detail, encoding) values ('1',
'giac', '[file]', '1', '0');
```

4. IP data is obfuscated in the logs. The mathematical operations performed in the logs to convert the IP to decimal treats non-numeric values as zero – which conveniently makes ACID treat them as zeros – meaning MY.NET.153.251 becomes 0.0.153.251
5. There are a number of fields that are not present in the logs, but are required to be non-null in order for ACID to operate properly (eg: all the TCP option data). We will fill these fields with standard values as if each packet was a "normal" TCP packet because it's the statistics and relationships we are concerned with, not the details (since details weren't provided in the logs).

The resulting file [alert.scans] was then imported into the database using the Perl script from [Appendix A] by invoking :

```
perl import.pl
```

That took about an hour to import the 1,049,957 events into the database. ACID was loaded to continue with the analysis, and forms the basis for the attached report.

Scan data was handled in a similar manner with a slightly modified version of the script in [Appendix A]. That import took about 3 hours for 3,523,821 alerts and forms the basis for the scan data in the attached report.

Security Assessment of Incidents.org University

TO : Chief Information Officer
RE : Security Analysis : April 1st – 5th, 2002

I have received your IDS logs for analysis and have prepared a summary report including recommendations based on the information contained therein. Since checksums were not provided, please verify the following information is correct from the logs submitted :

Filename	MD5 Checksum
alert.020401.gz	2d02c20466887012e92a8ccd101ac590
alert.020402.gz	a9bcc5e15836b24560f93adfb2b34212
alert.020403.gz	0484a7cef6f52d39273ff3821d838eb2
alert.020404.gz	f4bc71d66c923282a1ff0ffc7a9d2d36
alert.020405.gz	efecde791fc47926bc6282d480e09e27
oos_Apr.1.2002.gz	ddfed6dfcca2411d6a157919434b2435
oos_Apr.2.2002.gz	a4f0691d6f4ae7480886849c7358b323
oos_Apr.3.2002.gz	d9657cfccb74caab80f23b83d34c90d2
oos_Apr.4.2002.gz	96c4b3fed84a44a84117594aac747326
oos_Apr.5.2002.gz	18bb032562d49708362b79a3ab0e3c18
scans.020401.gz	3ab4be4eb741a202acaac99103d63963
scans.020402.gz	34682fe89cf9877c729444d3d10dcdf0
scans.020403.gz	c94c46774546d008b1c90fd5a31bd1a0
scans.020404.gz	70b7e64762d559548bf119045c33bf80
scans.020405.gz	a8d1c7f0374f949c10454dcfc7165bd4

This is a summary analysis only. Please consider the following :

The data provided contains only minimal detail which is obfuscated to conceal your internal network. Generally this practice is wise to avoid unintended disclosure; however it's presence in the provided logs complicates the analysis and may result in some inconclusive findings.

I have not received a copy of the existing security policy against which to perform the audit, so it will be assumed that none exists and recommendations will be made accordingly.

I have not received network diagrams indicating sensor placement, nor a copy of the ruleset in place during the period represented in the logs. I have drawn reference rulesets from the two most popular sources, www.whitehats.com and www.snort.org for comparison where appropriate.

Sincerely,

/s/ Michael Holstein

Executive Summary

Alert, Scan, and OOS (statistically out-of-spec) traffic alerts from the period beginning 04/01/02-00:00:00 and ending 04/05/02-23:59:59 were submitted from what appears to be a single instance of Snort observing the perimeter Internet gateway of a class B network.

1,049,957 individual alerts were generated during the period of analysis.
- 82 unique signatures were triggered
- 10,125 unique source addresses (both internal and external) were identified
- 2,355 unique destination addresses (both internal and external) were identified

General Recommendations :

- Remove common security exposures. Ensure default passwords and community strings are not used and that management of devices is restricted by ACL to internal hosts only.

- Implement RFC-1918 networks and a firewall which performs statefull inspection and network address translation (NAT).

- Rearrange network topology using a tiered architecture to provide concentric zones of protection, placing servers requiring public access in separate demilitarized zones (DMZ). Use detailed ACLs to permit DMZ hosts to communicate with internal hosts when required, permitting only those ports and protocols required. Implement switched VLANs internally to reduce congestion and provide additional security.

- Address use of insecure (FTP, Telnet, etc) and bandwidth abusive (Kazaa, Napster, etc) protocols with a security policy, and enforce through ACLs on perimeter devices.

- Implement additional, more powerful NIDS sensors, and update software to the current stable version. Implement a facility or procedure to ensure rulesets are regularly updated.

- Conduct a comprehensive on-site host and network security audit. Ensure the auditor is provided a detailed network diagram, copies of router ACLs, NIDS placement and copies of the ruleset, and a list of designated public servers and the services they host.

Table 1 : Alert Data : "Alerts by Frequency"

ALERT NAME	#alerts	SrcIP	DstIP
connect to 515 from inside	636038	163	5
SNMP public access	92595	25	154
spp_http_decode: IIS Unicode attack detected	86587	182	1017
SMB Name Wildcard	66946	300	315
spp_http_decode: CGI Null Byte attack detected	44305	34	41
ICMP Echo Request L3retriever Ping	33491	164	15
INFO MSN IM Chat data	22006	119	118
MISC Large UDP Packet	16799	21	13
High port 65535 udp - possible Red Worm	14653	222	178
INFO Inbound GNUTella Connect request	11680	8952	13
ICMP Echo Request Nmap or HPING2	5664	62	303
Watchlist 000220 IL-ISDNNET-990517	4840	19	15
FTP DoS ftpd globbing	4048	31	16
ICMP Fragment Reassembly Time Exceeded	2228	69	88
ICMP Router Selection	1490	137	1
WEB-IIS view source via translate header	1317	57	2
NMAP TCP ping!	841	18	325
WEB-MISC Attempt to execute cmd	723	28	34
INFO Outbound GNUTella Connect request	546	13	440
WEB-IIS _vti_inf access	322	110	1
Watchlist 000222 NET-NCFC	320	4	4
ICMP Echo Request Windows	301	32	26
WEB-FRONTPAGE _vti_rpc access	299	108	1
Null scan!	271	26	12
WEB-CGI scriptalias access	158	7	2
Possible trojan server activity	138	18	18
SCAN Proxy attempt	137	22	13
INFO napster login	122	1	29
ICMP Destination Unreachable (Prohibited)	103	1	1
ICMP traceroute	91	33	6
INFO Napster Client Data	91	20	71
INFO Possible IRC Access	89	24	21
WEB-CGI ksh access	74	1	1
ICMP Echo Request BSDtype	60	3	4
MISC traceroute	47	3	2
INFO - Possible Squid Scan	46	10	11
INFO FTP anonymous FTP	44	5	15
Queso fingerprint	40	10	9
Attempted Sun RPC high port access	30	6	19
EXPLOIT x86 NOOP	28	12	15
ICMP Destination Unreachable (Protocol)	28	4	4

WEB-MISC compaq nsight directory traversal	25	10	10
EXPLOIT NTPDX buffer overflow	25	9	6
SCAN Synscan Portscan ID 19104	24	24	9
Back Orifice	23	4	19
INFO napster upload request	22	3	1
MYPARTY - Possible My Party infection	22	3	1
SUNRPC highport access!	20	2	1
WEB-MISC 403 Forbidden	18	2	10
High port 65535 tcp - possible Red Worm	15	2	2
EXPLOIT x86 setuid 0	14	12	7
EXPLOIT x86 stealth noop	11	2	9
Port 55850 tcp - Possible myserver activity	11	6	6
Port 55850 udp - Possible myserver activity	8	6	7
RPC tcp traffic contains bin_sh	8	3	4
WEB-MISC http directory traversal	7	4	2
IDS552/web-iis_IIS ISAPI Overflow ida nosize	7	7	6
EXPLOIT x86 setgid 0	6	6	6
SCAN FIN	5	3	3
Incomplete Packet Fragments Discarded	5	5	3
WEB-IIS encoding access	4	3	2
RFB - Possible WinVNC - 010708-1	4	3	3
TFTP - Ext UDP conn to Int. tftp server	4	3	2
INFO Outbound GNUTella Connect accept	3	3	1
MISC PCAnywhere Startup	3	1	1
MISC source port 53 to <1024	2	2	2
WEB-MISC webdav search access	2	2	1
Probable NMAP fingerprint attempt	2	1	2
WEB-IIS asp-dot attempt	2	2	1
WEB-MISC whisker head	2	1	1
suspicious host traffic	2	2	1
WEB-CGI formmail access	2	2	2
TELNET access	2	1	2
TFTP - Int. UDP conn to Ext. tftp server	2	2	2
MISC Invalid PCAnywhere Login	2	1	1
WEB-MISC ICQ Webfront HTTP DOS	1	1	1
WEB-CGI redirect access	1	1	1
INFO Inbound GNUTella Connect accept	1	1	1
IDS475/web-iis_web-webdav-propfind	1	1	1
ICMP Router Selection (Undefined Code!)	1	1	1
x86 NOOP - unicode BUFFER OVERFLOW ATTACK	1	1	1
TCP SMTP Source Port traffic	1	1	1

Table 2 : Alert Data : "Top Talkers" – Source IP and Port

IP Address				IP Ports (TCP/UDP/ICMP)				
IP Source	alerts	sig	DstIP	Port	alerts	sig	SrcIP	DstIP
MY.NET.150.83	299723	3	5	137	66944	2	300	315
MY.NET.153.164	76134	6	31	512	62618	4	4	4
MY.NET.153.118	57453	6	32	0	44003	15	402	435
MY.NET.153.126	28181	5	19	2278	18663	3	11	10
MY.NET.153.119	18217	4	93	2280	16718	2	12	12
MY.NET.153.197	16880	6	32	2478	15757	3	6	5
MY.NET.11.6	15052	1	58	2276	15439	3	11	9
MY.NET.70.177	12354	2	33	1037	14235	2	4	3
MY.NET.153.113	11893	4	100	65535	11338	2	193	174
MY.NET.11.7	11501	2	59	1863	11268	7	71	73
MY.NET.153.193	10018	6	35	1041	10433	3	8	7
MY.NET.153.211	9813	5	132	1198	9881	6	27	23
MY.NET.88.203	9785	4	5	515	9043	1	2	2
MY.NET.88.207	9677	4	4	1100	9004	5	11	36
MY.NET.88.181	9665	4	6	1043	8238	5	7	6

Table 3 : Alert Data : "Top Talkers" – Destination IP and Port

IP Address				IP Ports (TCP/UDP/ICMP)				
IP Destination	alerts	sig	SrcIP	Port	alerts	sig	SrcIP	DstIP
MY.NET.150.198	331789	4	159	515	636038	1	163	5
MY.NET.151.77	299771	5	6	80	132805	26	384	1029
MY.NET.150.195	65778	6	28	161	92596	2	26	155
MY.NET.11.6	32994	3	59	137	66947	2	301	315
209.10.239.135	26730	1	7	0	44027	15	403	438
MY.NET.11.7	25442	3	59	65535	12953	2	206	172
MY.NET.11.5	11291	2	59	6346	12062	6	8877	397
211.115.213.202	8607	1	18	1863	10847	1	62	59
MY.NET.153.171	8079	9	24	21	4092	2	36	31
152.163.210.75	6563	2	3	4662	1556	8	10	1
MY.NET.152.109	5441	1	4	1647	1434	2	3	2
MY.NET.5.96	4955	26	234	2109	1190	1	1	1
MY.NET.150.83	4627	9	15	8080	1120	2	60	27
MY.NET.153.174	4555	6	43	2407	1106	1	1	1
MY.NET.153.143	4470	8	3350	1709	1038	1	1	1

Discussion of Alerts : Analysis, Severity and Recommendations

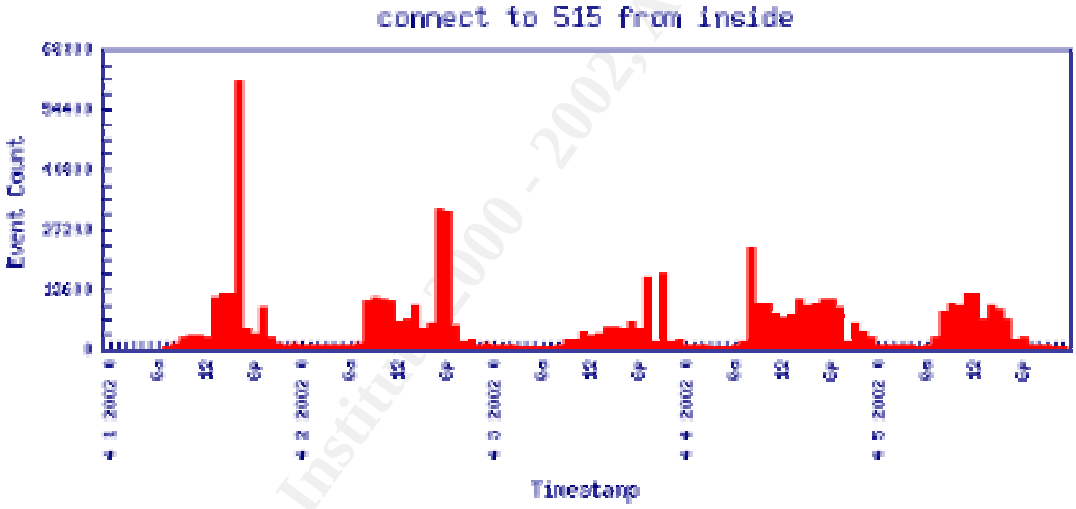
Alerts are discussed in the order in which they appear on **Table 1**.

Connect to 515 from Inside :

TCP/515 is the IANA registered port for the LPR (printer) service.[8]

This is the most frequently occurring alert, with over 630,000 events detected in the 5 day period examined (61%). The rule was likely configured to alert to anomalous activity as internal hosts attempted to spread various Unix worms to outside hosts by way of the LPD port.

The time-distribution of these alerts suggests a normal pattern of activity which follows the business day, and the large spike around 4pm on April 1 was most likely heavy printing activity.



There are only 5 destination IP addresses, all within MY.NET. The bulk of the alerts is divided between MY.NET.150.198 and MY.NET.151.77, with a much smaller amount to MY.NET.150.83. Those servers also have significant amounts of SNMP alerts for "public" access. It is therefore assumed that MY.NET.150 and MY.NET.151.77 are high-volume print servers.

If the concern is to alert on connection attempts from inside (as the alert would indicate), modify the values of $EXTERNAL_NET in "snort.conf" to exclude all local addresses.

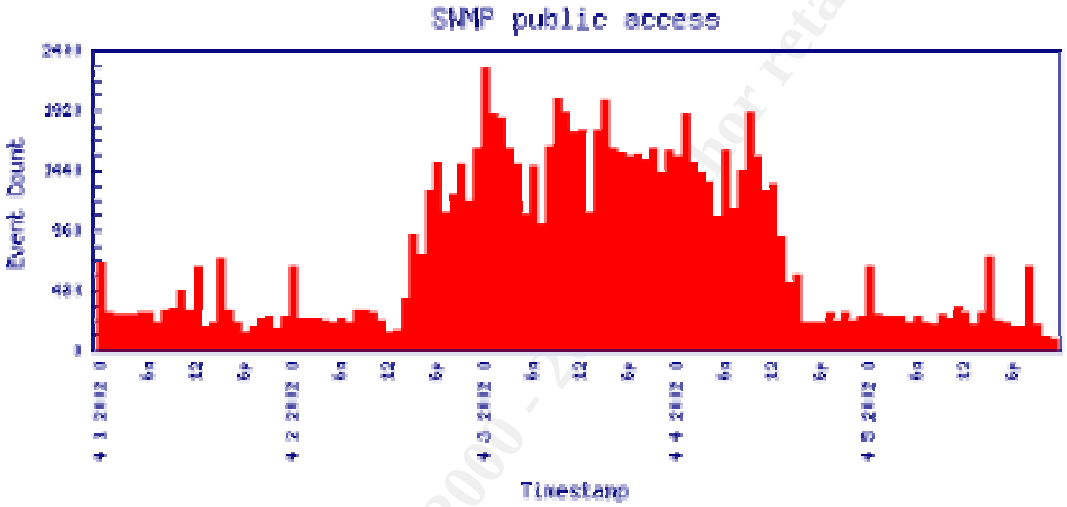
```
var EXTERNAL_NET [!MY.NET.0.0/16]
```

[8] http://www.iana.org/assignments/port-numbers

SNMP Public Access :

The Simple Network Management Protocol (SNMP) is widely used to gather statistics and for hardware management. A "community string" is used for authentication in versions 1 and 2 of the protocol, and by default is "public" for RO access and "private" for RW.

The "background level" for these alerts is around 300 per hour, and a huge spike was seen beginning at 3pm on 4/2/02, peaking at Midnight 4/3/02, and ending 3pm on 4/4/02. This coincides with reduced printer activity on the same day, and begins to taper off as a spike in printer activity occurs. This may be related to some unidentified malfunction.



25 unique source addresses triggered this alert, all from within MY.NET.0.0/16. The majority were to a single destination of MY.NET.150.198 which was previously identified as a print server and/or network management station.

Use of default community strings and/or passwords is a common security mistake.[9]

Block SNMP in both directions at the perimeter and disable SNMP on all hosts where it isn't absolutely necessary. On hosts requiring RO SNMP, enable only RO access, and change the community string to a complex password. On hosts requiring RW SNMP, use complex passwords and access-lists or tcpwrappers to restrict source IP addresses.

Spp_http_decode : IIS Unicode attack detected

Microsoft IIS servers are vulnerable to directory traversal attack by using Unicode characters in the URL passed to the webserver. This permits an attacker to run any command of choice on the target using "system" permissions.[10] [11]

[9] http://nsa2.www.conxion.com/support/guides/sd-7.pdf

The first request below uses ASCII notation to do the directory traversal, which is not permitted by the webserver. The second request uses the '%5c" Unicode encoding to accomplish the same goal, and unless the MS00-57 patch is applies, will permit the execution of the command interpreter.

```
GET /scripts/..\../winnt/system32/cmd.exe?/c+dir
GET /scripts/..%5c../winnt/system32/cmd.exe?/c+dir
```

These alerts are part of the background noise on the internet from Nimda[12] variants, and it is quite common to see large amounts of this alert with webservers as the target. Any alert with an internal address as the SOURCE (and a large number of external destinations) indicates a possible infection of an internal webserver and should be investigated immediately.

The following hosts had more than 3 destinations for this alert and should be audited :

MY.NET.153.110	MY.NET.153.121	MY.NET.150.165	MY.NET.152.183
MY.NET.153.119	MY.NET.153.145	MY.NET.88.251	MY.NET.88.254
MY.NET.153.108	MY.NET.153.118	MY.NET.152.12	MY.NET.150.97
MY.NET.153.124	MY.NET.152.247	MY.NET.152.248	MY.NET.152.21
MY.NET.153.112	MY.NET.152.215	MY.NET.153.106	MY.NET.152.160
MY.NET.153.143	MY.NET.153.123	MY.NET.151.73	MY.NET.152.172
MY.NET.153.141	MY.NET.152.162	MY.NET.150.103	MY.NET.152.157
MY.NET.153.113	MY.NET.152.19	MY.NET.152.175	MY.NET.152.46
MY.NET.88.148	MY.NET.152.216	MY.NET.150.210	MY.NET.149.27
MY.NET.153.144	MY.NET.88.151	MY.NET.152.16	MY.NET.152.178
MY.NET.153.111	MY.NET.152.249	MY.NET.153.71	MY.NET.151.64
MY.NET.153.120	MY.NET.153.109	MY.NET.152.163	MY.NET.153.135
MY.NET.153.114	MY.NET.153.137	MY.NET.152.161	MY.NET.152.171
MY.NET.153.142	MY.NET.153.126	MY.NET.153.140	MY.NET.150.226
MY.NET.153.125	MY.NET.152.11	MY.NET.152.15	MY.NET.152.164
MY.NET.153.115	MY.NET.153.127	MY.NET.152.244	MY.NET.88.140
MY.NET.153.117	MY.NET.88.171	MY.NET.152.182	MY.NET.152.166
MY.NET.88.243	MY.NET.153.107	MY.NET.152.169	

SMB Name Wildcard

This alert is triggered by Windows hosts requesting NetBIOS resources from other machines. [13] The "wildcard" indicates a request for all records, and is initiated with the command "nbtstat –a [IP address]".

---

[10] http://www.sans.org/newlook/digests/unicode.htm
[11] http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS00-078.asp
[12] http://www.cert.org/advisories/CA-2001-26.html
[13] http://www.sans.org/newlook/resources/IDFAQ/port_137.htm

All the source and destination addresses for this alert are contained within MY.NET.0.0/16, so it is assumed that NetBIOS traffic is blocked at the perimiter (as it should be).

This traffic is a part of normal Microsoft networking and should not be considered suspicious. To reduce the amount of "informational" activity in the alerts, you can configure this rule to use the "log" facility only.

```
log udp any any -> 192.168.1.0/24 137 (msg:"SMB Name Wildcard"; content:"CKAAA
AAAAAAAAAAAAAAAAAAAAAAAA|0000|";)
```

spp_http_decode: CGI Null Byte attack detected

This alert is triggered by a preprocessor, not a signature, and indicates the presence of a null byte (%00) at the end of a CGI request. It can sometimes be used to view the sourcecode of the CGI script, revealing passwords, paths, hostnames, or whatever else was encoded into the script.[14]

Only 2 hosts triggering this alert are external, one from an ATT.net dialup and the other from a RoadRunner cablemodem. The first targets only MY.NET.5.96, generating 16 alerts. The second is more malicious.

This host attempted several different attacks against webserver MY.NET.153.159, which by the alert pattern is assumed to be a Microsoft IIS host. Other alerts include "WEB-CGI Scriptalias access", "WEB-IIS encoding access", "IIS Unicode attack" and "WEB-MISC directory traversal". These occur over a period of about 40 minutes, and indicate specific targeting.

MY.NET.153.159 has 684 alerts as the source, including several "IIS Unicode attack" alerts to outside hosts and "IRC access" which is unusual for a server. This host is definitely compromised and should be immediately inspected.

An investigation of the source 24.162.83.132 is also warranted.

```
Server# whois 24.162.83.132@whois.arin.net
[whois.arin.net]
ServiceCo LLC - Road Runner (NET-ROAD-RUNNER-5)
   13241 Woodland Park Road
   Herndon, VA 20171
   US

   Netname: ROAD-RUNNER-5
   Netblock: 24.160.0.0 - 24.170.127.255
   Maintainer: SCRR

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

   Domain System inverse mapping provided by:
```

[14] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0149

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

       Record last updated on 06-Aug-2001.
       Database last updated on  27-May-2002 19:57:52 EDT.

    The ARIN Registration Services Host contains ONLY Internet
    Network Information: Networks, ASN's, and related POC's.
Please use the whois server at rs.internic.net for DOMAIN related
Information and whois.nic.mil for NIPRNET Information.
```

ICMP Echo Request L3retriever Ping

This event may indicate that someone is scanning your network using the L3 "Retriever 1.5" security scanner. This legitimate security tool, marketed by Symantec[15], is for authorized security assessment and should not be used on unauthorized networks.[16]

All alert source and destinations are contained within MY.NET.0.0/16.

Significant amounts of this traffic (300-800 alerts/IP) originate from MY.NET.152.0/24 and even higher amounts (2500-9000 alerts) originate from MY.NET.88.203, MY.NET.88.207, MY.NET.88.251. The majority of this traffic is directed at MY.NET.11.7, MY.NET.11.6, and MY.NET.11.5 – whose only other alerts are "SMB Name Wildcard" or "ICMP Echo Request".

In my experience the signature for "L3Retreiver Ping" is not as specific as references indicate, and is frequently a "false alarm". Packet dumps could confirm this and assist in identification of the exact source.

INFO MSN IM Chat Data

"MSN Instant Messenger" is a realtime text/audio/video chat client that is similar to AOL instant messenger and Yahoo instant messenger. Depending on the mood of AOL and the ability of the programmers at Microsoft, they also periodically interoperate with each other.

Large amounts of MSN Messenger traffic is seen to/from hosts in MY.NET.88.0/16, MY.NET.150.0/16, MY.NET.151.0/16, MY.NET.152.0/16 and MY.NET.153.0/16.

If use of MSN Messenger software is prohibited by your security policy, a block on TCP/1863 in both directions will stop it until users configure it to use HTTP. Blocking *that* will require use of a application proxy, or an IDS system with response capability (eg: ./configure –with-flexresp).

[15] http://www.symantec.com/press/security/n990923_ns.html
[16] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids311&view=event

MISC Large UDP Packet

This event indicates that an abnormally large UDP packet was sent to your server. This may indicate a denial of service attack or the use of a covert channel.[17] My reference Sort ruleset triggers this alert on datagrams larger than 4000 bytes :

```
alert udp $EXTERNAL_NET any -> $HOME_NET any (msg:"MISC Large UDP Pac
ket"; dsize: >4000; reference:arachnids,247; classtype:bad-unknown; sid:521; rev:1;
sid-msg.map:521 || MISC Large UDP Packet || arachnids,247
```

The major external sources for this alert are 61.78.35.42 and 61.78.35.44 (unregistered, Korea), 63.240.15.205 and 62.240.15.207 (owned by AT&T in California), 163.239.2.31 (Sogang University, Korea), 210.94.0.106 (Hanaro Telecom, Korea) – the only other alert for these hosts is "ICMP Fragment Reassembly Time Exceeded".

Analysis of the internal destinations for this alert indicate policy violations and compromised hosts. The following hosts should be immediately inspected :

MY.NET.153.110 : Nimda, Nmap activity, Chat activity.
MY.NET.153.121 : Nimda, Nmap activity, CGI attacks, IIS Unicode attacks.
MY.NET.153.147 : Nimda, Red Worm, Nmap activity, Chat activity, watched net activity.
MY.NET.153.157 : Red Worm, Chat activity, Napster, IIS Unicode attacks
MY.NET.153.171 : Nimda, Red Worm, Back Orfice, FTP globbing, Chat activity, CGI attacks.
MY.NET.153.164 : Nimda, Red Worm, Nmap activity, Chat activity, watched net activity.
MY.NET.153.153 : Nimda, Red Worm, Back Orfice, FTP globbing, Chat activity, CGI attacks,
Napster, TFTP traffic, IRC traffic

High port 65535 udp – possible Red Worm traffic

The "Red Worm" is more commonly known as the "Adore Worm", and is similar to the Ramen and Lion worms. Adore scans the Internet checking Linux hosts to determine whether they are vulnerable to any of the following well-known exploits: LPRng, rpc-statd, wu-ftpd and BIND. LPRng is installed by default on Red Hat 7.0 systems.[18] [19]

Traffic on port 65535 is unusual, and a strong indication of an infection. The following hosts should be examined :

MY.NET.152.166 MY.NET.152.245 MY.NET.153.170 MY.NET.5.79 MY.NET.149.65

[17] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids247&view=event
[18] http://www.sans.org/y2k/adore.htm
[19] http://www.redhat.com/support/alerts/Adore_worm.html

MY.NET.6.45
MY.NET.6.48
MY.NET.6.49
MY.NET.6.50
MY.NET.6.51
MY.NET.6.52
MY.NET.6.53
MY.NET.6.60
MY.NET.60.43
MY.NET.88.148
MY.NET.149.12
MY.NET.149.16
MY.NET.149.23
MY.NET.149.27
MY.NET.149.34
MY.NET.149.39
MY.NET.149.40
MY.NET.149.46
MY.NET.149.47
MY.NET.149.49
MY.NET.149.53
MY.NET.149.55
MY.NET.149.56
MY.NET.149.60
MY.NET.149.64
MY.NET.149.66
MY.NET.151.191
MY.NET.152.11
MY.NET.152.12
MY.NET.152.15
MY.NET.152.16
MY.NET.152.17
MY.NET.152.18
MY.NET.152.19
MY.NET.152.20
MY.NET.152.21
MY.NET.152.22
MY.NET.152.44
MY.NET.152.45
MY.NET.152.46
MY.NET.152.152
MY.NET.152.157
MY.NET.152.158
MY.NET.152.159
MY.NET.152.160
MY.NET.152.161
MY.NET.152.162
MY.NET.152.163
MY.NET.152.164
MY.NET.152.165
MY.NET.152.167
MY.NET.152.168
MY.NET.152.169
MY.NET.152.170
MY.NET.152.171
MY.NET.152.172
MY.NET.152.173
MY.NET.152.174
MY.NET.152.175
MY.NET.152.176
MY.NET.152.177
MY.NET.152.178
MY.NET.152.179
MY.NET.152.180
MY.NET.152.181
MY.NET.152.182
MY.NET.152.183
MY.NET.152.184
MY.NET.152.185
MY.NET.152.186
MY.NET.152.213
MY.NET.152.214
MY.NET.152.215
MY.NET.152.216
MY.NET.152.244
MY.NET.152.246
MY.NET.152.247
MY.NET.152.249
MY.NET.152.250
MY.NET.152.251
MY.NET.153.140
MY.NET.153.141
MY.NET.153.142
MY.NET.153.144
MY.NET.153.146
MY.NET.153.147
MY.NET.153.148
MY.NET.153.150
MY.NET.153.152
MY.NET.153.154
MY.NET.153.159
MY.NET.153.160
MY.NET.153.162
MY.NET.153.163
MY.NET.153.164
MY.NET.153.165
MY.NET.153.166
MY.NET.153.167
MY.NET.153.168
MY.NET.153.169
MY.NET.153.172
MY.NET.153.173
MY.NET.153.174
MY.NET.153.177
MY.NET.153.179
MY.NET.153.180
MY.NET.153.182
MY.NET.153.184
MY.NET.153.185
MY.NET.153.186
MY.NET.153.187
MY.NET.153.188
MY.NET.153.189
MY.NET.153.196
MY.NET.153.197
MY.NET.153.198
MY.NET.153.200
MY.NET.153.203
MY.NET.153.204
MY.NET.153.205
MY.NET.153.206
MY.NET.153.207
MY.NET.153.208
MY.NET.153.209
MY.NET.153.210

INFO Inbound GNUTella Connect request

GNUTella is an open-source file sharing package similar to Napster, Kazaa, Morpheus, etc.
There are a number of worms and viruses that can travel along this network, although GNUTella is specifically no worse than the others.

GNUTella traffic was detected to the following internal hosts :

MY.NET.150.209 MY.NET.153.143 MY.NET.153.164 MY.NET.153.174 MY.NET.153.211
MY.NET.152.164 MY.NET.153.153 MY.NET.153.170 MY.NET.153.175
MY.NET.152.185 MY.NET.153.160 MY.NET.153.171 MY.NET.153.194

If use of file-sharing software is prohibited by your security policy, a block on IP ports 6346 and 6347[20] in both directions will specifically stop GNUTella and all other clients using the GNUTella network (notably Morpheus and MusicCity).

[20] http://www.iana.org/assignments/port-numbers

ICMP Echo Request Nmap or HPING2

Nmap[21] and HPING[22] are powerful pieces of software that automates a variety of stealth scanning techniques and packet injection and are frequently used by other software and/or potential troublemakers to map the services on the target network.

This activity should be considered attempted reconnaissance when originating from external hosts, and a policy violation when originating internally. Traffic of internal origin can also indicate a compromised host that is testing other internal hosts for further attack.

There are 63 hosts within MY.NET.0.0/16 that triggered this alert, and all but one had traffic destined for MY.NET.11.6 or MY.NET.11.7 – which had other ICMP alerts along with some SMB traffic. Various other software can trigger this alert, but because there are consistently only 2 target IPs, these are likely benign.

MY.NET.253.10, triggered this alert 308 times to 299 different destination addresses. Other alerts from this host include Nmap TCP fingerprinting, Null scans, and Nmap TCP pings. This is obviously not normal and an investigation of this host and/or its user is in order.

Watchlist 000220 IL-ISDNNET-990517

This appears to be a local rule configured to alert to traffic to/from a specific network, probably because of past suspicious activity. In my experience, this is done to do full logging of *all* traffic on a host, regardless of if it triggers a Snort rule or not, but without access to the ruleset in use, I cannot determine the specific intent.

Ripe has IL-ISDNET-990517 as a provider in Israel. Traffic that triggered this rule is primarily a mix of TCP/80 and TCP/1214 (Kazaa) traffic. It should be determined if this rule is still required, and what specifically should be logged.

```
Server# whois IL-ISDNNET-990517@whois.ripe.net
[whois.ripe.net]
% This is the RIPE Whois server.
% The objects are in RPSL format.
% Please visit http://www.ripe.net/rpsl for more information.
% Rights restricted by copyright.
% See http://www.ripe.net/ripencc/pub-services/db/copyright.html

inetnum:      212.179.0.0 - 212.179.255.255
netname:      IL-ISDNNET-990517
descr:        PROVIDER
country:      IL
admin-c:      NP469-RIPE
tech-c:       TP1233-RIPE
tech-c:       ZV140-RIPE
tech-c:       ES4966-RIPE
status:       ALLOCATED PA
```

[21] http://www.insecure.org/nmap/nmap_documentation.html
[22] http://www.hping.org/manpage.html

```
mnt-by:       RIPE-NCC-HM-MNT
changed:      hostmaster@ripe.net 19990517
changed:      hostmaster@ripe.net 20000406
changed:      hostmaster@ripe.net 20010402
source:       RIPE

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

FTP DoS ftpd globbing

The FTP protocol is one of the least secure protocols that can be run. In addition to the various daemon vulnerabilities that continue to be discovered, commands (passwords included) are transmitted in cleartext, and in are generally contained within a single packet (requiring no stream reassembly as telnet would).

"Globbing" is the ability to use wildcards and pattern matching strings (similar to the behavior of most UNIX shells), so that the command mget *.c means retrieve all the files ending in ".c," and get ~foo/file.name means get the file named "file.name" in the home directory of foo. [23]

The ability of a remote or local user to deliver input patterns to glob() implementations allows for two general types of security exposures: glob() expansion vulnerabilities (essentially a buffer overflow where the daemon incorrectly assumes that the length of the user input is limited to the number of characters that are read in from the socket, typically 512 characters), and glob() implementation vulnerabilities (buffer overflows in their internal utility functions typically triggered by requesting a pattern that expands to a very large pathname, or by submitting a pattern that the user intends to have the daemon process twice). [24]

A successful attack results in arbitrary command execution with the permissions of the FTP daemon. The following internal hosts should be audited :

MY.NET.88.233	MY.NET.152.172	MY.NET.152.180	MY.NET.153.153	MY.NET.153.186
MY.NET.151.109	MY.NET.152.174	MY.NET.152.183	MY.NET.153.164	MY.NET.153.194
MY.NET.152.164	MY.NET.152.178	MY.NET.152.185	**MY.NET.153.171**	MY.NET.153.197
		MY.NET.153.150		

MY.NET.153.171 deserves the most immediate attention as it is responsible for 13 other alerts, including Back Orifice and Red Worm activity.

[23] http://www.cert.org/advisories/CA-2001-07.html
[24] http://www.pgp.com/research/covert/advisories/048.asp

ICMP Fragment Reassembly Time Exceeeded

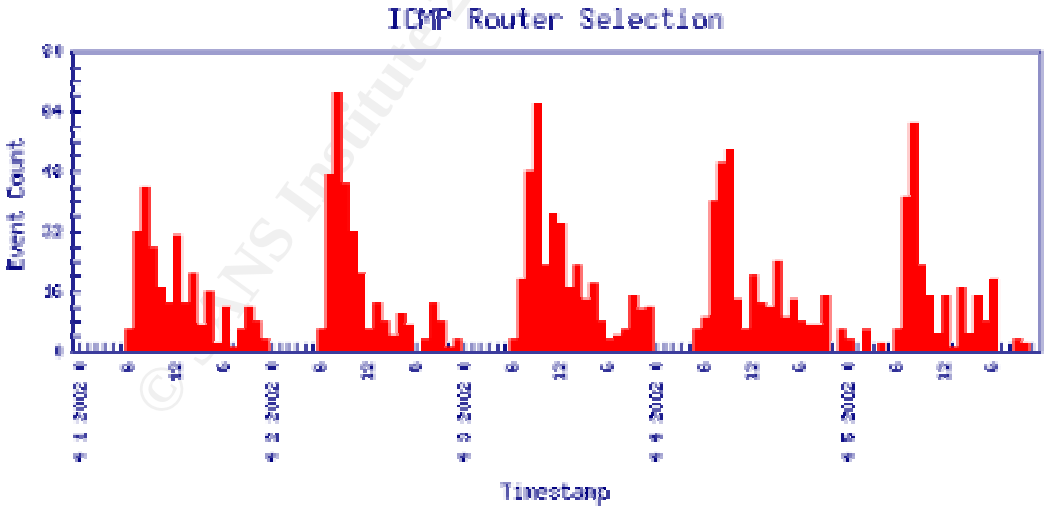
This alert records ICMP type 11 (Time Exceeded), Code 1 (Fragment) packets and is informational only.

The majority of these alerts are triggered by Asian hosts (the majority of those being in Korea) and by what appear to be streaming content sites. In the case of international links, some fragmentation is expected as timeouts are encountered en route; in the case of streaming content, it is possible that network overloads are causing these lower-priority packets to be dropped.

Again, however, MY.NET.153.171 is a major source of these alerts, and as mentioned previously, this host is almost certainly compromised by multiple Trojans and should be given top priority.

ICMP Router Selection

This alert records ICMP type 10 (Router Selection), Code 0 (Undef) packets and is informational only. All alerts were destined for the "all-routers" multicast of 224.0.0.2, and all sources were within MY.NET. While possible to generate these packets in order to change the default route entries on a Windows machine, possibly for denial of service attacks,[25] the time-distribution pattern (below) argues against any malicious activity – alert frequency is highest in the early morning, then step-decaying rapidly – a pattern which would be expected in normal use.



[25] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids174&view=event

WEB-IIS view source via translate header

Microsoft IIS 5.0 allows remote attackers to obtain source code for .ASP files and other scripts via an HTTP GET request with a "Translate: f" header, known as the "Specialized Header" vulnerability.[26] [27] Source code contained within ASP files can reveal database passwords, hostnames, and a wealth of other internal information.

This alert can be triggered by legitimate WebDAV requests also[28], but without the ability to examine the raw logs, this cannot be determined.

MY.NET.5.96 and MY.NET.150.220 should be examined to ensure the patch discussed in Q256888[29] and MS00-058[30] has been applied.

NMAP TCP ping!

NMAP is an open-source tool which facilitates a variety of port and IP scans.[31] This alert is triggered by using the syntax "nmap –sP [target] [32]".

This activity should be considered attempted reconnaissance when originating from external hosts, and a policy violation when originating internally. Traffic of internal origin can also indicate a compromised host that is testing other internal hosts for further attack.

MY.NET.253.10 triggered this alert 788 times to 325 different destination addresses. Other alerts from MY.NET.253.10 indicate a variety of NMAP activity, including fingerprinting attempts and Null scans. There is clear evidence that someone is performing network reconnaissance from MY.NET.253.10.

WEB-MISC Attempt to execute cmd

This alert is triggered when the string "cmd.exe" is detected within a HTTP GET request. The most frequent cause is background noise from the internet caused by the Nimda virus, CodeRedII worm and sadmindIIS worm.[33] On compromised hosts, /winnt/system32/cmd.exe (the command interpreter) is copied into the /scripts directory on the webserver where it is executable under IIS. This permits an attacker to remotely execute commands as "localsystem" on the server.

[26] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0778
[27] http://online.securityfocus.com/bid/1578/discussion/
[28] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids305&view=event
[29] http://support.microsoft.com/default.aspx?scid=kb;en-us;Q256888
[30] http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms00-058.asp
[31] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids28&view=event
[32] http://www.insecure.org/nmap
[33] http://www.cert.org/advisories/CA-2001-26.html

The following servers should be examined (priorities in **bold**) to ensure that at a minimum, patch MS00-78[34] is applied as it fixes the vulnerability exploited by Nimda and several others. It would be best, however, to apply the most recent Microsft IIS cumulative security patch, MS02-018.[35]

Check for presence of "root.exe" (anywhere) or "cmd.exe" (anywhere *except* /winnt/system32) to indicate signs of compromise. An excellent tool to assist in repairing an infected host is available at ftp://ftp.f-secure.com/anti-virus/tools/fsnimda3.exe

MY.NET.5.79	MY.NET.88.217	MY.NET.150.83	MY.NET.150.147	MY.NET.150.243
MY.NET.5.92	MY.NET.150.6	MY.NET.150.84	MY.NET.150.195	MY.NET.150.246
MY.NET.5.95	MY.NET.150.16	MY.NET.150.101	MY.NET.150.197	MY.NET.151.77
MY.NET.5.96	MY.NET.150.41	MY.NET.150.107	MY.NET.150.220	MY.NET.151.114
MY.NET.5.243	MY.NET.150.51	MY.NET.150.133	MY.NET.150.226	MY.NET.153.208
MY.NET.88.156	MY.NET.150.59	MY.NET.150.139	MY.NET.150.228	MY.NET.153.220
MY.NET.88.187	MY.NET.150.63	MY.NET.150.143	MY.NET.150.231	

INFO Outbound GNUTella Connect request

GNUTella is an open-source file sharing package similar to Napster, Kazaa, Morpheus, etc.
There are a number of worms and viruses that can travel along this network, although GNUTella is specifically no worse than the others.

A list of internal hosts using this software is available in this report under the heading **INFO Inbound GNUTella Connect request**

Blocking IP ports 6346 and 6347 at the perimeter will effectively stop internal clients from using any software which utilizes the GNUTella network.

WEB-IIS _vti_inf access

Microsoft Frontpage extensions are special virtual directories and files placed on an IIS webserver to permit Frontpage to read and directly publish websites to the server. This is one of them, and they are notoriously full of security problems.[36] [37]

Microsoft publishes a whitepaper on the "security of frontpage extensions"[38], but unless there is some critical business need for this functionality, I strongly recommend that these virtual directories be removed and their extensions unregistered within IIS.

[34] http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms00-078.asp
[35] http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms02-018.asp
[36] http://www.insecure.org/sploits/Microsoft.frontpage.insecurities.html
[37] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0341

This signature can be triggered by legitimate Frontpage posting activity, but also by Nimda traffic which has become part of the background noise on the internet. Presence of the following request in the server logs specifically indicates Nimda scans:[39]

```
GET /_vti_bin/..%5c../..%5c../..%5c../winnt/system32/cmd.exe?/c+dir
```

MY.NET.5.96 is the lone target for this alert, and it appears to be a default Microsoft IIS installation that is being attacked by most every method in the book (see table of alerts below). I strongly recommend following one of the "best practice" guides for securing Microsoft IIS and Windows NT servers available at the following URLs

http://nsa2.www.conxion.com/
http://rr.sans.org/web/web_apps.php
http://rr.sans.org/web/fix.php

~ MY.NET.5.96 ~

Alert Signature	#events	Alert Signature	#events
WEB-CGI scriptalias access	155	WEB-MISC Attempt to execute cmd	3
WEB-IIS encoding access	2	WEB-MISC 403 Forbidden	11
WEB-MISC webdav search access	2	WEB-MISC whisker head	2
WEB-MISC http directory traversal	5	ICMP Dest Unreachable (Protocol)	2
ICMP Echo Request L3retriever Ping	71	SNMP public access	2209
SMB Name Wildcard	642	CGI Null Byte attack detected	172
WEB-IIS _vti_inf access	322	WEB-CGI ksh access	74
WEB-FRONTPAGE _vti_rpc access	299	TCP/55850 - Possible myserver activity	1
WEB-IIS view source via translate header	1297	Queso fingerprint	3
IIS Unicode attack detected	1	Possible trojan server activity	5

Watchlist 000222 NET-NCFC

This appears to be a local rule configured to alert to traffic to/from a specific network, probably because of past suspicious activity. In my experience, this is done to do full logging of *all* traffic on a host, regardless of if it triggers a Snort rule or not, but without access to the ruleset in use, I cannot determine the specific intent.

Arin has NET-NCFC as the Academy of Sciences in China. Traffic that triggered this rule is primarily a mix of TCP/4662 (eDonkey –file sharing software similar to Kazaa) and TCP/1752[40] or TCP/1753[41] which are IANA registered ports for Leap-of-Faith and Translogic license managers (additional investigation is warranted). It should be determined if this rule is still required, and what specifically should be logged.

---

[38] http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnservext/html/fp2ksecuritywp.asp
[39] http://www.cert.org/advisories/CA-2001-26.html
[40] http://www.portsdb.org/bin/portsdb.cgi?portnumber=1752&protocol=ANY&String=
[41] http://www.portsdb.org/bin/portsdb.cgi?portnumber=1753&protocol=ANY&String=

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

   Netname: NCFC
   Netblock: 159.226.0.0 - 159.226.255.255

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

   Domain System inverse mapping provided by:

   NS.CNC.AC.CN                159.226.1.1
   GINGKO.ICT.AC.CN            159.226.40.1

   Record last updated on 25-Jul-1994.
   Database last updated on  28-May-2002 23:38:30 EDT.
```

The following internal hosts had traffic involving "watchlist 000222" Those in **bold** deserve particular attention and based on other alerts they have triggered, are infected with one or more Trojans :

MY.NET.88.186 **MY.NET.150.143** **MY.NET.153.153** MY.NET.153.164

~ MY.NET.150.143 ~

Alert Signature	#events
EXPLOIT x86 setuid 0	3
IIS Unicode attack detected	2
WEB-MISC Attempt to execute cmd	9
EXPLOIT x86 setgid 0	1
INFO MSN IM Chat data	50
INFO FTP anonymous FTP	1
Watchlist 000222 NET-NCFC	242
Watchlist 000220 IL-ISDNNET-990517	1285
Queso fingerprint	13
SCAN Synscan Portscan ID 19104	1
65535/tcp - possible Red Worm - traffic	15
EXPLOIT x86 NOOP	1
ICMP Echo Request Nmap or HPING2	1
Possible trojan server activity	16
NMAP TCP ping!	3

~ MY.NET.153.153 ~

Alert Signature	#events
65535/tcp - possible Red Worm - traffic	117
ICMP Fragment Reassembly Time Exceeded	18
CGI Null Byte attack detected	2222
TFTP - Int. UDP conn. to Ext. tftp server	1
MISC Large UDP Packet	2129
connect to 515 from inside	281
INFO Possible IRC Access	17
IIS Unicode attack detected	1265
INFO Napster Client Data	3
Null scan!	50
INFO Outbound GNUTella Connect request	29
INFO Inbound GNUTella Connect request	89
Watchlist 000220 IL-ISDNNET-990517	382
SCAN FIN	1
Watchlist 000222 NET-NCFC	50
ICMP Echo Request Nmap or HPING2	1
NMAP TCP ping!	1
WEB-MISC compaq nsight directory traversal	8
FTP DoS ftpd globbing	118

ICMP Echo Request Windows

This rule alerts only to "ping" traffic from a Windows source. This activity would be expected as a normal part of Microsoft networking. It is an informational rule only.

MY.NET.5.87 did several extended ping requests to a variety of hosts ("ping –t [host]") – but triggered no other alerts and is therefore not deemed suspicious.

WEB-FRONTPAGE _vti_rpc access

Microsoft Frontpage extensions are special virtual directories and files placed on an IIS webserver to permit Frontpage to read and directly publish websites to the server. This is another of them (the other is "_vti_inf").

Recommendations and and a list of vulnerabilities at the target (MY.NET.5.96) is available in this report under the heading **WEB-IIS _vti_inf access**.

Null Scan!

NMAP is an open-source tool which facilitates a variety of port and IP scans.[42] This alert is triggered by using the syntax "nmap –sN [target] [43]". And results in a packet with none of the TCP flags set (an illegal condition). This trick is used to sneak a scan attempt past a firewall, but Snort will catch it every time.

This activity should never be seen in a network under "normal" conditions and should be considered a reconnaissance attempt.

MY.NET.186.16 and MY.NET.253.10 are the only two internal hosts, but neither triggered any other alerts. MY.NET.186.16 in particular triggered this alert 143 times – always with a source port of TCP/23 (telnet), and destination ports of TCP/1987 and TCP/1111. Something is probably awry with this host, and further examination is in warranted.

WEB-CGI scriptalias access

The ScriptAlias problem is inherent in both NCSA httpd (all versions up to and including 1.5) and Apache httpd prior to 1.0. The problem is that configuring a ScriptAlias directory within the Document Root permits users to retrieve a CGI program rather than execute it. This will allow remote users to download scripts instead of executing them. In

[42] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids28&view=event
[43] http://www.insecure.org/nmap

effect this will give the attacker the ability to search your CGI forms for weaknesses and or download proprietary programs.[44] [45]
MY.NET.5.96 and MY.NET.153.159 were targeted with MY.NET.5.96 catching the majority (99%). MY.NET.5.96 has been previously identified as a host which exhibits clear indications of compromise.

68.55.176.169 (pcp233448pcs.elictc01.md.comcast.net – a Maryland Comcast cablemodem subscriber) appears to have found, then systematically tested (140 times) to exploit this vulnerability.

```
Server# whois 68.55.176.169@whois.arin.net
[whois.arin.net]
Comcast Cable Communications, Inc. (NETBLK-JUMPSTART-1) JUMPSTART-1
                                                  68.32.0.0 - 68.63.255.255
Comcast Cable Communications, Inc. (NETBLK-JUMPSTART-BALTIMORE-A) JUMPSTART-BALTIMORE-
A
                                                 68.54.80.0 - 68.55.255.255

Server# whois NETBLK-JUMPSTART-BALTIMORE-A@whois.arin.net
[whois.arin.net]
Comcast Cable Communications, Inc. (NETBLK-JUMPSTART-BALTIMORE-A)
   7377 Washington Blvd.
   Baltimore, MD 21227
   US

   Netname: JUMPSTART-BALTIMORE-A
   Netblock: 68.54.80.0 - 68.55.255.255

   Coordinator:
      Zeibari, Greg  (GZ64-ARIN)  gzeibari@comcastpc.com
      856-661-7929

   Domain System inverse mapping provided by:

   NS01.JDC01.PA.COMCAST.NET    66.45.25.71
   NS02.JDC01.PA.COMCAST.NET    66.45.25.72

   Record last updated on 15-Jan-2002.
   Database last updated on  28-May-2002 23:38:30 EDT.
```

I would suggest attempting to gather relevant server logs and other forensic data (if available) such as paket dumps from the IDS and prepare a formal complaint against Comcast (abuse@comcast.net). Depending on the amount of damage done and the amount of documentation that can be gathered as evidence, you may wish to submit the case for prosecution as well.

Possible trojan server activity

All of these alerts involve port TCP/27374 – activity associated with the Ramen worm[46] [47] [48].

[44] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids227&view=event
[45] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0236
[46] http://www.cert.org/incident_notes/IN-2001-01.html
[47] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids460&view=event
[48] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids461&view=event

Servers are infected in one of three ways :

- wu-ftpd (port 21/tcp) : Format string input validation error in wu-ftpd site_exec() function[49]
- rpc.statd (port 111/udp) : Remote root compromise via format string stack overwrite[50]
- lprng (port 515/tcp) : Input as a format string parameter to syslog() calls[51]

Once the worm has infected the host, it binds xinetd or inetd to port 27374 and accepts connections. It also will begin a massive effort to scan, then attempt to exploit, other servers in the network

A block on IP port 27374 in both directions at the permiter should implemented immediately.

The following internal hosts should be investigated. A general guideline for recovering from a root compromise is available at http://www.cert.org/tech_tips/root_compromise.html

MY.NET.5.29	MY.NET.5.44	MY.NET.5.83	MY.NET.150.113	MY.NET.191.20
MY.NET.5.42	MY.NET.5.50	MY.NET.5.96	MY.NET.150.143	
MY.NET.5.43	MY.NET.5.55	MY.NET.70.229	MY.NET.185.28	

SCAN Proxy Attempt

These alerts indicate an attempt to use a host on MY.NET.0.0/16 as an HTTP proxy. Unsecured proxy servers are commonly used to avoid filtering software and to conceal the identity of users as they visit sites they'd rather not be identified as using. All alerts are for TCP/1080 (socks-proxy) and TCP/8080 (http-proxy).

Access to external proxy servers is usually prohibited by security policy. Access to an internal proxy from the internet at large is an obvious security risk. A block on TCP/1080 and TCP/8080 in both directions at the perimeter is recommended.

Check the following servers to determine if an HTTP or SOCKS proxy is running. Determine if its use is required, and if not, disable the services. If their use is required, consult the documentation on how to configure the service so it permits connections only from MY.NET.0.0/16.

MY.NET.88.165	MY.NET.152.11	MY.NET.152.157	MY.NET.152.172	MY.NET.153.187
MY.NET.150.113	MY.NET.152.44	MY.NET.152.162	MY.NET.153.117	
MY.NET.151.79	MY.NET.152.46	MY.NET.152.166	MY.NET.153.171	

[49] http://www.kb.cert.org/vuls/id/29823
[50] http://www.kb.cert.org/vuls/id/34043
[51] http://www.kb.cert.org/vuls/id/382365

INFO napster login

Napster was a popular file-sharing program until being sued out of existence by the RIAA and several record labels. It's use should be governed by a security policy, but it should not continue to be an issue (although many other peer-to-peer programs serving the same purpose have taken its place).

ICMP Destination Unreachable (Communication Administratively Prohibited)

This alert records ICMP type 3 (Destination Unreachable), Code 13 (Administratively Prohibited) packets and is informational only. Firewalls and routers return this ICMP type/code when a request is denied by an ACL.

It happened 103 times, and always between MY.NET.150.1 and MY.NET.150.24. Assuming MY.NET is configured as most typical networks are, MY.NET.150.1 is probably a router. Investigation of the host MY.NET.150.24 to determine what it keeps attempting to do may be in order.

ICMP Traceroute

ICMP is a wonderful way to map an unknown network, and the "traceroute" facility is no exception. It is also a commonly used troubleshooting tool by system and network administrators alike.

Of 91 events, there were 33 sources and 6 destinations, all within MY.NET. The most common was MY.NET.152.1. Assuming MY.NET is configured as most typical networks are, MY.NET.152.1 is probably a router.

This activity is evenly distributed among hosts and time (business day), and is assumed to be a part of normal network troubleshooting and therefore not suspicious.

INFO Napster Client Data

Napster was a popular file-sharing program until being sued out of existence by the RIAA and several record labels. It's use should be governed by a security policy, but it should not continue to be an issue (although many other peer-to-peer programs serving the same purpose have taken its place).

Despite this, there are still internal hosts which attempt to use it. Since Napster has been offline since well before the analysis period, I am unsure as to how this is possible. My reference Snort ruleset alerts to the presence of ".mp3" in a request involving ports TCP/5555, TCP/6666, TCP/6699, and TCP/7777.

This alert may be falsely triggered by other file-sharing packages, but in absence of the full packet logs of this traffic it is impossible to determine.

INFO Possible IRC access

The Internet Relay Chat (IRC) protocol[52] was developed to provide an interactive text-based messaging system. ICQ and mIRC are examples of how developers added a graphical interface to it. It's use is still popular among the tech and hacker communities, but general use has given way to more user-friendly packages such as AOL, MSN, and Yahoo instant messenger.

IRC is also commonly used as a control channel for denial-of-service programs ("bots") which sleep on a host until receiving instructions in an IRC channel. An excellent description of how this works is available at http://grc.com/dos/drdos.htm

The following internal hosts triggered this alert. Those in **bold** deserve attention because based upon other alerts they are generally responsible for a large amount of *other* malicious activity.

MY.NET.150.113	**MY.NET.152.20**	**MY.NET.153.145**	**MY.NET.153.161**	**MY.NET.153.188**
MY.NET.150.165	**MY.NET.152.161**	**MY.NET.153.147**	**MY.NET.153.170**	**MY.NET.153.189**
MY.NET.151.79	MY.NET.153.105	**MY.NET.153.153**	**MY.NET.153.177**	**MY.NET.153.193**
MY.NET.151.110	MY.NET.153.115	**MY.NET.153.154**	**MY.NET.153.181**	**MY.NET.153.196**
MY.NET.152.11	**MY.NET.153.141**	**MY.NET.153.159**	**MY.NET.153.186**	

Blocking ports 6667, 6668, 6669, and 7000 outbound at the permiter would stop this activity based on the patterns observed in the logs.

WEB-CGI ksh access

"ksh" is a general-purpose UNIX command interpreter. Its presence in the cgi-bin directory of a webserver would allow an attacker to execute any command on the remote host which is executable by the interpreter.[53] [54]

Examine MY.NET.5.96 and ensure that command interpreters such as sh, csh, ksh, bash, perl, etc. are not in the cgi-bin directory (or anywhere else they could be executed by a remote user via the httpd). MY.NET.5.96 has been identified previously as having numerous problems relating to default installations, and is clearly infected with several Trojans.

[52] http://rfc.net/rfc1459.html
[53] http://www.cert.org/advisories/CA-1996-11.html
[54] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0509

ICMP Echo Request BSDtype

This alert records ICMP type 8 (Echo), Code 0 (Undefined) packets and is informational only.

This particular ping was probably generated by BSD/OS, FreeBSD, NetBSD, OpenBSD 2.5, Linux, or Solaris 2.5-2.7. [55]

MISC Traceroute

This alert records ICMP type 30 (Traceroute), Code 0 (Undefined) packets and is generally informational. While traceroute can be used to learn the location of perimeter firewalls and routers, it is frequently used as a troubleshooting tool.[56]

192.204.106.2 (unregistered host at Verio) triggered this alert 35 times (which is statistically excessive compared to the others). The only alerts observed for that host were GNUTella traffic, so a traceroute function may be part of the software in order to determine network latency. 216.136.171.200 (unregistered host at Exodus) triggered it 8 times, the only other alert regarding an RCP attempt which contained bin_sh. The other alerts are not suspicious.

INFO – Possible Squid Scan

Squid is a web-proxy and cache which is open-source and runs on most flavors of UNIX.[57] Proxy servers are frequently used to centrally log internet requests and to cache frequently accessed content. When misconfigured, it is also used by external clients to conceal activity and evade web filters.

Examine the following servers to determine if they are actually authorized proxy servers. If not, disable the http-gw service. If Squid proxys are not required or are prohibited by policy, an ingress filter for TCP/3128 will stop it.

MY.NET.88.165 MY.NET.152.11 MY.NET.152.46 MY.NET.152.162 MY.NET.152.172
MY.NET.150.113 MY.NET.152.44 MY.NET.152.157 MY.NET.152.166 MY.NET.153.117
MY.NET.151.79

[55] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids152&view=event
[56] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids118&view=event
[57] http://www.squid-cache.org/

INFO FTP anonymous FTP

Anonymous FTP is frequently used for software distribution. This service is generally centralized on a DMZ host within the network, but it appears that 15 different hosts within MY.NET.0.0/16 are permitting anonymous FTP from external clients.

It should be determined if the following hosts are permitted by policy to operate as FTP servers :

MY.NET.5.79	MY.NET.150.41	MY.NET.150.107	MY.NET.150.147	MY.NET.150.243
MY.NET.5.137	MY.NET.150.59	MY.NET.150.139	MY.NET.150.197	MY.NET.151.114
MY.NET.150.16	MY.NET.150.83	MY.NET.150.143	MY.NET.150.228	MY.NET.153.219

If you wish to prevent internal hosts from acting as FTP servers, an ingress filter for TCP/20 and TCP/21 will do it. Explicit "permit" entries can be made in the ACL to allow FTP traffic to those servers designated for that function while preventing rouge ones from popping up.

Queso fingerprint

Queso is a fingerprinting tool which crafts packets and then evaluates the response in an attempt to discover the remote O/S type and version. It is similar to "nmap -O".

The use of Queso in or against your network would be clear evidence of reconnaissance activity, but examination of the external hosts which triggered this alert indicate that in many cases, they also had GNUTella activity. It is possible that the normal operation of GNUTella clients may falsely trigger this signature, but without the ability to examine the signature it is impossible to determine.

Attempted Sun RPC high port access

This alert indicates an information gathering attempt against a Solaris host.[58] [59]

The following internal hosts were targets for these attempts :

MY.NET.152.17	MY.NET.153.165	MY.NET.153.173	MY.NET.153.186	MY.NET.153.203
MY.NET.152.182	MY.NET.153.167	MY.NET.153.175	MY.NET.153.188	MY.NET.153.207
MY.NET.153.141	MY.NET.153.169	MY.NET.153.184	MY.NET.153.196	MY.NET.153.209
MY.NET.153.161	MY.NET.153.172	MY.NET.153.185	MY.NET.153.202	

A ingress filter for TCP/32771 should be immediately applied and the preceeding list of hosts inspected for signs of compromise.

[58] http://www.whitehats.com/info/IDS429
[59] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0632

EXPLOIT x86 NOOP

This alert indicates and attempt to overflow a daemon with a long sting of "0x90" characters – a trick named the "NOOP sled"[60].

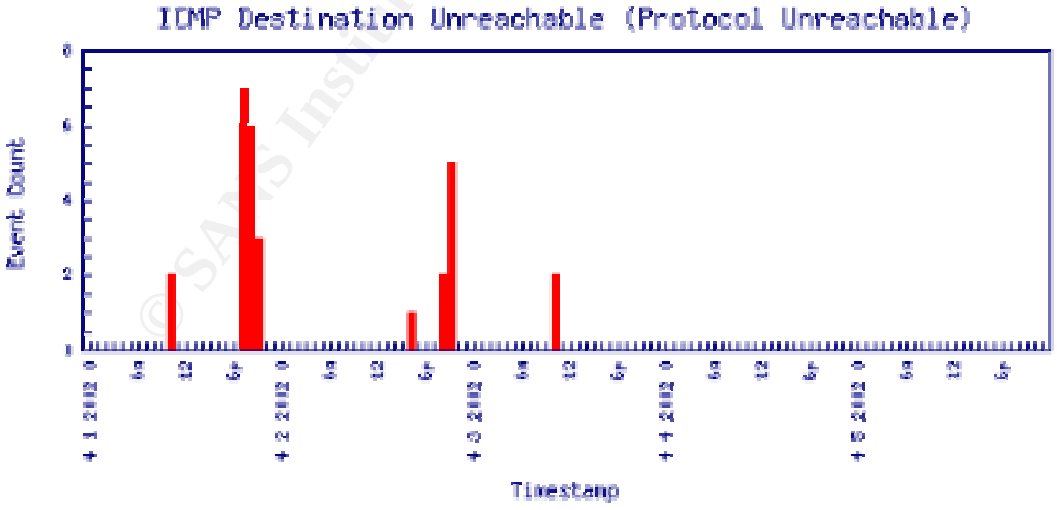
In my expierence, some web active content (eg: Macromedia) can trigger this alert. Supporting this theory is that in every instance, the source port was 80 and the destination port was empherical – indicating it was triggered by the reverse part of an established web session. That would not be the case in the event of an attack attempt (source would be empherical and destination would be 80).

More current versions of the Snort ruleset specifically exempt TCP/80 as a source port on shellcode alerts such as this.

ICMP Destination Unreachable (Protocol Unreachable)

This alert records ICMP type 3 (Destination Unreachable), Code 2 (Protocol Unreachable) packets and is informational only. A host returns this code when being sent a request for a protocol it doesn't speak.

All but 5 packets went to 24.200.165.198 (cable modem in Canada), and that was the only alert triggered for that host. MY.NET.150.41 was the primary source. This could be triggered by scanning activity, or simply be "background noise". A peak was noticed between 6-9pm on April 1, but without examination of the packet logs, an exact determination is impossible.



[60] http://www.giac.org/practical/David_Oborn_GCIA.html#detect4

WEB-MISC compaq nsight directory traversal

This event indicates an attempt to exploit a directory traversal vulnerability in the Compaq Web Management Agent. This allows a remote attacker to read arbitrary files.[61] It can also be triggered by a number of other benign events.

Examination of the alerts indicate that in every case the source port is 80 (destination is 2301, defined in the rule). My reference ruleset indicates the alert is triggered by the presence of "../" in the URL – the existence of which is entirely possible in otherwise normal traffic. More recent versions of Snort address this issue using "flows" – alerting only to one side of the TCP "conversation".

EXPLOIT NTPDX buffer overflow

This event is triggered by a buffer overflow attempt against the ntpd network time daemon. Some versions of ntpd and xntpd are vulnerable to remote root access in this manner.[62] [63]

All of the hosts triggering this alert also triggered others which indicate they have been compromised. Examination of the following hosts is in order :

MY.NET.88.155 MY.NET.152.246 MY.NET.153.45 MY.NET.153.46 MY.NET.153.211 MY.NET.151.125

If your network does not provide NTP (time) services to external clients, an ingress filter at the perimiter for UDP/123 is in order.

SCAN Synscan Portscan ID 19104

This alert is triggered by any TCP packet with the "SYN" flag set, and a TCP ID of 19104. Packets matching this criteria are generated by the "Synscan" tool[64].

Statistically, one would expect a TCP ID of any particular value (19104 included) to occur every so often. In your logs, 24 unique hosts triggered this alert, each only 1 time – indicating they are merely "noise".

[61] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids244&view=event
[62] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids492&view=event
[63] http://www.securityfocus.com/bid/2540
[64] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids521&view=event

Back Orifice

Back Orifice is a trojan written by the "Cult of the Dead Cow"[65] group, which succeeded in tricking large numbers of people into believing their software was for "remote administration". While not entirely false, it permits far more "remote administration" by far to many people than would ever be desired.

The signature in use on your sensor alerts to older versions of the package which use port UDP/31337. Newer versions do not restrict themselves to that port, but newer versions of Snort include a dedicated preprocessor to handle Back Orifice and Back Orifice 2000 (the latest version).

The following hosts were either sources or destinations for Back Orifice traffic (either case warrants investigation of the host and/or user).

MY.NET.6.48	MY.NET.152.13	MY.NET.153.211	MY.NET.153.171	MY.NET.153.190
MY.NET.6.49	MY.NET.152.16	MY.NET.152.248	MY.NET.153.181	MY.NET.153.198
MY.NET.6.50	MY.NET.152.44	MY.NET.152.250	MY.NET.153.184	MY.NET.153.204
MY.NET.6.52	MY.NET.152.157	MY.NET.153.142	MY.NET.153.185	MY.NET.153.206
MY.NET.151.125	MY.NET.152.182	MY.NET.153.154	MY.NET.153.189	MY.NET.153.207

INFO napster upload request

Napster was a popular file-sharing program until being sued out of existence by the RIAA and several record labels. It's use should be governed by a security policy, but it should not continue to be an issue (although many other peer-to-peer programs serving the same purpose have taken its place).

Napster has been offline since well before the analysis period -- I am unsure as to how these alerts are triggered, but actual use of Napster is not possible. This alert may be falsely triggered by other file-sharing packages, but in absence of the full packet logs it is impossible to determine.

MYPARTY – Possible My Party infection

MyParty is a generic name for members of the BackDoor-FB.svr.gen trojan[66] [67]

It begins as a mass-mailing email worm that attempts to social-engineer users into clicking an attachment with a name resembling a URL (www.myparty.yahoo.com) -- because *.com is executable by default in the Windows shell and also a top-level domain name.

[65] http://www.cultdeadcow.com/
[66] http://vil.mcafee.com/dispVirus.asp?virus_k=99333
[67] http://www.cert.org/incident_notes/IN-2002-01.html

After executing the email the host attempts to download additional code from http://209.151.250.170 (unregistered host at Cyberverse Online in California).

MY.NET.153.170, MY.NET.153.193 and MY.NET.153.199 should be cleaned.

SUNRPC highport access!

This alert indicates a successful connection to the RPC service on a Solaris host.[68] [69]

MY.NET.6.39 and MY.NET.253.114 were sources but had no other alerts. MY.NET.88.130 was the sole destination, and also triggered an alert for RPC traffic containing "bin_sh" (described elsewhere in this document). An examination of MY.NET.88.130 is warranted.

WEB-MISC 403 Forbidden

This is triggered when a webserver returns "HTTP/1.1 403" to a client.

Two internal webservers, MY.NET.5.92 and MY.NET.5.96 returned this error to the following external hosts and should be examined to determine the intent behind these attempts :

12.91.163.139	65.100.92.136	131.50.151.8	172.131.124.8	204.210.31.231
63.125.55.223	68.3.150.2	131.118.250.187	198.26.130.37	211.100.25.198

High port 65535 tcp – possible Red Worm traffic

This alert appears earlier in this report (#9, using UDP rather than TCP as is the case here).

The "Red Worm" is more commonly known as the "Adore Worm", and is similar to the Ramen and Lion worms. Adore scans the Internet checking Linux hosts to determine whether they are vulnerable to any of the following well-known exploits: LPRng, rpc-statd, wu-ftpd and BIND. LPRng is installed by default on Red Hat 7.0 systems.[70] [71]

Traffic on port 65535 is unusual, and a strong indication of an infection. MY.NET.150.143 should be examined. The source, 61.218.163.176 also warrants investigation as they appear to have started this mess :

[68] http://www.whitehats.com/info/IDS429
[69] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0632
[70] http://www.sans.org/y2k/adore.htm
[71] http://www.redhat.com/support/alerts/Adore_worm.html

```
Server# whois 61.218.163.183@whois.twnic.net
[whois.twnic.net]
Taiwan Bai Her Industry Co., Ltd.
   No.575, Her Guang Rd.
   Changhua Taiwan
   TW

   Netname: TAIWAN-BAI-HER-IN-CH-NET
   Netblock: 61.218.163.176 - 61.218.163.191

   Administrator contact:
      Hung Bao Chen (HBC2-TW) hn84134829@hn.hinet.net
      TEL: +886-4-757-5496

   Technical contact:
      Hung Bao Chen (HBC2-TW) hn84134829@hn.hinet.net
      TEL: +886-4-757-5496
```

EXPLOIT x86 setuid 0

In theory, this signature is triggered by an exploit attempt where the attacker sent the setuid(0) system call for the x86 platform. It is the most effective when monitoring protocols that usually consist of plaintext printable ASCII to catch remote x86 exploits.[72] In practice, because the pattern matched is so short, it is frequently triggered by a wide variety of non-suspect activity.

My reference Snort ruleset alerts on "|b017 cd80|" on any port except 80 or 8080 (web traffic). Your alerts indicate a mix of web and Kazaa (filesharing) traffic in addition to some unknowns. Complete packet logs will be required to make an exact determination.

EXPLOIT x86 stealth noop

This event may indicate that someone attempted to overflow one of your daemons with jmp 0x02 "stealth nops".[73] In my experience however, some web active content (eg: Macromedia) can trigger this alert (along with other Shellcode alerts).

Supporting this theory is that in every instance, the source port was 80 and the destination port was empherical – indicating it was triggered by the reverse part of an established web session. That would not be the case in the event of an attack attempt (source would be empherical and destination would be 80).

Port 55850 tcp – Possible myserver activity – ref. 010313-1
Port 55850 udp – Possible myserver activity – ref. 010313-1

MyServer is an obscure Distributed Denial of Service (DDoS) agent for UNIX which uses TCP or UDP port 55850 for control.

[72] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids436&view=event
[73] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids291&view=event

Analysis of your logs for this signature indicate that Kazaa activity appears to trigger this alert, but the remainder are suspicious. The following hosts should be examined :

MY.NET.5.79 MY.NET.6.50 MY.NET.6.53 MY.NET.153.191
MY.NET.6.49 MY.NET.6.52 MY.NET.150.133

RPC tcp traffic contains bin_sh

Ideally, this alert indicates an attempt to exploit a vulnerability in an rpc service. The rule alerts to the string "/bin/sh" in rpc traffic, which is often seen in rpc service attacks.[74]

In the instances logged, the source port for every event was 80, and the sources that triggered it had no other alerts which would indicate suspicious activity.

WEB-MISC http directory traversal

Similar to the "Unicode directory traversal" alert (discussed earlier in this report), this attempt is more blatant in that "../' is explicitly used in the URL (rather than using the Unicode %5c for a backslash). A wide variety of different attack and reconnaissance methods attempt this trick.

Of the 4 hosts which attempted it, 24.162.83.132 (RoadRunner Cablemodem) and 68.49.32.46 (Comcast Cablemodem) also attempted other web attacks on MY.NET.0.0/16 and are clearly up to no good. Complaints to abuse@rr.com and abuse@comcast.net are probably in order

Examination of MY.NET.5.96 and MY.NET.153.159 is also in order.

IDS552/web-iis_IIS ISAPI Overflow ida nosize

This event indicates that a remote attacker has attempted to exploit a vulnerability in Microsoft IIS. An unchecked buffer in the Microsoft IIS Index Server ISAPI Extension could enable a remote intruder to gain SYSTEM access to the web server.[75]

MY.NET.5.96 and MY.NET.153.159 should be examined. If the indexing service is not required, unmap the extensions for it, and ensure the latest Microsoft security patch has been applied for IIS.[76]

[74] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids544&view=event
[75] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids552&view=event
[76] http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms02-018.asp

EXPLOIT x86 setgid 0

In theory, this signature is triggered by an exploit attempt where the attacker sent the setgid(0) system call for the x86 platform. It is the most effective when monitoring protocols that usually consist of plaintext printable ASCII to catch remote x86 exploits.[77] In practice, because the pattern matched is so short, it is frequently triggered by a wide variety of non-suspect activity.

My reference Snort ruleset alerts on "|b0b5 cd80|" on any port except 80 or 8080 (web traffic). Your alerts indicate a mix of web and Kazaa (filesharing) traffic in addition to some unknowns. The "unknowns", MY.NET.150.143, MY.NET.153.191, MY.NET.152.164, and MY.NET.153.194 should be examined.

SCAN FIN

This is a stealth portscan where TCP packets are sent having only the FIN flag set. Nmap[78] is the most common way of doing it, using the syntax "nmap –sF [target]"

All of the sources are external, and each had only 1 packet trigger this alert. Scans such as this are generally part of the background noise on the internet, and should cause concern only when activity indicates a pattern of recognizance.

Incomplete Packet Fragments Discarded

This alert is frequently seen when using the older "frag" preprocessor with Snort. It has been fixed under the "frag2" preprocessor included in later releases. The alerts themselves happen so infrequently (only 4 times in 5 days) and are not a major concern.

WEB-IIS encoding access

Similar to the Unicode trick (discussed earlier in this report), this attack attempts to use hex encoding circumvent access control. IIS allows for invalid hex sequences. Example: %1u%1u translates to ".."[79] [80]

MY.NET.5.96 and MY.NET.153.159 need the patch discussed in MS99-061[81] at the very minimum. A cumulative patch is available as MS02-018[82] that fixes this and several newer flaws.

[77] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids284&view=event
[78] http://www.insecure.org/nmap
[79] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids200&view=event
[80] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0024
[81] http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms99-061.asp
[82] http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms02-018.asp

RFB – Possible WinVNC - 010708 -1

VNC (Virtual Network Computing) is remote Keyboard/Video/Mouse software for Windows platforms.[83] It is frequently used (legitimately) for remote administration, and although protected by passwords, it is not particularly secure.

It appears that MY.NET.207.182 may be running this service, and MY.NET.152.44, MY.NET.152.169 and MY.NET.175 are acting as clients (to both internal and external hosts).

MY.NET.207.182 has no other alerts.

If VNC is legitimately used within your network, consider switching to a more secure product such as PCanywhere. If not already in place, an ingress filter for TCP/5900 would be wise to prevent external clients from connecting to internal hosts.

TFTP – External UDP connection to internal tftp server

TFTP (Trivial File Transfer Protocol) is used in diskless workstations and by some network devices to transfer files. Unlike FTP, it is connectionless and uses a very limited command set. It is also used by some worms and Trojans to download the code after a bootstrap is executed. Nimda is an excellent example of this[84]

MY.NET.153.45 and MY.NET.153.46 should be examined to determine if they are legitimate TFTP servers. A ingress filter at the perimeter for TCP/69 and UDP/69 will stop external clients from connecting to hosts within MY.NET.0.0/16.

INFO Outbound GNUTella Connect accept
INFO Inbound GNUTella Connect accept

A number of similar alerts involving the GNUTella network are discussed earlier in this report. These particular signatures indicates a successful connection.

A list of internal hosts using this software is available in this report under the heading **INFO Inbound GNUTella Connect request**

Blocking IP ports 6346 and 6347 at the perimeter will effectively stop internal clients from using any software which utilizes the GNUTella network.

[83] http://www.uk.research.att.com/vnc/winvnc.html
[84] http://www.cert.org/advisories/CA-2001-26.html

MISC PCAnywhere Startup
MISC Invalid PCAnywhere Login

PCAnywhere is frequently used (legitimately) package to enable remote control of a Windows server or workstation. If implemented correctly, it is also fairly secure as it utilizes strong encryption and authentication methods.

MY.NET.5.141 is running PCAnywere and listening for connections. External clients are apparently permitted to connect as 208.228.181.250 (unregistered host on UUnet) keeps trying and subsequently getting the password wrong.

An ingress filter on TCP/5632 and UDP/5632 is in order.

MISC source port 53 to <1024

This event indicates that an attacker is making a connection to a privileged port using the source port 53 (dns). This should not normally occur. Old or misconfigured packet filters may allow the connection if they allow all dns traffic.[85]

Strangely, both attempts which triggered this alert have a destination port of 0, which is also invalid. 63.250.205.41 and 63.146.181.137 are the external sources, and also have RedWorm activity.

MY.NET.153.46 and MY.NET.88.155 were the internal targets and should be investigated.

WEB-MISC webdav search access

This event indicates that a remote user has attempted to use the SEARCH directive to retreive a list of directories on the web server. This may allow an attacker to gain knowledge about the web server that could be useful in an attack.[86]

MY.NET.5.96 was the target in both detected attempts, and as has been previously identified as a default installation of IIS with numerous problems. It should be examined immediately.

Probable NMAP fingerprint attempt

Using the syntax "nmap –O [target]" will result in crafted packets in an attempt to match the responses to a list of known behaviors in an attempt to determine the remote OS type and version.

[85] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids7&view=event
[86] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids474&view=event

In my reference Snort ruleset, this is triggered by TCP packets with the SYN FIN PSH and URG flags set. This activity is definitely suspect, and MY.NET.253.10 should be examined.

WEB-IIS asp-dot attempt

Triggered by a trailing period after a request for an ASP page (eg: ".asp."). The result on an unpatched server is revealing the source code on the requested page.

This was fixed way back in sp3. Ensure that MY.NET.5.95 is up-to-date on security patches.

WEB-MISC whisker head

Alerts to "HEAD/./" in a request to a webserver. One of many methods proscribed by Rain Forest Puppy in his "Anti-IDS" paper.[87]

12.91.163.139 (AT&T dialup user in Washington DC) tried it twice against MY.NET.5.96. but triggered no other alerts.

Suspicious host traffic

Without access to the Snort ruleset in use, I am unable to determine what this alert attempts to log. Examination of the logs yields nothing which is overtly suspicious.

This alert was triggered twice with unique external sources and MY.NET.5.44 as the target.

WEB-CGI formmail access

Matt Wright's Formmail CGI script is terribly insecure. It is frequently used to turn webservers into SPAM relays, in addition to suffering internally from a number of input buffer overflows.[88] [89]

This script should be removed from MY.NET.5.95 and MY.NET.150.139 and a different means found to accomplish this functionality (if required).

[87] http://www.wiretrip.net/rfp/pages/whitepapers/whiskerids.html
[88] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids226&view=event
[89] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0172

TELNET access

Telnet is a widely used protocol for communicate in a shell with a UNIX host. It is not very secure as it transmits data unencrypted. A much better alternative is SSH.[90]

In both instances, this alert was triggered by MY.NET.5.79 connecting to external hosts, and is not suspicious.

TFTP – Ixternal UDP connection to external tftp server

TFTP (Trivial File Transfer Protocol) is used in diskless workstations and by some network devices to transfer files. Unlike FTP, it is connectionless and uses a very limited command set. It is also used by some worms and Trojans to download the code after a bootstrap is executed. Again, Nimda is an excellent example of this[91]

A egress filter for UDP/69 at the perimeter will prevent these connections.

Interestingly, one event between 64.124.157.10 and MY.NET.153.45 is suspect because privileged ports (<1024) are used for both source and destination. Something is probably amiss with MY.NET.153.45, which triggered 11 other alerts that are also suspicious (RedWorm, NTP buffer overflows, Nmap activity, etc).

WEB-MISC ICQ Webfront HTTP DOS

The ICQ webserver has a bug allowing attackers to read files outside the user's personal directory.[92] In my reference Snort ruleset, an attacker issues a string of period characters following the request (eg: "GET /page.html…….")

63.16.114.130 (unregistered host at UUnet) tried this against MY.NET.5.96.

WEB-CGI redirect access

ColdFusion ClusterCATS appends stale query string arguments to a URL during HTML redirection, which may provide sensitive information to the redirected site.[93]

152.163.188.37 (unregistered host at UUnet) tried this against MY.NET.150.83.

Read the Macromedia security bullitin regarding this issue[94] and apply the appropriate patch to MY.NET.150.83

[90] http://www.openssh.org
[91] http://www.cert.org/advisories/CA-2001-26.html
[92] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0474
[93] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0382

IDS475/web-iis_web_webday_propfind

This event indicates that a remote user has attempted to use the webdav PROFIND directive to retrieve a directory listing on the web server. This may allow an attacker to gain knowledge about the web server that could be useful in an attack.[95] [96]

If WebDAV is not required on MY.NET.5.96, locate the <ifDefine DAV> section in httpd.conf and change it to "off" and restart apache.[97]

ICMP Router Selection (Undefined Code!)

Normally, ICMP router selection packets go to the "all routers" multicast address of 224.0.0.2. I was unable to find this rule in my reference ruleset, but according to RFC1256[98], a type 10 ICMP packet (router selection) does not have any codes defined other than 0 (undefined).

I assume that a packet was observed with options set for type 10 and a code of something other than 0, a condition which would be invalid. This packet was observed between MY.NET.11.7 and MY.NET.152.15. Analysis of the complete packet log will be required to confirm exactly what the intent and/or problem is.

x86 NOOP – Unicode BUFFER OVERFLOW ATTACK

Like previous alerts of similar name, this alert indicates and attempt to overflow a daemon with a long sting of "0x90" characters – a trick named the "NOOP sled"[99].

In my experience, some web active content (eg: Macromedia) can trigger this alert. Supporting this theory is that in every instance, the source port was 80 and the destination port was empherical – indicating it was triggered by the reverse part of an established web session. That would not be the case in the event of an attack attempt (source would be empherical and destination would be 80).

More current versions of the Snort ruleset specifically exempt TCP/80 as a source port on shellcode alerts such as this.

---

[94] http://www.macromedia.com/v1/handlers/index.cfm?ID=15697&Method=Full
[95] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids475&view=event
[96] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0869
[97] http://online.securityfocus.com/bid/1656/solution/
[98] http://www.iana.org/assignments/icmp-parameters
[99] http://www.giac.org/practical/David_Oborn_GCIA.html#detect4

TCP SMTP Source Port traffic

Reserved ports (<1024) should not be used as source ports in an outgoing (initiating) connection. SMTP (TCP/25) is no exception.

209.242.15.114 sent a packet to MY.NET.152.141 meeting this criteria. Also odd is that some service is apparently running on port 799 at MY.NET.152.141 that accepted this connection. An investigation of MY.NET.152.141 is in order.

Table 4 : Scan Data : "Top Talkers" – Source IP and Port

IP Address			IP Ports (TCP/UDP/ICMP)			
IP Source	Scans	DstIP	Port	Scans	SrcIP	DstIP
MY.NET.60.43	462096	173	7000	490776	297	201
MY.NET.150.143	283592	5788	123	360884	77	159
MY.NET.6.45	196947	156	7001	313846	284	237
MY.NET.6.48	181565	140	0	203675	315	203
MY.NET.6.49	179920	136	28800	125122	17	1685
MY.NET.6.52	168898	141	1347	89310	181	322
MY.NET.6.50	136484	136	514	78186	70	62
MY.NET.11.8	88843	59	137	78109	180	196
MY.NET.6.53	83955	142	1057	75609	205	747
MY.NET.6.60	72094	146	2196	49995	125	813
MY.NET.150.113	51611	1698	88	46041	21	111
MY.NET.6.51	46173	91	6970	39605	78	93
MY.NET.150.246	34600	1960	778	17582	29	73
MY.NET.11.6	24324	59	2350	15480	113	135
MY.NET.60.11	19476	70	1076	15187	183	589

Table 5 : Scan Data : "Top Talkers" – Destination IP and Port

IP Address			IP Ports (TCP/UDP/ICMP)			
IP Destination	Scans	SrcIP	Port	Scans	SrcIP	DstIP
MY.NET.1.3	119115	441	7001	490885	297	199
MY.NET.1.7	89703	172	80	402264	469	20361
MY.NET.6.45	83781	155	7000	279424	290	422
MY.NET.1.4	79953	325	53	199344	449	118
MY.NET.60.43	58884	173	0	166723	311	203
MY.NET.11.6	43507	59	4665	145236	32	2422
MY.NET.6.53	36624	142	28800	101622	18	1614
MY.NET.6.60	34833	146	514	89826	194	69
MY.NET.153.209	34465	26	1346	89346	26	130
MY.NET.11.7	33655	59	137	75404	165	192
MY.NET.153.207	27135	25	4662	37083	27	2591
MY.NET.153.202	26267	24	7003	36263	185	58
MY.NET.88.148	25130	21	123	27181	194	107
MY.NET.153.148	24555	24	88	24690	130	56
MY.NET.153.140	24409	23	139	20924	92	102

Discussion of Scan Data

The most popular destination ports searched for on MY.NET.0.0/16 are :

7001 Probably triggered by Everquest, a multiplayer online role-playing game.[100]
80 IANA registered port for HTTP servers.[101]
7000 IANA registered port for AFS3 file server, also used by some trojans[102]
53 IANA registered port for Domain Name Server (DNS) daemon[103]
0 ICMP traffic – "ping" activity
4665 Used by eDonkey 2000 – Napster-like filesharing program[104 105]
28800 Used by some Microsoft multiplayer games[106]
514 IANA registered port for Syslog (UDP) or Shell (TCP)[107]
1346 Unknown?
137 Windows Networking (NetBIOS Name Service)[108]
4662 Used by eDonkey 2000 – An "adult filesharing and search engine"
7003 IANA registered port for AFS3 file server, also Everquest MORPG[109]
123 IANA registered port for Network Time Protocol (NTP)[110]
88 IANA registered port for Kerberos Daemon[111]
139 Windows Networking – (NetBIOS Session Service)[112]

None of these ports should be permitted into internal networks through the perimeter router. Public services such as HTTP (TCP/80) and DNS (UDP/53) should be in a DMZ segment logically separated from the internal network by firewall.

All 15 of the "top talkers" were located within MY.NET.0.0/16, but three hosts in particular seem to be doing an extremely high amount of scan activity in terms of the number of hosts. Inspect MY.NET.150.143, MY.NET.150.113, and MY.NET.150.246.

[100] http://www.portsdb.org/bin/portsdb.cgi?portnumber=7001
[101] http://www.portsdb.org/bin/portsdb.cgi?portnumber=80
[102] http://www.portsdb.org/bin/portsdb.cgi?portnumber=7000
[103] http://www.portsdb.org/bin/portsdb.cgi?portnumber=53
[104] http://www.dshield.org/port_report.php?port=4665
[105] http://www.edonkey2000.com/faq.html#port
[106] http://support.microsoft.com/support/kb/articles/Q309/1/28.asp
[107] http://www.portsdb.org/bin/portsdb.cgi?portnumber=514
[108] http://www.portsdb.org/bin/portsdb.cgi?portnumber=137
[109] http://www.portsdb.org/bin/portsdb.cgi?portnumber=7003
[110] http://www.portsdb.org/bin/portsdb.cgi?portnumber=123
[111] http://www.portsdb.org/bin/portsdb.cgi?portnumber=88
[112] http://www.portsdb.org/bin/portsdb.cgi?portnumber=139

Discussion of Out of Spec (OOS) Data

The majority of OOS data submitted resulted from P2P (filesharing) traffic using the Fasttrak (Kazaa, Morpheus), eDonkey2000 and GNUTella networks. It has been my experience that these protocols are notoriously misbehaved in their attempts to keep ahead of administrators determined to block them.

The remainder is somewhat suspicious :

192.115.135.8 (line135-8.adsl.actcom.co.il) had several with source ports below 1024. Based on other alerts triggered by this host, attempts were being made to fingerprint (Queso) hosts on MY.NET to determine what remote OS type and version was running.

24.141.97.182 (d141-97-182.home.cgocable.net) was doing Null scans with nmap.

211.37.21.179 (unknown host) sent a variety of strange traffic to MY.NET.150.46 using reserved bits in the TCP header. Investigation of 211.37.21.179 returns nothing, but MY.NET.150.46 should be investigated because it's odd that a host would have MSN messenger traffic and ICMP router selection messages.

217.235.147.155 (pD9EB939B.dip.t-dialin.net) sent traffic employing reserved bits in the TCP header to MY.NET.153.160 and triggered alerts for Queso fingerprinting.

142.51.44.123 (unknown host) performed a variety of scans against MY.NET.88.162 that had an assortment of problems. MY.NET.88.162 triggered several exploit signatures, and had significant amounts of traffic to watched network IL-ISDNNET-990517

202.153.244.62 (unknown host) was very interested in the webserver at MY.NET.150.83 – a server which should itself be examined for default configurations and web vulnerabilities.

References

AT&T Research Group
http://www.uk.research.att.com/vnc/winvnc.html

Carnegie Melon Software Engineering Institute, CERT Coordination Center
http://www.cert.org/advisories/CA-1996-11.html
http://www.cert.org/advisories/CA-2001-26.html
http://www.cert.org/advisories/CA-2001-07.html
http://www.cert.org/advisories/CA-2001-26.html
http://www.cert.org/incident_notes/IN-2001-01.html
http://www.cert.org/incident_notes/IN-2002-01.html
http://www.kb.cert.org/vuls/id/29823
http://www.kb.cert.org/vuls/id/34043
http://www.kb.cert.org/vuls/id/382365

Cult of the Dead Cow
http://www.cultdeadcow.com/

Fydor
http://www.insecure.org/nmap
http://www.insecure.org/nmap/nmap_documentation.html
http://www.insecure.org/sploits/Microsoft.frontpage.insecurities.html

Internet Assigned Numbers Authority
http://www.iana.org/assignments/icmp-parameters
http://www.iana.org/assignments/port-numbers

Internet Ports Database
http://www.portsdb.org/bin/portsdb.cgi?portnumber=53
http://www.portsdb.org/bin/portsdb.cgi?portnumber=80
http://www.portsdb.org/bin/portsdb.cgi?portnumber=88
http://www.portsdb.org/bin/portsdb.cgi?portnumber=123
http://www.portsdb.org/bin/portsdb.cgi?portnumber=137
http://www.portsdb.org/bin/portsdb.cgi?portnumber=139
http://www.portsdb.org/bin/portsdb.cgi?portnumber=514
http://www.portsdb.org/bin/portsdb.cgi?portnumber=1752
http://www.portsdb.org/bin/portsdb.cgi?portnumber=1753
http://www.portsdb.org/bin/portsdb.cgi?portnumber=7000
http://www.portsdb.org/bin/portsdb.cgi?portnumber=7001
http://www.portsdb.org/bin/portsdb.cgi?portnumber=7003

Internet Request for Comments (RFC) Repository
http://rfc.net/rfc1459.html

McAfee Corporation
http://vil.mcafee.com/dispVirus.asp?virus_k=99333
http://www.pgp.com/research/covert/advisories/048.asp

Macromedia Corporation
http://www.macromedia.com/v1/handlers/index.cfm?ID=15697&Method=Full

MetaMachine, Inc.
http://www.edonkey2000.com/faq.html#port

Microsoft Corporation
http://support.microsoft.com/default.aspx?scid=kb;en-us;Q256888
http://support.microsoft.com/support/kb/articles/Q309/1/28.asp
http://www.microsoft.com/technet/security/bulletin/ms99-061.asp
http://www.microsoft.com/technet/security/bulletin/ms00-058.asp
http://www.microsoft.com/technet/security/bulletin/ms00-078.asp
http://www.microsoft.com/technet/security/bulletin/ms02-018.asp
http://msdn.microsoft.com/library/en-us/dnservext/html/fp2ksecuritywp.asp

Mitre Corporation, Common Vulnerabilities and Exposures
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0236
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0474
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0509
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0632
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0869
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0172
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0024
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0149
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0382
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0778
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0341

Rain Forest Puppy
http://www.wiretrip.net/rfp/pages/whitepapers/whiskerids.html

Redhat Corporation
http://www.redhat.com/support/alerts/Adore_worm.html

Sanfilippo, Salvatore
http://www.hping.org/manpage.html

Securityfocus
http://online.securityfocus.com/bid/1578/discussion/
http://online.securityfocus.com/bid/1656/solution/
http://www.securityfocus.com/bid/2540

SSH Communications Security
http://www.openssh.org

Squid Web Proxy Cache
http://www.squid-cache.org/

Symantec Corporation
http://www.symantec.com/press/security/n990923_ns.html

System Administration and Network Security (SANS) Institute
http://www.dshield.org/port_report.php?port=4665
http://www.giac.org/practical/David_Oborn_GCIA.html#detect4
http://www.sans.org/newlook/resources/IDFAQ/port_137.htm
http://www.sans.org/y2k/adore.htm

Whitehats Network Security Resource
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids7&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids28&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids118&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids152&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids174&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids200&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids226&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids227&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids244&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids247&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids284&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids291&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids305&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids311&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids429&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids436&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids460&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids461&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids474&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids475&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids492&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids521&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids544&view=event
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids552&view=event

Appendix A – Perl script used in data import

```
#!/usr/bin/perl
#
# This program was written to take syslog-style SNORT output
# and put it into a MySQL database so it can be manipulated
# using ACID (www.cert.org/kb/acid).
#
# Dependencies: Perl DBI module, available from CPAN (www.cpan.org)
#
# Giving Credit to those who helped :
#       Sean Brown : "snort2pl" script provided the starting point
#       Marty Roesch : "spo_database.c" from Snort-1.8.6
#       Kevin Likes : Co-Worker who answered my dumb questions
#       Brian Zust : Co-Worker who answered my dumb questions
#
# Copyright (C) 2002, Michael Holstein
#
# This program is free software; you can redistribute it and/or
# modify it under the terms of the GNU General Public License
# as published by the Free Software Foundation; either version 2
# of the License, or (at your option) any later version.
#
# $Author: Michael Holstein, <moholstein@hotmail.com> $
# $Date: 2002/05/10 19:47:00 $
#
open INFILE, "</import/alert.tcp>";

$SID=1;                 #SID number the import will use
$database=giac;         #MySQL database name
$host=pluto;            #MySQL Server hostname
$username=giac;         #username for $database
$password=giac;         #password for $username on $database

#Here we go!
use DBI();
#DBI->trace(5, "trace.log"); #Uncomment this line to log DBI stuff
$dbh = DBI->connect("DBI:mysql:database=$database;host=$host",
        $username, $password, {'RaiseError' => 1})
        || die "Unable to connect to DB: $dbh->errstr\n";

#SUBROUTINES
sub getcid() {
        my $cid = $dbh->prepare("SELECT max(cid) from event where sid = $SID");
        $cid->execute() || die "Unable to execute query: $dbh->errstr\n";
        my $ref = $cid->fetchrow_arrayref;
        $maxcid = $$ref[0];
        return $maxcid;
}

#LINE PROCESSOR LOOP
while (chomp($line = <INFILE>))
        {($timestamp, $alert, $address) = split '\[\*\*\]', $line;
        ($month, $day, $time) = split /\/|\-/, $timestamp;
        ($hour, $minute, $second, $milisecond) = split /\:|\./, $time;
        $alert =~ s/^\s+//;     #cut leading space off alert
        $alert =~ s/\s+$//;     #cut trailing space off alert
        ($srcADR, $dstADR) = split /\->/, $address;
        ($srcIP, $srcPT) = split /\:/, $srcADR;
        ($dstIP, $dstPT) = split /\:/, $dstADR;
        ($sip1, $sip2, $sip3, $sip4) = split /\./, $srcIP;
        $srcIPdec = (($sip1*16777216)+($sip2*65536)+($sip3*256)+$sip4);
        ($dip1, $dip2, $dip3, $dip4) = split /\./, $dstIP;
        $dstIPdec = (($dip1*16777216)+($dip2*65536)+($dip3*256)+$dip4);

#FUNCTION LOOPS
$newcid = &getcid() + 1;
$sqltimestamp = "2002-$month-$day $hour:$minute:$second";
```

```
$maxsig = $dbh->prepare("select max(sig_id) from signature");
        $maxsig->execute();
        $maxsig = $maxsig->fetchrow_array() + 1;
$signum = $dbh->prepare("select sig_id from signature where sig_name = '$alert'");
        $signum->execute();
        $signum = $signum->fetchrow_array();
        if ($signum) {
                $dbh->do("insert into event (sid, cid, signature, timestamp) values
('$SID', '$newcid', '$signum', '$sql
timestamp')") || die "Problem with EVENT import\n";
                $dbh->do("INSERT INTO tcphdr (sid, cid, tcp_sport, tcp_dport, tcp_seq,
tcp_ack, tcp_off, tcp_res,tcp_fla
gs, tcp_win, tcp_csum, tcp_urp) VALUES ('$SID', '$newcid', '$srcPT', '$dstPT',
'0','0','0','0','0','0','0','0')") || die
 "Problem with TCPHDR import\n";
                $dbh->do("INSERT INTO iphdr (sid, cid, ip_src, ip_dst, ip_ver, ip_hlen,
ip_tos, ip_len, ip_id, ip_flags,
 ip_off, ip_ttl, ip_proto, ip_csum) VALUES ('$SID', '$newcid', '$srcIPdec', '$dstIPdec',
'4', '40', '5', '0', '40', '0',
 '0', '64', '6', '0')") || die "Problem with IPHDR import\n";
                printf STDOUT "imported sid=$SID cid=$newcid sig_name=$alert
timestamp=$sqltimestamp\n";
                }

        else {
                $dbh->do("insert into signature (sig_id, sig_name, sig_class_id,
sig_priority, sig_rev, sig_sid) values
('$maxsig', '$alert', '0', '0', '0', '0')") || die "Problem adding NEW SIG\n";
                $dbh->do("insert into event (sid, cid, signature, timestamp) values
('$SID', '$newcid', '$maxsig', '$sql
timestamp')") || die "Problem with EVENT import\n";
                $dbh->do("INSERT INTO tcphdr (sid, cid, tcp_sport, tcp_dport, tcp_seq,
tcp_ack, tcp_off, tcp_res, tcp_fl
ags, tcp_win, tcp_csum, tcp_urp) VALUES ('$SID', '$newcid', '$srcPT', '$dstPT',
'0','0','0','0','0','0','0','0')") || di
e "Problem with TCPHDR import\n";
                $dbh->do("INSERT INTO iphdr (sid, cid, ip_src, ip_dst, ip_ver, ip_hlen,
ip_tos, ip_len, ip_id, ip_flags,
 ip_off, ip_ttl, ip_proto, ip_csum) VALUES ('$SID', '$newcid', '$srcIPdec', '$dstIPdec',
'4', '40', '5', '0', '40', '0',
 '0', '64', '6', '0')") || die "Problem with IPHDR import\n";
                printf STDOUT "imported sid=$SID cid=$newcid sig_name=$alert
timestamp=$sqltimestamp\n";
        }
}
```