Global Information Assurance Certification Paper

John Sylar
Intrusion Detection In Depth
GCIA Practical Assignment Version 2.9
Baltimore SANS, May, 2001

Contents:

Detects
RPC Scan
```
messages.18052630:May 26 18:06:23 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 213.141.72.163/111 to a.b.c.21/111
flags SYN
messages.18052630:May 26 18:06:23 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 213.141.72.163/111 to a.b.c.22/111
flags SYN
messages.18052630:May 26 18:06:23 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 213.141.72.163/111 to a.b.c.23/111
flags SYN
messages.18052630:May 26 18:06:23 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 213.141.72.163/111 to a.b.c.30/111
flags SYN
messages.18052630:May 26 18:06:23 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 213.141.72.163/111 to a.b.c.31/111
flags SYN
```

Source of Trace: Company network.

Detect was generated by: PIX firewall logging to syslog.
The field format is:
`Syslog timestamp: Firewall timestamp Host PIX message code:`
`Event description Source address/port Destination address/port Flags (if TCP)`

Probability the Source Address was spoofed: The source address was probably not spoofed. This guy was running up and down my subnet looking for a hole, so I'm pretty sure he wanted to see a response. .

Description of Attack: This is a reconnaissance scan for RPC services. The attacker is looking for something to do later, possibly the statd buffer overflow attack (Ref: CVE-1999-0018: "Buffer Overflow in statd allows root privileges") as that seems to be the RPC exploit du jour, but there are many others. The ArachNIDS database has a detail and summary of the scan here: http://www.whitehats.com/info/IDS428

Attack Mechanism: The scan attempts to connect to the portmapper daemon at TCP port 111. Portmapper keeps track of the port locations of RPC services on various flavors of Unix. CERT Advisory IN-99-04 (http://www.cert.org/incident_notes/IN-99-04.html) outlines a number of RPC services vulnerable to buffer overflow attacks across Unix platforms. If the port connection can be made and RPC services enumerated, the attacker can choose a number of ways to break in.

Correlation:
From: http://www.dshield.org

IP Address: 213.141.72.163
HostName: www.datakunskap.se
DShield Profile:

Country:	SE
Contact E-mail:	fredric@datakunskap.se
Total Records against IP:	56
Number of targets:	53
Date Range:	2001-05-23 to 2001-05-28

Date	Source	Source Port	Target Port	Protocol	Flags
2001-05-28	213.141.72.163	111	111	6	S
2001-05-27	213.141.72.163	111	111	6	S
2001-05-27	213.141.72.163	111	111	6	S
2001-05-27	213.141.72.163	111	111	6	S
2001-05-27	213.141.72.163	111	111	6	S
2001-05-27	213.141.72.163	111	111	6	S
2001-05-27	213.141.72.163	111	111	6	S
2001-05-27	213.141.72.163		111	0	
2001-05-27	213.141.72.163	111	111	6	S
2001-05-26	213.141.72.163	111	111	6	S
2001-05-26	213.141.72.163	111	111	6	S

Evidence of Active Targeting: None. The attacker is scanning. Correlations show this attacker has been busy probing other networks.

Severity:

Criticality	2	The attacker is scanning my whole subnet.
Lethality	2	This seems to be reconnaissance only.
System Countermeasures	5	Systems with RPC services are fully patched.
Network Countermeasures	5	The attack was blocked by the firewall.

Severity = (2+2)-(5+5) = -6

Test Question: See Appendix A

FTP Scan

```
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.47/21 flags
SYN
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.47/21 flags
RST
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.252/21
flags SYN
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.252/21
flags RST
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.50/21 flags
SYN
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.50/21 flags
RST
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.251/21
flags SYN
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.251/21
flags RST
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.43/21 flags
SYN
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.43/21 flags
RST
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.249/21
flags SYN
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.249/21
flags RST
```

Source of Trace: Company network.

Detect was generated by: PIX firewall logging to syslog.
The field format is:
`Syslog timestamp: Firewall timestamp Host PIX message code:`
`Event description Source address/port Destination address/port Flags (if TCP)`

Probability the Source Address was spoofed: The source address was probably not spoofed. This guy was running up and down my subnet looking to fingerprint my machines, so I'm pretty sure he wanted to see a response.

Description of Attack: This is a reconnaissance scan for FTP servers and/or OS fingerprints. The attacker might be trying to collect the connection banners from responsive hosts, as those banners usually describe the host and/or operating system.

Attack Mechanism: The attacker is scanning my subnet using reflexive source and destination ports – both are port 21 (FTP). The SYN and RST flags are alternately set during this scan. This seems to be an odd way of doing things, but in any case, the scan will open and then immediately close a connection to an FTP server, similar to a SYN-FIN scan. FTP banners are grabbed from those hosts that respond in order to read the datestamp. The datestamp determines the operating system of the machine. If the right type of machine is found, an attack can be launched against those machines with publicly available exploits. Right now, Redhat 6.2 or 7.0 servers are most vulnerable, but there are a host of other FTP vulnerabilities. See http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=FTP and www.cert.org for details.

Correlations:
From http://www.dshield.org

IP Address: 208.15.87.15
HostName: ns1.aladdinscomm.com
DShield Profile:

Country:	US
Contact E-mail:	NOC@SPRINT.NET
Total Records against IP:	2
Number of targets:	1
Date Range:	2001-05-26 to 2001-05-26

Date	Source	Source Port	Target Port	Protocol	Flags
2001-05-26	208.15.87.15	21	21	6	S
2001-05-26	208.15.87.15	21	21	6	

Evidence of Active Targeting: None. The attacker is scanning. Correlations show this attacker has been busy probing other networks.

Severity:

Criticality	2	The attacker is scanning my whole subnet.
Lethality	2	This seems to be reconnaissance only.
System Countermeasures	5	Systems are fully patched and without anonymous access.
Network Countermeasures	5	The attack was blocked by the firewall.

Severity = (2+2)-(5+5) = -6

Defensive Recommendation: The attack was successfully blocked by the firewall.

Test Question: See Appendix A

Unicode Attack

Source	SPort	Dest	DP	Time	Payload
202.206.240.11	51886	a.b.c.65	80	5/26/2001 13:30	..%c0%af. .*HTTP | GET /scripts/..%c0%af.. /winnt/system32/cmd.exe?/c+dir HTTP\0'
202.206.240.11	50439	a.b.c.173	80	5/26/2001 13:39	'..%c0%af.. *HTTP | GET /scripts/..%c0%af.. /winnt/system32/cmd.exe?/c+dir HTTP\0'
202.206.240.11	50491	a.b.c.173	80	5/26/2001 13:39	'..%c1%9c. .*HTTP | GET /scripts/..%c1%9c.. /winnt/system32/cmd.exe?/c+dir HTTP\0'
202.206.240.11	53605	a.b.c.198	80	5/26/2001 13:40	'..%c0%af.. *HTTP |GET /scripts/..%c0%af.. /winnt/system32/cmd.exe?/c+dir HTTP\0'
202.206.240.11	53656	a.b.c.198	80	5/26/2001 13:40	'..%c1%9c.. *HTTP |GET /scripts/..%c1%9c. ./winnt/system32/cmd.exe?/c+dir HTTP\0'

Source of Trace: Company network.

Detect was generated by: IBM NetRanger IDS.

Probability the Source Address was spoofed: The source address was definitely not spoofed. This guy was hitting my Windows web servers directly, so I know he wanted to see a response.

Description of Attack: This attack is aimed at Microsoft IIS 4.0 and 5.0 web servers. It's sometimes called the "Web Server Folder Traversal" vulnerability. If successful, the attacker can run commands or execute code against the local machine, as in this case, where he's trying to open a command prompt. Refer to CVE-2000-0884.

Attack Mechanism: The attack can be successful because of a canonicalization - or, file name equivalency – error, affecting CGI scripts and ISAPI extensions in Microsoft IIS web servers. If an URL requesting a CGI script or an executable were formatted a certain way, the operating system can be "fooled" into resolving the location of a file in a different folder than it actually is. In other words, the "GET" string documented above can cause the operating system to resolve the terminal application (in the system folder) as part of the web root. The string is usually passed from a command line via telnet connection.
See http://www.kb.cert.org/vuls/id/111677 and
http://www.microsoft.com/technet/security/bulletin/MS00-078.asp for details.

Correlations:
From http://www.dshield.org

IP Address: 202.206.240.11
HostName: mail.ysu.edu.cn

DShield Profile:

Country:	CN
Contact E-mail:	root@pku.edu.cn
Total Records against IP:	4
Number of targets:	2
Date Range:	2001-06-20 to 2001-07-09

Whois:

% Rights restricted by copyright. See
http://www.apnic.net/db/dbcopyright.html
% (whois5.apnic.net)

inetnum: 202.206.240.0 - 202.206.255.255
netname: YSU-CN
descr: ~{Q`I=4sQ'~}
descr: Yan Shan University
descr: Qin HuangDao,066004
country: CN
admin-c: XK1-CN
tech-c: XK1-CN
notify: address-allocation-staff@net.edu.cn
changed: xing@ocean.net.edu.cn 960615
source: APNIC

person: Xiangdong Kong
address: ~{Q`I=4sQ'T:3$0l9+JR~}
address: The Office of Yan Shan University
phone: +86-0335-8057099
fax-no: +86-0335-8051148
e-mail: root@pku.edu.cn
nic-hdl: XK1-CN
notify: address-allocation-staff@net.edu.cn
mnt-by: MAINT-NULL
changed: szhu@net.edu.cn 19960429
source: APNIC

Date	Source	Source Port	Target Port	Protocol	Flags
2001-07-09	202.206.240.11	39198	80	6	S
2001-06-20	202.206.240.11	54596	80	6	
2001-06-20	202.206.240.11	54596	80	6	S
2001-06-20	202.206.240.11	54596	80	6	S

Evidence of Active Targeting: Yes. The attacker had done enough reconnaissance to know which of my servers run Windows. Correlations show this attacker has been busy trying his luck elsewhere.

Severity:

Criticality	5	The attacker knows which machines are which.
Lethality	5	My systems could be completely compromised.
System Countermeasures	5	Systems are fully patched and monitored.
Network Countermeasures	2	Source address can be filtered at the border router.

Severity = (5+5)-(5+2) = 3

Defensive Recommendation: Affected systems are fully patched and monitored.

Test Question: See Appendix A

Authd/Identd Probe

```
May 27 12:08:10 pixpublic-active %PIX-2-106001: Inbound TCP
connection denied from 204.235.225.20/54037 to a.b.c.120/113 flags SYN
on interface outside
May 27 12:08:10 pixpublic-active %PIX-2-106001: Inbound TCP
connection denied from 204.235.225.20/54039 to a.b.c.120/113 flags SYN
on interface outside
May 27 12:08:10 pixpublic-active %PIX-2-106001: Inbound TCP
connection denied from 204.235.225.20/54041 to a.b.c.120/113 flags SYN
on interface outside
May 27 12:08:10 pixpublic-active %PIX-2-106001: Inbound TCP
connection denied from 204.235.225.20/54043 to a.b.c.120/113 flags SYN
on interface outside
```

Source of Trace: Company network.

Detect was generated by: PIX firewall logging to syslog.
The field format is:
`Syslog timestamp: Firewall timestamp Host PIX message code:`
`Event description Source address/port Destination address/port Flags (if TCP)`

Probability the Source Address was spoofed: The source address was not spoofed. The server that issued the request wanted a response.

Description of Attack: This isn't an attack at all, but a legitimate mail server running SendMail responding to traffic from my network. That might not always be the case, though. Authd is

susceptible to buffer overflows and Trojan attacks. Refer to CVE-1999-204, CVE-1999-746, CVE-2001-006, and http://www.simovits.com/nyheter9902.html for details.

Attack Mechanism: Port 113 (TCP) is reserved for Authd – Unix; and Identd – Windows, its purpose being authentication of users connecting to a service. Because it reveals more information than people are comfortable with these days, maybe only a handful of systems still use Authd. However, one holdout is SendMail. As demonstrated above in the log entries, SendMail will try to establish the identity of a user initiating a connection to send mail. While SendMail doesn't require successful authentication to function, it will wait up to 5 minutes for a reply. Such a wait can slow down mail delivery considerably, not to mention clogging up the network with Ident requests (SendMail is widely deployed).
Some firewalls, like GnatBox (www.gta.com) will send fake replies to Ident queries. Others, like the PIX firewall (www.cisco.com), can be configured to send resets in response. Either method is acceptable and will keep SendMail servers happy.

Correlations:
There is no direct correlation to this host at www.incidents.org, www.dshield.org, or www.mynetwatchman.com. Like me, this signature has tripped up a few folks though:
http://www.incidents.org/archives/y2k/021301-1000.htm

Server used for this query: [whois.arin.net]

U S WEST - Interact Internet Services

600 Stinson Blvd NE Minneapolis, MN 55413 us

Netname: USW-INTERACT97

Netblock: 209.180.0.0 - 209.181.255.255

Maintainer: USW

Feb 9 22:09:57 hostmf /kernel: Connection attempt to TCP

a.b.f.167:113 from 209.180.6.226:4559

http://www.incidents.org/archives/y2k/012000.htm
(San Jose checks in...above.net What exactly did they expect me to authenticate?)

98924650 ms IP Open request denied from 209.133.83.18

98924650 ms IP Requested destination port 113

98927180 ms IP Open request denied from 209.133.83.18

98927180 ms IP Requested destination port 113

98933175 ms IP Open request denied from 209.133.83.18

98933175 ms IP Requested destination port 113

98945110 ms IP Open request denied from 209.133.83.18

98945110 ms IP Requested destination port 113

Sometimes, the service is scanned for:
http://www.incidents.org/archives/y2k/030101-1500.htm
Started getting the attached scan last evening and it is still continuing as of this writing. The excerpt attached is from only one of my DMZ's but it is happening on all three. The host and it's nearest upstream neighbor are "verio.net" of Englewood, CO. I reported a source port 23 scan a month or so ago and this appears to be of the same vein. The use of 113 (auth) makes a little more sense than telnet. I gotta believe there are quite a few responsible persons that are unaware that their mail servers play that game unnoticed

and like a professional. It might help the educational process to get people aware of what might be happening unbeknownst to them and the possible serious repercussions.

2/28/01,8:16:56 PM,OUR NET,2923: list 130 denied tcp 206.239.85.84(113)
-> xxx.xxx.100.7(5761), 1 packet,

2/28/01,8:30:31 PM,OUR NET,2925: list 130 denied tcp 206.239.85.84(113)
-> xxx.xxx.100.80(4389), 1 packet,

2/28/01,8:35:12 PM,OUR NET,2926: list 130 denied tcp 206.239.85.84(113)
-> xxx.xxx.100.33(38171), 1 packet,

2/28/01,8:42:06 PM,OUR NET,2927: list 130 denied tcp 206.239.85.84(113)
-> xxx.xxx.100.116(56545), 1 packet,

2/28/01,9:04:05 PM,OUR NET,2928: list 130 denied tcp 206.239.85.84(113)
-> xxx.xxx.100.26(34363), 1 packet,

http://www.incidents.org/archives/y2k/030201-0900.htm
Matt, Yep I am seeing the same thing. Started yesterday at 18:20 EST. I manage a few different networks on different ISP's so I've seen this on 12.27.x.y, 64.55.e.f, 64.241.c.d and 64.242.a.b networks.

Date	Time	Origin	Source IP	Dest port	Dest IP	Src Port	length
28-Feb-01	18:20:02	12.27.x.y	206.239.85.84	24732	Internal_host	113	40
28-Feb-01	18:20:12	64.242.a.b	206.239.85.84	431	Internal_host	113	40
28-Feb-01	19:25:46	64.241.c.d	206.239.85.84	53195	Internal_host	113	40
28-Feb-01	20:13:34	12.27.x.y	206.239.85.84	10092	Internal_host	113	40
28-Feb-01	20:41:11	12.27.x.y	206.239.85.84	38381	Internal_host	113	40
28-Feb-01	22:20:07	12.27.x.y	206.239.85.84	25208	Internal_host	113	40
28-Feb-01	22:36:38	12.27.x.y	206.239.85.84	11591	Internal_host	113	40
28-Feb-01	22:43:29	64.55.e.f	206.239.85.84	26411	Internal_host	113	40
1-Mar-01	0:23:34	12.27.x.y	206.239.85.84	11913	Internal_host	113	40
1-Mar-01	8:59:15	12.27.x.y	206.239.85.84	35502	Internal_host	113	40
1-Mar-01	12:29:10	64.55.e.f	206.239.85.84	457	Internal_host	113	40

Evidence of Active Targeting: Well, yes and no. Yes in that the query is directed towards a specific machine. No because my local machine is not really a target in the strictest sense of the word.

Severity:

Criticality	1	Normal machine traffic.
Lethality	2	Authd needs be monitored, as with all exploitable services.
System Countermeasures	5	Authd is not available on Internet-connected systems.
Network Countermeasures	5	The query was blocked by the firewall.

Severity = (1+2)-(5+5) = -7

Defensive Recommendation: The “attack” was successfully blocked by the firewall.

Test Question: See Appendix A

BackOrifice/Schoolbus Scan

```
messages.19052855:May 28 12:53:05 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 216.61.164.172/54321 to
a.b.c.30/54321 flags SYN
messages.19052855:May 28 12:53:05 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 216.61.164.172/54321 to
a.b.c.31/54321 flags SYN
messages.19052855:May 28 12:53:05 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 216.61.164.172/54321 to
a.b.c.145/54321 flags SYN
messages.19052855:May 28 12:53:05 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 216.61.164.172/54321 to
a.b.c.32/54321 flags SYN
messages.19052855:May 28 12:53:05 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 216.61.164.172/54321 to
a.b.c.50/54321 flags SYN
```

Source of Trace: Company network.

Detect was generated by: PIX firewall logging to syslog.
The field format is:
`Syslog timestamp: Firewall timestamp Host PIX message code:`
`Event description Source address/port Destination address/port Flags (if TCP)`

Probability the Source Address was spoofed: The source address was probably not spoofed. This guy was running up and down my subnet looking for open Trojan ports, so I'm pretty sure he wanted to see a response.

Description of Attack: This is a reconnaissance scan for Back Orifice 2000 and/or Schoolbus servers. Refer to CAN-1999-0660, http://www.simovits.com/nyheter9902.html and http://www.cert.org/vul_notes/VN-98.07.backorifice.html.

Attack Mechanism: Quote the CDC: "Back Orifice is a remote administration tool allowing a user to control a computer over a TCP/IP connection using a simple console or GUI application…BO gives its user more control of the remote Windows machine than the person at the keyboard of the remote machine has." (http://www.cultdeadcow.com/tools/bo.html). Back Orifice is well developed, written in C++, and runs on all varieties of Windows machines. Schoolbus is pretty much the same type of application, although it isn't as well developed (it doesn't have the fancy GUI or encrypted communication) and its default listening port, 54321, can't be changed. Schoolbus runs only on Windows95/98.
Back Orifice and Schoolbus are known to spread via email attachments or some "trojanized" software. Once installed, the servers listen on ports 31337, 31338, 54320, and 54321. One mark of the success of Back Orifice is the continuous scanning by remote hosts looking for systems already compromised.

Correlations: There is no correlation to the source IP address at www.dshield.org, but Laurie Zirkle saw the same scan from the same source on the same day. So did Matt Fearnow. There are loads of similar scans in the www.incidents.org archives as well:
Date: Mon, 28 May 2001 10:52:02 -0400
From: Laurie Zirkle <lat@xxxxxxxxxx>
Subject: May 27, 2001 probes

Server used for this query: [whois.arin.net]

Creative Labs (NETBLK-CREATIVE84)

1523 Cimarron Plaza Stillwater, OK 74075 US

Netname: CREATIVE84

Netblock: 216.61.164.0 - 216.61.167.255

May 27 02:18:45 216.61.164.172:54321 -> a.b.c.20:54321 SYN ******S*

May 27 02:18:45 216.61.164.172:54321 -> a.b.c.22:54321 SYN ******S*

May 27 02:18:45 216.61.164.172:54321 -> a.b.c.27:54321 SYN ******S*

May 27 02:18:45 216.61.164.172:54321 -> a.b.c.33:54321 SYN ******S*

May 27 02:18:45 216.61.164.172:54321 -> a.b.c.50:54321 SYN ******S*

May 27 02:18:45 216.61.164.172:54321 -> a.b.c.51:54321 SYN ******S*

May 27 02:18:45 216.61.164.172:54321 -> a.b.c.59:54321 SYN ******S*

From: Matt Fearnow [mailto:matt@xxxxxxxxxxxx]

> Sent: Tuesday, May 29, 2001 1:56 PM

> To: Erickson Brent W KPWA; 'intrusions@xxxxxxxxxxxx'

> Subject: Re: Distilled Weekend Recap

>

> I too seen this scan on a network I monitor.

>

> May 28 12:41:41 ipmon[5875]: xl1 @0:19 b 216.61.164.172,54321 ->

> a.b.c.2,54321 PR tcp len 20 40 -S IN

> May 28 12:41:41 ipmon[5875]: xl1 @0:19 b 216.61.164.172,54321 ->

> a.b.c.1,54321 PR tcp len 20 40 -S IN

> May 28 12:41:56 ipmon[5875]: xl1 @0:19 b 216.61.164.172,54321 ->

> a.b.c.1,54321 PR tcp len 20 40 -R IN

Evidence of Active Targeting: None. The attacker is scanning. Correlations show this attacker has been busy probing other networks.

Severity:

Criticality	2	The attacker is scanning my whole subnet.
Lethality	2	This seems to be reconnaissance only.
System Countermeasures	5	Systems are fully patched and monitored.
Network Countermeasures	5	The attack was blocked by the firewall.

Severity = (2+2)-(5+5) = -6

Defensive Recommendation: The attack was successfully blocked by the firewall.

Test Question: See Appendix A

Evaluate an Attack or Present A Challenge
Parsing PIX Firewall Logs

Introduction
Fresh back from SANS Baltimore 2001 and Stephen Northcutt's Intrusion Detection course, I was newly paranoid and certain some compromised system lay around every corner of my network; some nefarious cracker was camped out in every cube. My outfit is new to intrusion detection, and doesn't have a lot invested in the technology. Me, as a matter of fact, and that's about it, with a sprawling network to cover. There are probably loads of new analysts in the same boat.
So I'm sitting in my manager's office giving him the run-down on what I learned. I'm telling him about SANS, the tools and techniques available to us and our attackers, where I think we need to place sensors, and describing all of those pre-packaged Intrusion Detection Systems so fancy I saw at the trade show there, when it dawns on me that he's looking at me like some kid looks at a new piece of software. He doesn't care if everyone in the world with a desktop system is out to get us. No. He wants Plug-n-Play; he wants me to start spitting out reports. So much for paranoia….

OK, so the sales pitch didn't exactly work. I didn't get a budget or a promise of new equipment, but I did come away with a challenge to make some sense of what we have. Well, what we have is bunch of PIX firewalls and a couple of gigs worth of logs….

Now, anyone who's ever worked with a PIX firewall knows that when it comes to finding out what the thing is actually doing, you're on your own. There isn't a piece of software or out-of-the-box system in the world – other than the ones produced by Cisco – that will manage them. Not to mention they generate gobs of logs. Something akin to 50 Megs a day from only one PIX sitting on one Internet connection; and we have a half dozen of them. Which brings me to the challenge: with only spare parts and no money, parse, and if possible, organize a load of PIX firewall logs. Get the log entries into a database and produce reports for management that are suitable for framing. In the section that follows, I'm going to demonstrate how to do just that with Kiwi, Perl, ACID, PHP, MySQL, and a little slight-of-hand….

Meeting the Challenge: What you need
I did this on a Windows2000 machine so you'll need one of those. This will also work on 98 or NT. If you don't have (or want) a Windows box, you should still get enough information here to run your platform of choice. Whatever machine you wind up with, it needs to be fairly beefy. Like I said before, PIX firewalls generate a lot of logs in a short time, so you'll want plenty of drive space and some processing power. For my part, I walked up to the lab and snagged a box from the engineers (caveat: for your part, if you follow my lead, make sure you pick a box that won't be missed). I chose for myself an older model Compaq – these days, the equivalents are called DL380's - with a 450M processor, 512 RAM, three 18G SCSI's, and one 100Mb Ethernet NIC (yeah, OK, so I work for a big company). The box was already loaded with Windows2000 Professional, IIS, IE5.5, and Service Pack 2.

You'll also need an Internet connection. Once you have your box set up, use it to download the latest versions of:
Kiwi Syslogd for Windows
http://www.kiwi-enterprises.com/software_downloads.htm
ActiveState Perl
http://aspn.activestate.com/ASPN/Downloads/ActivePerl/
MySQL Shareware 3.23.39:
http://www.silicondefense.com/software/win32/mysql-3.23.39a-win.zip

The "create_mysql" database creator:
http://www.silicondefense.com/software/win32/create_mysql
PHP 4.0.6:
http://www.silicondefense.com/software/win32/php-4.0.6-Win32.zip
ADODB 1.1.2:
http://www.silicondefense.com/software/win32/adodb112.zip
ACID 0.9.6b6:
http://www.silicondefense.com/software/win32/acid-0.9.6b6.tar.gz
Logsnorter
http://www.snort.org/Files/logsnorter.tar.gz

Got 'em? Good. Now, assuming you're running some variety of Windows, follow these steps:
****IMPORTANT:** the instructions for the installation of MySQL, PHP, ADODB, and ACID were borrowed from Michael Steele's page, "Snort on Windows 98/ME/NT/2000 using MySQL and ACID", rev 1.5 dated 23 July 2001, posted at http://www.snort.org/papers/SnortOnAcid_Win32.htm. His work is only slightly revised here to accommodate this particular installation. **My respects and apologies to Michael.**

Install Kiwi using all the default settings.
Note: If you don't think you'll be parsing your PIX syslog entries in realtime, you can skip the Kiwi installation.

Install ActiveState Perl using all the default settings.
After the installation completes, open a command prompt, type "ppm", <press Enter>.
PPM will start. At the ppm> prompt, type "install net-syslog" <press Enter>.
The Syslog package will download and install.
Next, at the ppm> prompt, type "install dbi" <press Enter>.
The Database Interface package will download and install.
Finally, at the ppm> prompt, type "install dbd-mysql" <press Enter>.
The MySQL database drivers will download and install.
Type "quit" and the "exit"
Depending your version of Perl, those extra modules should be all you need. The fastest thing to do is run the script to see where it dies. If you find you're missing a module, use PPM to download and install whatever you need. Otherwise, I recommend checking your installation of Perl before you start.
Note: If you don't think you'll be parsing your PIX syslog entries in realtime, you can comment out all of the modules in the script except for GetOpt::Std and DBI.

Install MySQL using ALL the default settings.
Note: If you're running Windows 2000 Server or Advanced Server, at the command prompt prior to installation type: "Change User /install", or, install MySQL from the Add/Remove Programs panel.

If everything installed correctly you should now have a MySQL icon in the system tray.
Right click the MySQL icon in the system tray and select "Show Me".
Select the "Start Check" tab. Everything in the tab should read "Ok" and "Yes".
Note: If there are any errors then you might reboot and check the tab again prior to proceeding.

Select the "my.ini setup" tab. Comment out the User Name and Password, and then click "Save Modifications".
Click the "Create Shortcut on Start Menu" button.
Note: This will create an entry in the startup folder that will run the administration panel each time you restart your machine.

Creating a Win32 MySQL database

Right click on the MySQL icon in the System Tray and select "Show Me". The MySQL GUI will display.
Choose the Database tab, right click on your server name, and select “Create Database”.
Type your database name in dialog box. I called mine “Snort”.

You’ll need to create a user at the command prompt.
Open a command prompt and navigate to the "C:\MySQL\Bin" directory. At the prompt, type “MySQL“.
The prompt will change to "mysql> "
At mysql>, type: \u mysql; <press Enter> (sets the database to mysql)
Type: grant INSERT,SELECT,CREATE,DELETE on snort.* to snort@localhost; <press Enter>

To confirm the user addition, at "mysql>" type: \u mysql <press enter> (this sets the database to mysql)
At "mysql> " type: show tables; (you should see a table’s list with a user entry)
At "mysql> " type: select * from user; (you should see the user "snort" listed)

Creating Tables into MySQL for Acid
Copy the file called "create_mysql" into the C:\MySQL\Bin folder.
At the command line, navigate to the "C:\MySQL\Bin" folder.
At the prompt, type: MySQL -u snort snort < C:\MySQL\Bin\create_mysql
Note: To ensure the tables were added, right click on the MySQL icon in the system tray and choose “Show Me”.
Select the Database tab, and then, in the Databases window, select “Snort”.
Below, in the Databases Tables pane, you should see some entries under Snort.

Installing the Acid Plug-in and associated programs
Note: There are five tasks to complete before Acid will display: install a Web server; install PHP; install ADODB; edit acid_conf.php'; and finally, edit 'adodb.php.conf'.

Install the web server on your Windows machine before continuing.
If you don’t know how, go here:
http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/prodtechnol/iis/tips/iis.asp

Installing PHP (the HTML embedded scripting language)
Uncompress PHP into the C:\Snort\PHP folder.
Copy the file ‘php.ini-dist’ to your ROOT directory and rename it to ‘php.ini’.
Note: Your WINDOWS or SYSTEMROOT directory is typically:
c:\windows for Windows 95/98
c:\winnt or c:\winnt40 for NT/2000 servers

Configure PHP extensions for 98/NT/2000 Server running IIS 4/5 or PWS
Start the Microsoft Management Console (it may also be named “Internet Services Manager” in either your Windows NT 4.0 Option Pack branch, or in the Control Panel=>Administrative Tools branch under Windows 2000).

Right click on your Web server node (it will probably be named “Default Web Server”), and select “Properties”.
Under “Home Directory”, “Virtual Directory”, or “Directory”, click the “Configuration” button, and then select the “Applications Mappings” tab.

Click “Add”, and in the Executable box, type: c:\snort\php\php.exe
In the Extension box, type: “.php”.
Leave “Method exclusions” blank if there is one .
Check the Script engine checkbox.
Note: You may also want to check the “check that file exists” box. For a small performance

penalty, IIS (or PWS) will check that the script file exists and sort out authentication before firing up php. This means that you will get sensible 404 style error messages instead of cgi errors complaining that php did not output any data.

Click "OK" then "Apply" then "OK"

Install ADODB - A high quality database library
Uncompress ADODB into the C:\Snort\ADODB folder.
Edit the ADODB.INC.PHP file to reflect the location of the ADODB folder, i.e.:
$ADODB_Database = 'C:\Snort\adodb';
Uncompress and move the Acid folder into the root folder of your default website, i.e.:
C:\Inetpub\wwwroot\
Configure the Acid 'acid_conf.php' file in the Acid folder. You should only have to edit the variables below:

$DBlib_path = "C:\Snort\ADODB";
$alert_dbname = "snort";
$alert_host = "localhost";
$alert_port = "";
$alert_user = "snort";
$alert_password = "";

Reboot your machine!

Start your browser and type: http://localhost/Acid/Index.html
Note: You will receive a configuration error the first time Acid is run. Proceed to the Setup page and click on the "Setup" option hyperlink, then click "Create ACID AG" to complete the Acid Alert Group configuration.

Return to your browser and retype "http://localhost/Acid/Index.html"

Set up Logsnorter
Uncompress Logsnorter to whatever folder from which you like to run your Perl scripts.
****IMPORTANT:** Logsnorter.pl is a really cool Perl script written by Jason Haar (jhaar@users.sourceforge.net). The original intent of the script was to parse IPChains, Cisco router, or Linux IPFW syslog messages for insertion into a MySQL database. The idea is to be able to correlate your firewall logs with your Snort logs in ACID.
To get the script to work with PIX firewall logs, I had to modify some of the subroutines, as well as develop a few extra sets of regular expressions. If, like me, you're not going to be working with IPChains, IPFW, or Cisco routers, I'm going to recommend that you comment out some of Jason's code. **My respects and apologies to Jason**.
****Caveat:** The format of PIX logs can change from IOS version to IOS version. The IOS of my firewalls is 4.3.2. I don't guarantee that the modifications I submit here will work with any other IOS versions. As a matter fact, I don't guarantee anything at all. This setup works for me, but just like everything else opensource, you'll have to play with the script and the database structure to get it to produce the results you want. Make sure you have read the script **completely** before make **any** modifications!
Having said that, and having apologized to Jason:
Modify Logsnorter.pl with:

```
### New find_sensor
sub find_sensor {  }
sub crash {  my( $msg )=@_; print STDERR "CRASH: $msg\n"; }
sub getaddr { my( $name ) = @_; return $name; }
sub insert_event {  }
```

```
sub get_ip_proto {
  my ($proto)=@_;
  if ($proto =~ /^[0-9]+$/) {
          $ip_proto=$proto
  } else {
          my $ip_proto_name,$ip_proto_alias,$ip_proto_num;
          ($ip_proto_name,$ip_proto_alias,$ip_proto_num)=getprotobyname($proto);
          $ip_proto=$ip_proto_num;
  }
}

#### NEW STUFF
#### Paste this into the logsnorter script. Add a call around line #127
#### (in the while loop) to call parse_cisco_pix_logs()

sub parse_cisco_pix_logs {

###PIX configuration files have a built-in host table to simplify the logs, as well as the
###construction of conduits and ACL's. You'll need to get the IP addresses back
###in the log entries.
###Paste the host table from your PIX config here, below "%names = ( :

  %names = (
  'first_server'                => 'a.b.c.65',
  'second_server'               => 'a.b.c.32',
  'third_server'                => 'a.b.c.50',
  'fourth_server'               => 'a.b.c.31',

  );


  $logtype='ciscopix';
  $interface_prepend="syslog_";

  s/^messages\.([0-9]+)://g;

  $data = $_;

  if (/PIX/i) {
          $ip_ver=4;
          $ip_hlen=$ip_tos=$ip_len=$ip_id=$ip_flags=$ip_off=$ip_ttl=$ip_proto=$ip_csum=0;
          $tcp_seq=$tcp_ack=$tcp_off=$tcp_res=$tcp_flags=$tcp_win=$tcp_csum=$tcp_urp=0;
          $udp_len=$udp_csum=0;
          $icmp_csum=$icmp_id=$icmp_seq=0;


          if( /^(\w+)\s+([0-9]+) ([0-9]+):([0-9]+):([0-9]+) ([^\s]+) %PIX-([0-9]+)-([0-9]+): Deny inbound
icmp/  ) {

                  /^(\w+)\s+([0-9]+) ([0-9]+):([0-9]+):([0-9]+) ([^\s]+) ([^\s]+): ([\w]+) ([\w]+) ([\w]+)
([\w]+) ([\w]+):([0-9\.]+) ([\w]+) ([\w]+):(.*) \(type ([0-9]+), code ([0-9]+)\)$/;

                  $ruleset = 1;     # Do I need this??
                  $interface = "PIX_INTERFACE";
                  $num_packets = 1;
```

```
                $month=$1;
                $day=$2;
                $hour=$3;
                $min=$4;
                $sec=$5;
                $cisco=$6;
                $junk=$7;
                $auth=$8;
                $junk=$9;
                $proto=$10;
                $junk=$11;
                $junk=$12;
                $src_addr=$13;
                $junk=$14;
                $junk=$15;
                $dst_addr=$16;
                $icmp_type=$17;
                $icmp_code=$18;
                next if ($auth ne "Deny");
                if ( $cisco eq "" ) {
                &crash("Cannot interpret Cisco icmp line: $_\n");
                }
                $cisco=~s/\'/\'/g;
                $hstip=&getaddr($cisco);
#                }
                if ($hstip ne $sensor_ip) {
        $sensor_ip=$hstip;
        &find_sensor($sensor_ip,$interface);
                }

        }
        elsif ( /^(\w+)\s+([0-9]+) ([0-9]+):([0-9]+):([0-9]+) ([^\s]+) %PIX-([0-9]+)-([0-9]+): Deny
inbound UDP/i   ) {

                /^(\w+)\s+([0-9]+) ([0-9]+):([0-9]+):([0-9]+) ([^\s]+) ([^\s]+): ([\w]+) ([\w]+) ([\w]+)
([\w]+) ([0-9\.]+)\/([0-9]+) ([\w]+) ([0-9\.]+)\/([0-9]+)$/;

                $ruleset = 1;     # Do I need this??
                $interface = "PIX_INTERFACE";
                $num_packets = 1;

                $month=$1;
                $day=$2;
                $hour=$3;
                $min=$4;
                $sec=$5;
                $cisco=$6;
                $junk=$7;
                $auth=$8;
                $junk=$9;
                $proto=$10;
                $junk=$11;
                $src_addr=$12;
                $src_port=$13;
                $junk=$14;
                $dst_addr=$15;
```

```
                $dst_port=$16;
                next if ($auth ne "Deny");
                if ( $cisco eq "" ) {
        &crash("Cannot interpret udp Cisco line: $_\n");
         }
                $cisco=~s/\'/\'/g;
                $hstip=&getaddr($cisco);
                if ($hstip ne $sensor_ip) {
        $sensor_ip=$hstip;
        &find_sensor($sensor_ip,$interface);
                }
         }
         elsif (/^(\w+)\s+([0-9]+) ([0-9]+):([0-9]+):([0-9]+) ([^\s]+) %PIX-([0-9]+)-([0-9]+): Inbound
TCP connection denied/i) {

                /^(\w+)\s+([0-9]+) ([0-9]+):([0-9]+):([0-9]+) ([^\s]+) ([^\s]+): ([\w]+) ([\w]+) ([\w]+)
([\w]+) ([\w]+) ([0-9\.]+)\/([0-9]+) ([\w]+) ([0-9\.]+)\/([0-9]+) ([\w]+) ([\w]+)$/;

                $ruleset = 1;      # Do I need this??
                $interface = "PIX_INTERFACE";
                $num_packets = 1;

                $month=$1;
                $day=$2;
                $hour=$3;
                $min=$4;
                $sec=$5;
                $cisco=$6;
                $junk=$7;
                $junk=$8;
                $proto=$9;
                $junk=$10;
                $auth=$11;
                $junk=$12;
                $src_addr=$13;
                $src_port=$14;
                $junk=$15;
                $dst_addr=$16;
                $dst_port=$17;
                $junk=$18;
                $tcp_flags=$19;

                next if ($auth ne "denied");
                if ( $cisco eq "" ) {
        &crash("Cannot interpret tcp Cisco line: $_\n");
                }
                $cisco=~s/\'/\'/g;
                $hstip=&getaddr($cisco);
                $dst_addr=$hstip;
#                $ip_proto="0";
                if ($hstip ne $sensor_ip) {
        $sensor_ip=$hstip;
        &find_sensor($sensor_ip,$interface);
                }
         }
```

```
        else {
                &crash("Cisco error line $line: doesn't match known type: $_\n");
        }

        $dst_addr = $names{$dst_addr} if( $dst_addr =~ /[^0-9\.]/ );

        #Convert proto back to number
        &get_ip_proto($proto);

        if (!$src_addr || !$dst_addr || $ip_proto eq "" || ($ip_proto == 1 && !$icmp_type) ||
!$ruleset) {
                &crash("Cisco Error line $line: insuffient info - broken syslog entry!
(src_addr=$src_addr,dst_addr=$dst_addr,ip_proto=$ip_proto,icmp_type=$icmp_type,acl=$rulese
t)");
        }


        $src_addr=~/^([0-9]+)\.([0-9]+)\.([0-9]+)\.([0-9]+)$/;
        $array_src_addr[0]=$1;
        $array_src_addr[1]=$2;
        $array_src_addr[2]=$3;
        $array_src_addr[3]=$4;

        $dst_addr=~/^([0-9]+)\.([0-9]+)\.([0-9]+)\.([0-9]+)$/;
        $array_dst_addr[0]=$1;
        $array_dst_addr[1]=$2;
        $array_dst_addr[2]=$3;
        $array_dst_addr[3]=$4;

        if ($num_packets == 0) {
                &crash("Whaa! No number of packets for line $line - duh...");
        }
        &insert_event;
  }
}
```

****Notes:** Comment out the following calls and subroutines if you won't be working with these:

```
  &parse_cisco_logs;
  &parse_ipchains_logs;
  &parse_ipfwadm_logs;
```

I managed to work out all the syntax errors, as well as the errors caused by the regular expressions. The script no longer errors-out on any of my test data (i.e. none of the &crash() calls are made), and it now seems to run just fine on it's own.
You might want to test the subroutine on some static logs. To do so, paste "parse_cisco_pix_logs", along with "find_sensor", into your favorite text editor and add the following:

```
### HACKED TEST ENVIRONMENT
MAIN: {
 open IN, "<test.txt" || die("Could not open logfile");
 while( <IN> ) { &parse_cisco_pix_logs(); }
 close IN;
}
```

Save the file as <whatever>.pl and run it from the command line. In this simple form, you can use the script to process static log files if you've decided not to pull the entries from a syslog stream. Just set the script up in AT (or Task Scheduler) and let her rip….
Also, I used "Hacked Test Environment" in the main script to test the database insertion, too.
As always, your mileage may vary…

Point your PIX firewall syslog stream towards your box
Check the Kiwi console to ensure you're receiving your log entries.

Fire up Logsnorter
Make sure you configure Logsnorter with the location of your log files. You can then run Logsnorter from the command line or from AT (or Task Scheduler). If everything is working correctly, you should see your database begin to grow exponentially. Let it run for a while, and check the database for errors before you try to pull anything up with ACID.

The Challenge Met: Produce your reports
Once you've checked everything out and you're beginning to see your log entries in ACID, give your boss the URL of your web server and let him marvel at your resourcefulness. Or, condense and can the reports yourself and distribute them via e-mail. Either way, you'll be due for a big raise and a promotion. You might even be rewarded with one of those fancy intrusion detection systems you saw at SANS.
Or, since you've created your database with Snort in mind, you can refer back to http://www.snort.org/papers/SnortOnAcid_Win32.htm to add and configure Snort for your system. You can then begin correlating your firewall logs with your intrusion data.

References: See Appendix C

Analyze This

Introduction

This section will examine some of MY.NET's Snort intrusion detection system data. The analysis aims to focus on external threats and potential problems on the internal network.

Summary
For brevity, and because of time constraints, five days worth of logs, collected between 3.25.01 and 3.29.01, were chosen at random for examination. The five days worth of logs turned out to be more than enough to get some good insight into the network that produced them.

The logs were analyzed with various tools and scripts. Please refer to Appendix B for details.

The data supplied for analysis consisted of Snort alert logs, Snort portscan logs and Snort OOS logs. These files were downloaded:

	ALERT FILES		SCAN FILES		OOS FILES	
DATE	FILE	SIZE	FILE	SIZE	FILE	SIZE
3/25/2001	Alert-26-Mar.gz	124k	SnortScan-26-Mar.gz	416k	OOS-Mar.26.2001.packets.de0.gz	5k
3/26/2001	Alert-27-Mar.gz	144k	SnortScan-27-Mar.gz	328k	OOS-Mar.27.2001.packets.de0.gz	4k
3/27/2001	Alert-28-Mar.gz	173k	SnortScan-28-Mar.gz	271k	OOS-Mar.28.2001.packets.de0.gz	3k
3/28/2001	Alert-29-Mar.gz	149k	SnortScan-29-Mar.gz	258k	OOS-Mar.29.2001.packets.de0.gz	57k
3/29/2001	Alert-30-Mar.gz	94k	SnortScan-30-Mar.gz	262k	OOS-Mar.30.2001.packets.de0.gz	79k

Overall, the network is plagued with security problems, from open servers to Trojans to unrestricted traffic. In the five days covered in this analysis, the MY.NET intrusion detection system recorded better than 20,000 events covering nineteen unique signatures:
Earliest alert at **00:02:37**.218486 *on 03/25/2001*
Latest alert at **23:44:53**.587717 *on 03/29/2001*

Total events:	20765
Signatures recorded:	19
Source IP recorded:	387
Destination IP recorded:	2644
Portscan detected:	2301

None of the events are trivial. At the time the logs were generated, the ten external hosts in the table below represent the majority of detects on the MY.NET network. The table can be thought of as a "Top 10 Attackers" list. Registration information is provided:

Attacks	Signature	IP Address	Host	Registration
8926	Attempted Sun RPC high port access	63.121.232.185	Does Not Resolve	UUNET Technologies, Inc. (NETBLK-UUNET63) UUNET63 63.64.0.0 - 63.127.255.255 Sigecom (NETBLK-UU-63-121-232) UU-63-121-232 63.121.232.0 - 63.121.239.255
6473	Watchlist 000220 IL-ISDNNET-990517	212.179.4.50	Does Not Resolve	**inetnum: 212.179.4.48 - 212.179.4.63** netname: SCP-SYSTEMS-LTD descr: SCP-SYSTEMS-LAN country: IL admin-c: ES4966-RIPE tech-c: NP469-RIPE status: ASSIGNED PA notify: hostmaster@isdn.net.il mnt-by: RIPE-NCC-NONE-MNT **route: 212.179.0.0/17** descr: ISDN Net Ltd. origin: AS8551 notify: hostmaster@isdn.net.il mnt-by: AS8551-MNT **person: Eran Shchori** address: BEZEQ INTERNATIONAL address: 40 Hashacham Street address: Petach-Tikva 49170 Israel phone: +972 3 9257710 fax-no: +972 3 9257726 e-mail: hostmaster@bezeqint.net

				nic-hdl: ES4966-RIPE
1432	SYN-FIN scan!	61.11.252.117	Does Not Resolve	inetnum 61.11.249.64 - 61.11.254.255 Origin HK-IMS-3 descr Thaicom-IMS, IR country TH Admin. Contact PB29-AP Tech. Contact TU8-AP mnt-by MAINT-TH-THAICOM changed parkb@thaicom.net 20010405 person Park Boonyubol address 41/103 Ratanathibet Road, address Bangkasor, Nonthaburi 11000 country TH phone +66-2-591-0736 fax-no +66-2-591-0719 e-mail parkb@thaicom.net NIC Handle PB29-AP mnt-by MAINT-TH-THAICOM
963	Watchlist 000220 IL-ISDNNET-990517	212.179.5.89	clnt-5089.bezeqint.net	See Above
420	Watchlist 000220 IL-ISDNNET-990517	212.179.5.184	clnt-184.bezeqint.net	See Above
372	Possible RAMEN server activity	164.67.21.63	ts11-54.dialup.bol.ucla.edu	Campus Network Services (NET-UCLANET3) UCLA Communications Technology Services Bldg CSB1 2nd floor Los Angeles, CA 90095-1363 US Netname: UCLANET3 Netblock: 164.67.0.0 - 164.67.255.255 Coordinator: University of California, Los Angeles (NO102-ORG-ARIN) noc@NOC.UCLA.EDU +1 310 206 5345
293	Watchlist 000222 NET-NCFC	159.226.92.9	lsec.cc.ac.cn	The Computer Network Center Chinese Academy of Sciences (NET-NCFC) P.O. Box 2704-10, Institute of Computing Technology Chinese Academy of Sciences Beijing 100080, China CN Netname: NCFC Netblock: 159.226.0.0 - 159.226.255.255 Coordinator: Qian, Haulin (QH3-ARIN) hlqian@NS.CNC.AC.CN +86 1 2569960
283	connect to 515 from outside	216.191.147.13	www.holodesign.net	MetroNet Communications Group Inc. (NETBLK-

				METRONET-CIDR-2) 100 King St. West, Suite 2900 Toronto, Ontario M5X 1B5 CA Netname: METRONET-CIDR-2 Netblock: 216.191.0.0 - 216.191.255.255 Maintainer: MTCO Coordinator: Noc, Metronet Toronto (MTN-ARIN) NOCToronto@METRONET.CA
235	Watchlist 000220 IL-ISDNNET-990517	212.179.80.156	PT712156.bezeqint.net	See Above
109	Back Orifice	24.162.245.198	rdu162-245-198.nc.rr.com	ServiceCo LLC - Road Runner (NET-ROAD-RUNNER-5) 13241 Woodland Park Road Herndon, VA 20171 US Netname: ROAD-RUNNER-5 Netblock: 24.160.0.0 - 24.170.127.255 Maintainer: SCRR Coordinator: ServiceCo LLC (ZS30-ARIN) abuse@rr.com 1-703-345-3416

The ten internal hosts in the table below make up the MY.NET "Top Talkers" list. These hosts received the highest number of attacks and should be thought of as in need of the quickest attention:

Host	Signature	Attacks
MY.NET.221.198	Attempted Sun RPC high port access	8926
MY.NET.222.154	Watchlist 000220 IL-ISDNNET-990517	6565
MY.NET.202.202	Watchlist 000220 IL-ISDNNET-990517	526
MY.NET.219.38	Watchlist 000220 IL-ISDNNET-990517	479
MY.NET.223.254	Watchlist 000220 IL-ISDNNET-990517	409
MY.NET.144.54	Watchlist 000222 NET-NCFC	293
MY.NET.206.254	Watchlist 000220 IL-ISDNNET-990517	235
MY.NET.202.54	Queso fingerprint	98
MY.NET.178.42	Russia Dynamo - SANS Flash 28-jul-00	40
MY.NET.206.118	Possible RAMEN server activity	15

To help to correct the problems noted here, the analysis has been split into three sections. The first section will cover the general alerts. The alerts will be prioritized by occurrence, followed by a brief description of each. Each description will include a table of "Top Talkers" and a recommended course of action.
The second section will cover the scans seen on the network.
The third section will summarize the Out Of Specification (OOS) traffic recorded.

Alert Details

Detects prioritized by occurrence:

Signature	Alerts	Destinations	Sources
Attempted Sun RPC high port access	8926	1	1
Watchlist 000220 IL-ISDNNET-990517	8439	27	29
SYN-FIN Scan!	1436	1436	2
Possible RAMEN server activity	609	364	145
Watchlist 000222 NET-NCFC	324	8	7
Connect to 515 from outside	299	247	2
External RPC call	265	231	5
SMB Name Wildcard	182	77	102
Queso fingerprint	132	20	10
Back Orifice	132	132	2
WinGate 1080 Attempt	87	47	43
Russia Dynamo - SANS Flash 28-jul-00	41	2	2
Tiny Fragments - Possible Hostile Activity	27	12	2
NMAP TCP ping!	26	11	8
Null scan!	23	19	19
Port 55850 tcp - Possible myserver activity - ref. 010313-1	13	7	5
SUNRPC Highport Access!	10	1	1
Connect to 515 from inside	1	1	1
STATDX UDP attack	1	1	1

Attempted Sun RPC High Port Access

Attempted Sun RPC HPA alerts can be generated when a remote host attempts to connect to an internal host on a port that may be used for Remote Procedure Calls. The port is typically numbered above 32000. Access from the Internet to RPC ports should not be allowed unless under very controlled conditions, and then monitored carefully. Most probes or connection attempts from the Internet to RPC ports are with the intent to compromise a system (see http://www.sans.org/infosecFAQ/cmsd.htm for an example). Ports 111 and 32771 are favorite targets for attackers. Another good discussion about RPC is here: http://www.sans.org/newlook/resources/IDFAQ/trouble_RPCs.htm

There were 8,926 Attempted Sun RPC HPA events generated by one remote host to one internal client in one day over five hours. The culprit was:

Alerts	IP Address	Host
8926	63.121.232.185	Does not resolve

Snortsnarf synopses:

03/26-19:42:24.114048 [**] Attempted Sun RPC high port access [**] 63.121.232.185:32768 -> 123.456.221.198:32771

03/26-19:42:27.178486 [**] Attempted Sun RPC high port access [**] 63.121.232.185:32768 -> 123.456.221.198:32771

03/26-19:42:27.602308 [**] Attempted Sun RPC high port access [**] 63.121.232.185:32768 ->

123.456.221.198:32771

Correlation
There is no correlation in the OOS logs.
There is no direct correlation to the remote host at www.incidents.org, www.mynetwatchman.com, or www.dsheild.org; nor are any of the ports listed on the trojan list at http://www.sys-security.com/html/papers/trojan_list.html. There is this at http://www.incidents.org/archives/y2k/123199-2030.htm:
"If it's any help, it's most likely that 32772 is any of several RPC services that get started by default on Solaris. Since the default "anonymous" port for TCP and UDP is set to 32768, these things wind up slightly above that. So you're probably seeing scans for rstatd and sadmind, both of which have instant-root buffer overflows."
As well, there are loads of scans to port 32771 reported to Incidents.org here http://www.incidents.org/archives/y2k/021600.htm, here http://www.incidents.org/archives/y2k/022000.htm, and here http://www.incidents.org/archives/y2k/011900.htm, just to name a few. But none of the scans reported in the archives start on ports above 32000 and last for five hours.
Then there is the case of ICQ. It's been documented that ICQ can generate alerts on this ruleset. However, ICQ of the common AOL variety tends to source at around port 4000. See http://www.shmoo.com/mail/fw1/mar00/msg00020.shtml and http://www.sans.org/y2k/practical/Leigh_Heyman.doc

Defensive Recommendations
I don't think this is ICQ. It's possible, but I can't correlate it to anything similar. I don't think it's a scan, either. It's a bold (or stupid) attacker that would scan one host for five hours. It looks to me like the two boxes are connected by a service and passing traffic.
It would be a good idea to have a look at the local machine. If nothing else, start a packet capture to find out what's going on. The machine could be compromised, or it could be on "Special Assignment"; but it would be worth a look to find out if RPC services are available to the Internet. As well, the firewall rulesets should be reviewed. Any traffic to TCP port 11 to and from the Internet should be blocked.
In the event the traffic turns out to be ICQ, it would be a good idea to review the current security policy for references to ICQ. ICQ can adversely impact productivity and is notoriously insecure. Many attackers use ICQ as a conduit for spreading trojans and viruses. For a discussion on the associated security issues, see http://blacksun.box.sk/icq.html

Watchlist 000220 IL-ISDNNET-990517

A watchlist of this sort is created to monitor traffic from a network that has a history of problems with security. In this case, the ruleset is meant to capture traffic originating from an Israeli ISP, Bezeq International (ISDN.NET.IL).

There were 8,439 Watchlist 000220 IL-ISDNNET-990517 events generated by 29 remote hosts to 27 internal hosts.

The top five source addresses:

Alerts	IP Address	Host
6473	212.179.4.50	Does Not Resolve
963	212.179.5.89	clnt-5089.bezeqint.net.
420	212.179.5.184	clnt-5184.bezeqint.net.
235	212.179.80.156	pt712156.bezeqint.net.
83	212.179.27.6	clnt-27006.bezeqint.net.

The top five destination hosts:

Alerts	Host
6565	MY.NET.222.154
526	MY.NET.202.202
479	MY.NET.219.38
409	MY.NET.223.254
235	MY.NET.206.254

The top five destination ports:

Alerts	Port	Service
6473	4969	Unknown
526	4898	Unknown
437	6346	Gnutella
409	6688	Napster
435	4352	Unknown

Correlation:

http://www.incidents.org/archives/y2k/090800.htm

09/06-06:16:00.880267 62.40.188.165:1 -> MY.NET.205.238:4969
TCP TTL:112 TOS:0x0 ID:57534 DF
21SFR*AU Seq: 0x1A2B00F1 Ack: 0xC9470184 Win: 0x5010
1A 2B 00 F1 C9 47 01 84 1E F7 50 10 22 38 2A 43 .+...G....P."8*C
90 A8 87 1B AB E1 ......

http://www.incidents.org/archives/y2k/032600-2000.htm

03/25-05:01:50.296285 169.229.110.69:1039 -> MY.NET.201.142:6688
TCP TTL:113 TOS:0x0 ID:65477 DF
SFRPA* Seq: 0x7F Ack: 0xD6AC00A6 Win: 0x5010
00 00 3E 89 36 39 0D D8 CD 2E EC 4B AB D2 ..>.69.....K..

03/25-08:12:23.539724 24.200.89.143:1105 -> MY.NET.97.80:6688
TCP TTL:114 TOS:0x0 ID:50662 DF
SF**A*21 Seq: 0x451920 Ack: 0x5F1C Win: 0x5010
TCP Options => EOL EOL EOL EOL EOL EOL SackOK Opt 141 (40):
82C1 0014 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000

03/25-08:12:32.906576 24.200.89.143:1105 -> MY.NET.97.80:6688
TCP TTL:114 TOS:0x0 ID:53480 DF
SF*PA*21 Seq: 0x45 Ack: 0x19205F1D Win: 0x5010
19 20 5F 1D 20 DB 50 10 21 80 80 89 00 00 00 00 . _. .P.!.....
00 00 ..

03/25-08:14:10.442604 24.200.89.143:1105 -> MY.NET.97.80:6688
TCP TTL:114 TOS:0x0 ID:33274 DF
SF*P*U21 Seq: 0x45 Ack: 0x19205F20 Win: 0x5010
TCP Options => EOL EOL EOL EOL EOL EOL SackOK Opt 20 (21):
1617 1819 0000 0000 0000 0000 0000 0000 0000 0000 EOL EOL
EOL EOL EOL EOL EOL EOL EOL EOL

03/25-08:14:32.936127 24.200.89.143:1105 -> MY.NET.97.80:6688
TCP TTL:114 TOS:0x0 ID:59390 DF
SF***U2 Seq: 0x45 Ack: 0x19205F21 Win: 0x5010
TCP Options => EOL EOL EOL EOL EOL EOL SackOK Opt 20 (21):
1617 1819 0000 0000 0000 0000 0000 0000 0000 0000 EOL EOL
EOL EOL EOL EOL EOL

http://www.sans.org/y2k/practical/Paul_Asadoorian_GIAC.doc

Defensive Recommendation

Port 6688 is commonly associated with Napster: http://www.sans.org/infosecFAQ/napster.htm

Port 6346 is commonly associated with Gnutella: http://www.snort.org/Database/portsearch.asp

Port Number	Protocol	Port Name
6346	TCP	Gnutella
6346	TCP	Gnutella-svc

The remaining ports are unknown.
Napster and Gnutella allow users to share files on their PC's with any other PC connected to the Internet that is using the same software. For the most part, the programs are used to download Mp3's or pictures.
The traffic on port 4969 was sourced from 212.179.4.50 and destined primarily for MY.NET.222.154. That same source generated just about all of the traffic to the internal network. After reading the Gnutella documentation, I find that you can change the default port on which the application listens, and that might explain the traffic to ports 4969, 4898, and 4352. Another explanation could be gaming. Quake comes to mind, but there are a myriad of others. There is no direct correlation to any one game. As well, SysSecurity's Trojan list http://www.sys-security.com/html/papers/trojan_list.html does not list a trojan for these ports.
Any machines on the internal network allowing connections from 212.79 to any of the documented ports should be given some scrutiny; especially connections to TCP ports 4969, 4898, and 4352. If access doesn't fall under the acceptable use policy, connections to these systems should be blocked. It might also be a good idea to start a packet capture just to find out what's going on.
As well, a review of current security policy for references to Napster and Gnutella is advised. While neither is a destructive application, it does affect productivity and can have an adverse effect on network bandwidth and system storage. Just shutting it off at the router or firewall might be the best course of action.
For more information see http://www.napster.com and http://gnutella.wego.com.

SYN-FIN Scan!

A SYN-FIN scan is a series of TCP packets sent with the SYN and FIN flags set in the header. TCP packets formatted in that way don't occur normally, and so indicate an intentional probe. Sometimes the probe is used as a way of fingerprinting an OS. Or, sometimes just to see if a service answers. A patient person can map an entire target network in this way. Scans like this can also be a prelude to a direct attack. See http://www.whitehats.com/info/IDS198 for details.
In this case, the scans were targeted across a network range and not to any one particular host. Since this attack specifically targets a port, the targeted port correlates to the service being probed.

There were 1436 SYN-FIN Scan! events generated from two remote hosts to 1436 internal hosts.

The two culprits:

Alerts	Source IP	Host
1432	61.11.252.117	Does Not Resolve
4	24.131.172.251	El10-24-131-172-251.ce.mediaone.net

The two destination ports:

Alerts	Port	Service
1427	21	FTP
4	111	RPC

Correlation
There is no direct correlation to these particular hosts, but this type of scan is very common: http://www.sans.org/y2k/102600.htm

```
Oct 20 19:40:56 My.Net.Here.209 ASCEND: wan4 tcp 255.255.255.255;21   <- 158.38.74.130;21 40 syn fin !pass (totcp-1)
Oct 20 19:42:06 My.Net.Here.209 ASCEND: wan4 tcp My.Net.Here.208;21   <- 158.38.74.130;21 40 syn fin !pass (totcp-1)
Oct 20 19:42:06 My.Net.Here.209 ASCEND: wan4 tcp My.Net.Here.209;21   <- 158.38.74.130;21 40 syn fin !pass (totcp-1)
Oct 20 19:42:06 My.Net.Here.209 ASCEND: wan4 tcp My.Net.Here.211;21   <- 158.38.74.130;21 40 syn fin !pass (totcp-1)
Oct 20 19:42:06 My.Net.Here.209 ASCEND: wan4 tcp My.Net.Here.212;21   <- 158.38.74.130;21 40 syn fin !pass (totcp-1)
```

http://www.sans.org/y2k/081600-1500.htm

Dear GIAC Folks, This is an edited version of my internal security report for the week of Aug 8 - Aug 13, 2000. This is "dumbed down" a bit for non-security people.

DETECT ANALYSIS

08-09-2000, 16:13. Eleven denied TCP connections from 195.42.167.130/21 to every IP address in our global address block (ext.net.230.98 – ext.net.230.110), TCP port 21 (FTP control channel). Three instances of FIN SYN flags.

This appears to be a sequential FIN/SYN scan for an open FTP port. We have no externally accessible FTP servers. The FIN SYN flag combination does not appear in normal TCP/IP traffic and is the result of a crafted packet generator or some type of automated scanning tool. FIN is what is used to tell a system that a connection is being terminated. SYN is what is used to tell a system that a connection is being made. Attackers use FIN SYN flags together to attempt to bypass firewalls and to evade intrusion detection systems. It is highly unlikely that an intruder would be able to guess a state table entry and actually connect to our internal DNS, let alone obtain any information from it. Risk: low This scan appears to originate from a domain in Russia. Various sites in the .ru domain have been the source of network scans over the last several weeks, as reported by the SANS Institute.

http://www.sans.org/y2k/092800.htm

(Laurie@.edu)

```
Sep 24 22:16:56 hosty snort[395880]: SCAN-SYN FIN: 63.97.196.34:21 -> z.y.w.34:21
Sep 24 22:16:57 hostj snort[341]: SCAN-SYN FIN: 63.97.196.34:21 -> z.y.w.66:21
Sep 24 22:16:58 hostmi snort[15718]: SCAN-SYN FIN: 63.97.196.34:21 -> z.y.w.98:21
```

OOS alerts were generated. Each entry had flags that were set to **SF****. This is the mark of a crafted packet.

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/29-20:08:55.537797 61.11.252.117:21 -> MY.NET.1.9:21
TCP TTL:17 TOS:0x0 ID:39426
**SF**** Seq: 0x5FCCB718   Ack: 0x56468236   Win: 0x404
45 7D 41 96 2D 12                          E}A.-.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/29-20:08:55.709406 61.11.252.117:21 -> MY.NET.1.18:21
TCP TTL:18 TOS:0x0 ID:39426
**SF**** Seq: 0x5FCCB718   Ack: 0x56468236   Win: 0x404
00 00 00 00 00 00                          ......

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/29-15:03:27.720232 24.131.172.251:111 -> MY.NET.132.8:111
TCP TTL:33 TOS:0x0 ID:39426
**SF**** Seq: 0x3104ECC4   Ack: 0x6EA8402F   Win: 0x404
00 00 00 00 00 00                          ......

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/29-15:03:32.795547 24.131.172.251:111 -> MY.NET.133.8:111
TCP TTL:33 TOS:0x0 ID:39426
**SF**** Seq: 0x55D30C26   Ack: 0x36B3877D   Win: 0x404
00 00 00 00 00 00                          ......

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

Defensive Recommendations

Port 21

MyNetWatchman http://www.mynetwatchman.com/mynetwatchman/topports.asp consistently ranks port 21 in the top 10 ports scanned by attackers. This particular scan should be interpreted as a hostile attempt to find vulnerable FTP servers. It is also an attempt to fingerprint each system in an attempt to gather more information about the internal network. In this case, it should also be considered a prelude to an attack.

Port 111

MyNetWatchman http://www.mynetwatchman.com/mynetwatchman/topports.asp consistently ranks port 111 in the top 10 ports scanned by attackers. RPC is usually only active on Unix-based

systems and vulnerability exists in rpc.statd in several Linux distributions. RPC basically allows a remote computer to execute procedures and system calls on the target host.
All open FTP servers should be identified. Security on servers that allow FTP access should be reviewed. No server should allow anonymous access. If anonymous access is allowed, it should be closely monitored. Write access for FTP under any circumstance should be restricted.
See http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=FTP and www.cert.org for details.
For RPC, port 111 should be blocked at the firewall, or careful consideration given to specific services made available through RPC. It might also be a good idea to run a port scanner against the internal network to check for open services or vulnerable machines. An example of an RPC exploit is here: IN-99-04 Attacks Using Various RPC Services, but there are *many* more.
For detailed protocol information see: RFC 1057.
For more information see:
CERT Advisory CA-2000-17 http://www.cert.org/advisories/CA-2000-17.html
CVE-2000-0666 - http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0666

Possible RAMEN Server Activity

Ramen Server is a self-propagating worm particular to Linux. Once installed, the worm sets up a small web service listening on port 27374 to distribute a copy of itself to any connection making a request on that port. Please refer to
http://www.linuxsecurity.com/articles/network_security_article-2335.html for more detail.
****Note:** Port 27374 is also the default port of Sub7, a Windows-based Remote Access Trojan. Sub7 is not self-propagating.

There were 609 Possible Ramen Server Activity events generated from 145 remote and internal hosts to 364 remote and internal destination hosts.

The top five source hosts:

Alerts	Source IP	Host
314	164.67.21.63	ts11-54.dialup.bol.ucla.edu
19	MY.NET.209.86	Internal
15	63.10.42.245	1Cust245.tnt2.washington.dc.da.uu.net
14	MY.NET.206.118	Internal
11	63.10.42.73	1Cust73.tnt2.washington.dc.da.uu.net

The top five destination hosts:

Alerts	IP Address	
91	164.67.21.63	ts11-54.dialup.bol.ucla.edu
23	63.10.40.155	1Cust155.tnt1.washington.dc.da.uu.net
15	MY.NET.206.118	Internal
13	63.10.42.73	1Cust73.tnt2.washington.dc.da.uu.net
9	24.180.160.210	cc511169-a.twsn1.md.home.com

MY.NET hosts responding on TCP port 27374

MY.NET.97.10	MY.NET.97.46	MY.NET.97.211	MY.NET.98.1	MY.NET.98.45
MY.NET.98.64	MY.NET.98.89	MY.NET.98.84	MY.NET.98.85	MY.NET.98.72
MY.NET.98.105	MY.NET.98.100	MY.NET.98.120	MY.NET.98.117	MY.NET.98.113
MY.NET.98.93	MY.NET.98.121	MY.NET.98.116	MY.NET.98.133	MY.NET.98.112
MY.NET.98.132	MY.NET.98.169	MY.NET.98.177	MY.NET.98.160	MY.NET.156
MY.NET.98.164	MY.NET.98.176	MY.NET.98.165	MY.NET.98.148	MY.NET.98.193
MY.NET.98.200	MY.NET.98.201	MY.NET.98.193	MY.NET.98.189	MY.NET.99.16
MY.NET.99.28	MY.NET.99.37	MY.NET.99.32	MY.NET.99.21	MY.NET.99.60
MY.NET.99.52	MY.NET.99.65	MY.NET.99.85	MY.NET.99.104	MY.NET.99.116

MY.NET.99.133	MY.NET.99.152	MY.NET.99.168	MY.NET.99.169	MY.NET.99.145
MY.NET.99.172	MY.NET.99.185	MY.NET.100.3	MY.NET.100.78	MY.NET.100.111
MY.NET.100.148	MY.NET.100.186	MY.NET.100.202	MY.NET.100.199	MY.NET.100.210
MY.NET.100.227	MY.NET.100.223	MY.NET.104.46	MY.NET.104.42	MY.NET.104.66
MY.NET.104.71	MY.NET.104.211	MY.NET.105.30	MY.NET.105.178	MY.NET.106.164
MY.NET.106.204	MY.NET.107.64	MY.NET.107.105	MY.NET.107.141	MY.NET.107.169
MY.NET.107.165	MY.NET.107.197	MY.NET.107.236		

Defensive Recommendation:
The best course of action would be to shut off TCP port 27374 at the border router or the firewall and begin taking a close look at the internal network, starting with the list of responsive hosts. Traffic from 164.67.21.63 appears to be a scan. The Ramen worm itself does not scan for port 27374, so the scanner could be searching for machines already compromised by either Sub7 or Ramen. The problem is, 164.67.21.63 received 78 responses from MY.NET, meaning that the responsive boxen may already be compromised. The remainder of the source and destination traffic appears to be conversations between compromised hosts on TCP port 27374 as the traffic takes place over hours or days.

Watchlist 000222 NET-NCFC

A watchlist of this sort is created to monitor traffic from a network that has a history of security problems. In this case, the ruleset is meant to capture traffic originating from the Computer Network Center Chinese Academy of Sciences.

There were 324 Watchlist 000222 NET-NCFC events generated by seven remote hosts to eight internal hosts.

The top five source addresses:

Alerts	IP Address	Host
293	159.226.92.9	lsec.cc.ac.cn
22	159.226.41.166	Does Not Resolve
3	159.226.114.1	Does Not Resolve
2	159.226.45.3	aphy.iphy.ac.cn
2	159.226.201.43	Does Not Resolve

Connections
FTP
MY.NET.144.54 managed more than two dozen FTP sessions to 159.226.92.9 between 12:38:36 on 3.26.01 and 14:44:24 on 3.28.01.

Telnet
MY.NET.110.81 is allowing telnet connections from 159.226.41.166 to TCP port 38848. The only instance recorded was from 17:40:19 to 17:47:19 on 3.26.01

Port 25
MY.NET.6.34, MY.NET.253.42, and MY.NET.6.47 are allowing SMTP connections from 159.226.114.1, 159.226.45.3, and 159.226.201.43 respectively. If the MY.NET hosts are the regular mailservers, these connections my not necessarily be a problem, but they should be monitored.

Correlations
http://www.sans.org/y2k/practical/Brent_Deterding_GCIA.doc

```
attacks  from            to              method
=============================================================
   1847     159.226.63.190    MY.NET.253.43      Watchlist 000222 NET-NCFC
   1828     159.226.63.190    MY.NET.253.42      Watchlist 000222 NET-NCFC
   1827     159.226.63.190    MY.NET.253.41      Watchlist 000222 NET-NCFC
```

```
1150     159.226.45.108    MY.NET.6.7          Watchlist 000222 NET-NCFC
872     159.226.114.129   MY.NET.162.199      Watchlist 000222 NET-NCFC
```

http://www.sans.org/y2k/practical/Herschel_Gelman.html

Watchlist 000222 NET-NCFC: This is another custom rule someone at this site added; this one watches the Computer Center at the Chinese Academy of Sciences (159.226.0.0/16). The traffic captured is mostly SMTP traffic (9,914 of the 10,655 packets), but there is some very suspicious traffic in the remainder of the packets. There appear to be outgoing telnet sessions from the local network to various machines at the Chinese Academy of Sciences. MY.NET.162.121, MY.NET.163.32, and MY.NET.99.51 all telnetted out to 2 machines there, 159.226.41.99 and 159.226.45.3.
Not knowing the function of the monitored site, it is difficult to determine the importance of these sessions. For a university, or a corporation with interns from other countries, this may not be a big deal. For military installations this would probably be a high priority detect, and the local machines that telnetted out should be investigated.

http://www.sans.org/y2k/070800.htm

Defensive Recommendation
Telnet connections from the Internet should be blocked at the firewall. Review the security and acceptable use policy for references to external telnet connections. For more information on the risks associated with telnet, please refer to:
http://www.sans.org/newlook/resources/IDFAQ/telnet_rlogin.htm
It's important to review the email server logs. If logging isn't enabled on the email servers, it would be a good idea to get it started.
Review the acceptable use policy for references to FTP. Outbound FTP service is recommended only for authorized users. Inbound FTP connections should be blocked at the border router or the firewall. If it's necessary to provide public FTP services, the server should be closely monitored.

Connect To 515 From Outside

Port 515 is commonly associated with the Line Printer Daemon (LPD) in Unix. There are known exploits for this port that can allow an attacker to gain root access. Refer to http://www.cert.org/advisories/CA-2000-22.html, http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0917, http://www.incidents.org/cid/query/top_10port_7.php and http://www.securityfocus.com/bid/1711 for details.

There were 299 Connect To 515 From Outside events generated from 2 remote hosts to 247 internal hosts.

The two source addresses:

Alerts	IP Address	Host
283	216.191.147.13	www.holodesign.net
16	64.28.107.215	asnetworks107.215.asnetworks.net

The two source hosts show up in the scan logs as well:

```
Mar 27 05:34:27 216.191.147.13:3509 -> MY.NET.132.13:515 SYN **S*****
Mar 27 05:34:27 216.191.147.13:3511 -> MY.NET.132.15:515 SYN **S*****
Mar 27 05:34:27 216.191.147.13:3512 -> MY.NET.132.16:515 SYN **S*****
Mar 27 05:34:27 216.191.147.13:3515 -> MY.NET.132.19:515 SYN **S*****
Mar 27 05:34:27 216.191.147.13:3517 -> MY.NET.132.21:515 SYN **S*****
Mar 27 05:34:27 216.191.147.13:3518 -> MY.NET.132.22:515 SYN **S*****
Mar 27 05:34:27 216.191.147.13:3521 -> MY.NET.132.25:515 SYN **S*****
Mar 27 05:34:27 216.191.147.13:3523 -> MY.NET.132.27:515 SYN **S*****
Mar 27 05:34:27 216.191.147.13:3525 -> MY.NET.132.29:515 SYN **S*****
Mar 27 05:34:27 216.191.147.13:3526 -> MY.NET.132.30:515 SYN **S*****

Mar 27 22:10:20 64.28.107.215:1204 -> MY.NET.134.167:515 SYN **S*****
Mar 27 22:10:20 64.28.107.215:1251 -> MY.NET.134.214:515 SYN **S*****
```

Mar 27 22:10:20 64.28.107.215:1257 -> MY.NET.134.220:515 SYN **S*****
Mar 27 22:10:20 64.28.107.215:1259 -> MY.NET.134.222:515 SYN **S*****
Mar 27 22:10:20 64.28.107.215:1265 -> MY.NET.134.228:515 SYN **S*****
Mar 27 22:10:20 64.28.107.215:1492 -> MY.NET.135.200:515 SYN **S*****
Mar 27 22:10:20 64.28.107.215:1348 -> MY.NET.135.56:515 SYN **S*****
Mar 27 22:10:20 64.28.107.215:1350 -> MY.NET.135.58:515 SYN **S*****

Correlation
http://www.sans.org/y2k/011601-1430.htm
(Laurie@edu)

Server used for this query: [whois.arin.net]
Potomac Digitek Inc. (NETBLK-UU-63-70-211-192)
814 W. Diamond Ave Gathersburg, MD 20878 US
Netname: UU-63-70-211-192
Netblock: 63.70.211.192 - 63.70.211.223

Jan 6 04:58:00 hostmf /kernel: Connection attempt to TCP a.b.f.167:21 from 63.70.211.209:1534
Jan 6 05:00:39 hosth inetd[58148]: refused connection from 63.70.211.209, service ftpd (tcp)
Jan 6 05:16:31 hostmf /kernel: Connection attempt to TCP a.b.f.167:23 from 63.70.211.209:4625
Jan 6 05:19:32 hosth telnetd[58170]: ttloop: peer died: Undefined error: 0
Jan 6 05:00:39 hosth inetd[58148]: refused connection from 63.70.211.209, service ftpd (tcp)

Jan 6 05:00:38 63.70.211.209:4474 -> a.b.c.20:515 SYN ******S*
Jan 6 05:00:38 63.70.211.209:4486 -> a.b.c.32:515 SYN ******S*
Jan 6 05:00:38 63.70.211.209:4487 -> a.b.c.33:515 SYN ******S*
Jan 6 05:00:38 63.70.211.209:4516 -> a.b.c.62:515 SYN ******S*
Jan 6 05:00:39 63.70.211.209:4670 -> a.b.c.62:21 SYN ******S*
Jan 6 05:00:38 63.70.211.209:4521 -> a.b.c.67:515 SYN ******S*
Jan 6 05:00:39 63.70.211.209:4534 -> a.b.c.80:515 SYN ******S*
Jan 6 05:00:39 63.70.211.209:4668 -> a.b.c.207:515 SYN ******S*
Jan 6 05:00:39 63.70.211.209:4678 -> a.b.c.211:515 SYN ******S*
Jan 6 05:00:39 63.70.211.209:4679 -> a.b.c.212:515 SYN ******S*
Jan 6 05:00:39 63.70.211.209:4713 -> a.b.c.244:515 SYN ******S*
Jan 6 05:00:41 63.70.211.209:4555 -> a.b.c.101:515 SYN ******S*

Jan 6 05:00:42 63.70.211.209:4568 -> a.b.c.114:515 SYN ******S*

Jan 6 05:00:42 63.70.211.209:4575 -> a.b.c.121:515 SYN ******S*

Jan 6 05:00:42 63.70.211.209:4776 -> a.b.d.52:515 SYN ******S*

Jan 6 05:00:43 63.70.211.209:4929 -> a.b.d.202:515 SYN ******S*

Jan 6 05:00:45 63.70.211.209:4972 -> a.b.d.245:515 SYN ******S*

Jan 6 05:00:45 63.70.211.209:4973 -> a.b.d.246:515 SYN ******S*

Jan 6 05:00:45 63.70.211.209:4976 -> a.b.d.249:515 SYN ******S*

=-=-=-=-=-=-=-=-=-=-=

Defensive Recommendation
These alerts appear to be a scan for LPD services, so it would be a good idea to have a look at the local machines. If nothing else, start a packet capture to find out what's going on. The machines may or may not be compromised, but it would be worth a look to find out if LPD services are available to the Internet. As well, the firewall rulesets should be reviewed. Any traffic to and from TCP port 515 to the Internet should be blocked.

External RPC Call

External RPC Call alerts can be generated when a remote host attempts to connect to an internal host on a port that may be used for Remote Procedure Calls. This alert focuses on attempts to TCP port 111. Access from the Internet to RPC ports should not be allowed unless under very controlled conditions, and then monitored carefully. Most probes or connection attempts from the Internet to RPC ports are with the intent to compromise a system (see http://www.sans.org/infosecFAQ/cmsd.htm for an example). Ports 111 and 32771 are favorite targets for attackers. Another good RPC discussion is here: http://www.sans.org/newlook/resources/IDFAQ/trouble_RPCs.htm

There were 265 External RPC Call events generated from 5 remote hosts to 231 internal hosts.

The top five source addresses:

Alerts	IP Address	Host
81	61.129.39.161	Does not resolve
66	195.127.111.251	raq3.fra.awic.net
44	209.70.72.22	Does not resolve
43	210.158.26.57	Does not resolve
31	210.117.174.31	Does not resolve

The top five destination addresses:

Alerts	IP Address	Port
4	MY.NET.135.175	111
2	MY.NET.132.101	111
2	MY.NET.133.102	111
2	MY.NET.132.109	111
2	MY.NET.132.110	111

Correlation
http://www.sans.org/y2k/010900.htm
Hi,
more from S. Korea:

```
Jan 8 14:46:06 quark tcplogd: sunrpc connection attempt from root@[211.46.144.2
Jan 8 14:46:08 aeon tcplogd: sunrpc connection attempt from root@[211.46.144.2
```

http://www.sans.org/y2k/071000.htm
(Laurie@.edu)
=-=-=-=-=-=-=-=-=-=
Brazilian Research Network
```
Jul 5 14:47:23 dns3 snort[15604]: RPC Info Query:
  200.196.84.120:2790 -> z.y.w.98:111
```

Defensive Recommendation
These alerts appear to be a scan for RPC services, so it would be a good idea to have a look at the local machines. If nothing else, start a packet capture to find out what's going on. The machine may or may not be compromised, but it would be worth a look to find out if RPC services are available to the Internet. As well, the firewall rulesets should be reviewed. Any unnecessary traffic to and from the internal network should be blocked.

SMB Name Wildcard
SMB Wildcard alerts are generated when a query by IP address is received for NetBIOS information (nbtstat –a <address>). If the alerts are generated from a source address on the Internet, this is most likely a remote system attempting system reconnaissance. Please refer to http://www.robertgraham.com/pubs/firewall-seen.html#10 and http://www.microsoft.com/technet for details.

There were 182 SMB Name Wildcard events generated from 102 source hosts to 77 internal hosts.

The top 5 source addresses:

Alerts	IP Address	Host
6	211.118.86.11	Does not resolve
6	4.41.3.11	Does not resolve
6	217.1.75.169	pd9014ba9.dip.t-dialin.net
5	10.0.0.140	IANA Reserved Address Block
5	211.45.211.6	Does not resolve

Correlation
http://www.sans.org/y2k/practical/JoanneTreurniet.html
SMB Name Wildcard
This is a very odd scan of 137 from within our own network, always to the same hosts. This warrants some investigation, in particular the placement of the IDS with respect to these hosts. MY.NET.101.192 is also involved in the SNMP public access mystery (above), for which the times interleave with the SMB name wildcard alerts.

06/30-09:27:47.890735 MY.NET.101.160:137 -> MY.NET.101.192:137
06/30-09:29:05.193413 MY.NET.101.160:137 -> MY.NET.101.192:137
06/30-09:29:06.691558 MY.NET.101.160:137 -> MY.NET.101.192:137
07/11-14:23:50.181921 MY.NET.70.66:137 -> MY.NET.101.116:137
07/11-15:56:56.690460 MY.NET.101.160:137 -> MY.NET.101.192:137
07/11-15:56:58.079944 MY.NET.101.160:137 -> MY.NET.101.192:137
07/11-16:00:19.379699 MY.NET.101.160:137 -> MY.NET.101.192:137
07/11-16:00:20.866723 MY.NET.101.160:137 -> MY.NET.101.192:137
07/12-16:36:47.188910 MY.NET.70.66:137 -> MY.NET.101.55:137
07/12-17:20:45.412157 MY.NET.70.66:137 -> MY.NET.101.117:137
07/12-18:17:34.633869 MY.NET.101.160:137 -> MY.NET.101.192:137
07/12-18:32:11.853647 MY.NET.101.160:137 -> MY.NET.101.192:137
07/12-18:32:14.856611 MY.NET.101.160:137 -> MY.NET.101.192:137
07/12-18:35:30.556771 MY.NET.101.160:137 -> MY.NET.101.192:137
07/12-18:35:30.571607 MY.NET.101.160:137 -> MY.NET.101.192:137
07/12-18:46:18.501496 MY.NET.101.160:137 -> MY.NET.101.192:137
07/12-18:46:19.960384 MY.NET.101.160:137 -> MY.NET.101.192:137
07/12-19:00:29.220109 MY.NET.101.160:137 -> MY.NET.101.192:137
07/14-08:13:21.447045 MY.NET.101.160:137 -> MY.NET.101.192:137
07/14-08:13:22.972607 MY.NET.101.160:137 -> MY.NET.101.192:137

Since they both use port 137 (source and destination), MY.NET.101.160 is a Windows machine (Firewall FAQ). This type of traffic is normal on the Internet, but the odd thing here is that the source and destination are on the same network, and it shouldn't get to an IDS at the border.

http://www.sans.org/y2k/practical/Joseph_Rach.html
05/28-13:18:26.406462 [**] SMB Name Wildcard [**] MY.NET.101.160:137 -> MY.NET.101.192:137

Defensive Recommendation
Allowing NetBIOS traffic to and from the Internet is not a recommended security practice, nor is it considered good network managment. NetBIOS is inherently insecure and can reveal much about a network. Block all outbound NetBIOS traffic at the firewall or the border router. Block all inbound NetBIOS traffic at the border router.

Queso Fingerprint

Queso is a port-scanning tool designed to fingerprint operating systems. It relies on standard responses to TCP packets with seven combinations of flags set in the header. The response returned to each type of packet will determine the operating system.
For details, please refer to:
http://www.securityfocus.com/focus/ids/articles/portscan.html
http://www.sans.org/newlook/resources/IDFAQ/TCP_fingerprinting.htm

There were 132 Queso fingerprint events generated from 10 remote hosts to 20 internal hosts. As well, 113 of the 132 Queso fingerprint alerts were generated for TCP port 6346.

The top five source addresses:

Alerts	IP Address	Host
99	120.206.170.20	Does not resolve
14	158.75.57.4	Does not resolve
10	130.233.26.197	jt11jm-res-5.tky.hut.fi
2	128.46.156.117	csociety-ftp.ecn.purdue.edu
2	66.1.65.32	cpe-66-1-65-32.az.sprintbbd.net

The top five destination addresses:

Alerts	IP Address	Port
98	MY.NET.202.54	6346
10	MY.NET.219.134	various
2	MY.NET.60.38	2287, 2305
2	MY.NET.98.189	6346
2	MY.NET.226.146	6346

Correlation
http://www.sans.org/y2k/practical/david_whyte_gcia.zip
```
09/05-09:00:54.631991  [**] Queso fingerprint [**] 24.3.161.193:32814 -> MY.NET.145.9:110
09/05-09:00:54.631991  [**] Queso fingerprint [**] 24.3.161.193:32814 -> MY.NET.145.9:110
09/07-19:27:42.236314  [**] Queso fingerprint [**] 64.80.63.121:4114 ->
MY.NET.204.214:6355
09/07-19:27:42.236314  [**] Queso fingerprint [**] 64.80.63.121:4114 ->
MY.NET.204.214:6355
```

http://www.sans.org/y2k/100200.htm
(Robin)
```
09/29-00:22:54.391105 [**] spp_portscan: PORTSCAN DETECTED from 24.3.161.193 (STEALTH)
[**]
09/29-00:11:58.584512 [**] Queso fingerprint [**] 24.3.161.193:32811 -> MY.NET.145.9:110
09/29-00:22:56.486852 [**] spp_portscan: portscan status from 24.3.161.193: 1 connections
across 1 hosts: TCP(1), UDP(0) STEALTH [**]
```

```
09/29-00:22:59.244973 [**] spp_portscan: End of portscan from 24.3.161.193 (TOTAL HOSTS:1
TCP:1 UDP:0) [**]
```

Defensive Recommendation
It's possible that the top talkers are generating false positives. All of the alerts generated from events directed towards TCP port 6346 occurred over a long period of time. OS fingerprinting of one system probably wouldn't take 99 attempts spread out over several hours. The same could be true of some the other top talkers as well. It may be necessary to review and test the Snort ruleset, as well as monitor the traffic from the remaining source addresses. Since TCP port 6346 is commonly associated with Gnutella, it might also be a good idea to have a look at the local machines.

Back Orifice

Quote the CDC: "Back Orifice is a remote administration tool allowing a user to control a computer over a TCP/IP connection using a simple console or GUI application…BO gives its user more control of the remote Windows machine than the person at the keyboard of the remote machine has." (http://www.cultdeadcow.com/tools/bo.html).
Back Orifice is well developed, written in C++, and runs on all varieties of Windows machines. It's known to spread via email attachments or other "trojanized" software. Once installed, the BO servers listen on ports 31337, 31338, 54320, and 54321. One mark of the success of the Trojan is the continuous scanning by remote hosts looking for already compromised systems.
For details about Back Orifice refer to:
http://www.cert.org/vul_notes/VN-98.07.backorifice.html
http://www.sans.org/infosecFAQ/agents.htm
http://www.sans.org/infosecFAQ/back_orifice.htm
http://www.cultdeadcow.com/tools/bo.html
http://www.bo2k.com/
http://www.symantec.com/avcenter/warn/backorifice.html
http://www.nsclean.com/psc-bo2k.html

There were 132 BackOrifice events recorded from 2 remote hosts to 132 internal hosts.

The two source addresses:

Alerts	IP Address	Host
109	24.162.245.198	rdu162-245-198.nc.rr.com
23	4.54.130.215	pppa90-resalephiladelphiametro2-2r7193.dialinx.net

The destination addresses:

MY.NET.21.102	MY.NET.4.191	MY.NET.7.39	MY.NET.17.25	MY.NET.15.67
MY.NET.12.150	MY.NET.13.130	MY.NET.210.51	MY.NET.210.202	MY.NET.10.57
MY.NET.1.113	MY.NET.15.77	MY.NET.18.50	MY.NET.210.61	MY.NET.210.215
MY.NET.210.218	MY.NET.210.147	MY.NET.21.60	MY.NET.17.6	MY.NET.21.197
MY.NET.17.8	MY.NET.5.115	MY.NET.5.117	MY.NET.7.53	MY.NET.7.57
MY.NET.20.6	MY.NET.9.110	MY.NET.2.105	MY.NET.1.127	MY.NET.17.49
MY.NET.6.179	MY.NET.20.54	MY.NET.210.155	MY.NET.13.54	MY.NET.6.101
MY.NET.15.16	MY.NET.6.105	MY.NET.4.88	MY.NET.7.67	MY.NET.18.137
MY.NET.20.156	MY.NET.4.11	MY.NET.11.24	MY.NET.7.234	MY.NET.4.13
MY.NET.210.231	MY.NET.11.28	MY.NET.21.162	MY.NET.15.125	MY.NET.15.20
MY.NET.5.132	MY.NET.210.11	MY.NET.5.138	MY.NET.12.114	MY.NET.12.114
MY.NET.9.32	MY.NET.6.119	MY.NET.20.160	MY.NET.9.34	MY.NET.210.17
MY.NET.13.70	MY.NET.21.96	MY.NET.6.67	MY.NET.9.2	MY.NET.10.166
MY.NET.2.76	MY.NET.5.51	MY.NET.17.102	MY.NET.210.187	MY.NET.20.102
MY.NET.20.104	MY.NET.20.106	MY.NET.7.110	MY.NET.20.108	MY.NET.7.116
MY.NET.210.35	MY.NET.7.118	MY.NET.12.136	MY.NET.9.54	MY.NET.21.115

MY.NET.21.150	MY.NET.210.105	MY.NET.21.152	MY.NET.13.108	MY.NET.210.107
MY.NET.2.121	MY.NET.20.166	MY.NET.9.201	MY.NET.20.168	MY.NET.18.83
MY.NET.7.95	MY.NET.18.164	MY.NET.13.118	MY.NET.1.161	MY.NET.7.98
MY.NET.4.101	MY.NET.2.143	MY.NET.1.167	MY.NET.9.224	MY.NET.20.90
MY.NET.17.118	MY.NET.7.120	MY.NET.17.16	MY.NET.12.142	MY.NET.210.120
MY.NET.9.62	MY.NET.210.2	MY.NET.21.243	MY.NET.210.125	MY.NET.12.148
MY.NET.21.42	MY.NET.18.178	MY.NET.2.157	MY.NET.6.17	MY.NET.4.117
MY.NET.7.21	MY.NET.12.74	MY.NET.210.195	MY.NET.20.96	MY.NET.210.199
MY.NET.21.50	MY.NET.5.6	MY.NET.10.125	MY.NET.20.130	MY.NET.15.71
MY.NET.17.120	MY.NET.5.71	MY.NET.9.167	MY.NET.17.122	

Correlation

http://www.sans.org/y2k/101800-1230.htm

Laurie@.edu

> Server used for this query: [whois.arin.net]
> UUNET Technologies, Inc. (NETBLK-NETBLK-UUNET97DU)
> 3060 Williams Drive, Suite 601 Fairfax, va 22031 US
> Netname: NETBLK-UUNET97DU
> Netblock: 63.0.0.0 - 63.53.255.255
> Maintainer: UUDA

```
Oct 16 22:55:17 hostre portsentry[425]: attackalert: Connect from host:
  1Cust194.tnt1.sierra-vista.az.da.uu.net/63.11.161.194 to UDP port: 31337
Oct 16 22:55:17 hostp portsentry[544]: attackalert: Connect from host:
  1Cust194.tnt1.sierra-vista.az.da.uu.net/63.11.161.194 to UDP port: 31337
Oct 16 22:55:17 hostp portsentry[544]: attackalert: Connect from host:
  1Cust194.tnt1.sierra-vista.az.da.uu.net/63.11.161.194 to UDP port: 31337
Oct 16 22:55:18 hostbe portsentry[26280]: attackalert: Connect from host:
  1Cust194.tnt1.sierra-vista.az.da.uu.net/63.11.161.194 to UDP port: 31337
Oct 16 22:56:34 hostca portsentry[265]: attackalert: Connect from host:
  1Cust194.tnt1.sierra-vista.az.da.uu.net/63.11.161.194 to UDP port: 31337
Oct 16 22:56:34 hostca portsentry[265]: attackalert: Connect from host:
  1Cust194.tnt1.sierra-vista.az.da.uu.net/63.11.161.194 to UDP port: 31337
Oct 16 22:56:34 hostdo portsentry[497]: attackalert: Connect from host:
  1Cust194.tnt1.sierra-vista.az.da.uu.net/63.11.161.194 to UDP port: 31337
Oct 16 22:55:37 hostcr portsentry[18062]: attackalert: Connect from host:
  1Cust194.tnt1.sierra-vista.az.da.uu.net/63.11.161.194 to UDP port: 31337
Oct 16 22:56:38 hostba portsentry[1114]: attackalert: Connect from host:
  1Cust194.tnt1.sierra-vista.az.da.uu.net/63.11.161.194 to UDP port: 31337
```

http://www.sans.org/y2k/020501-1400.htm

" …Jan 29 15:31:05 128.187.252.215:4510 -> a.b.c.15:31337 SYN ******S*

Jan 29 15:31:05 128.187.252.215:4540 -> a.b.c.30:31337 SYN ******S*

Jan 29 15:31:05 128.187.252.215:4543 -> a.b.c.32:31337 SYN ******S*

Jan 29 15:31:06 128.187.252.215:4642 -> a.b.c.62:31337 SYN ******S*

Jan 29 15:31:06 128.187.252.215:4655 -> a.b.c.71:31337 SYN ******S*

Jan 29 15:31:06 128.187.252.215:4667 -> a.b.c.80:31337 SYN ******S*

Jan 29 15:31:07 128.187.252.215:4812 -> a.b.c.101:31337 SYN ******S*"

Defensive Recommendation

It's difficult to tell whether all or any of the internal hosts are compromised. The best course of action may be to cut off the source addresses at the border router or the firewall, and then scan the internal network for any machine that listens on TCP ports 31337, 31337, 31338, 54320, and 54321. The internal machines listed in the table above should be closely inspected. Review the acceptable use policy for references to remote administration tools. If the BackOrifice server is found installed on any local machines, there are a number of tools created to remove the software. One is located here: http://packetstorm.securify.com/trojans/bo/; another here

http://students.depaul.edu/~iestrada/ and here
http://www.nwinternet.com/~pchelp/bo/removingBO.htm.

WinGate 1080 Attempt

Connection attempts to TCP port 1080 usually mean a scan for open SOCKS ports. The SOCKS protocol allows users to proxy outbound service requests through a single local machine. The most common SOCKS protocol software is Wingate, used to allow multiple computers to simultaneously share a single Internet connection (http://wingate.deerfield.com/).
It is notoriously insecure (http://www.sans.org/newlook/resources/IDFAQ/socks.htm).
Please refer also to CVE-1999-0290, CVE-1999-0291, CVE-1999-0441, CVE-1999-0494, CAN-1999-0657, and CAN-2000-1048 for details.

There were 87 WinGate 1080 Attempt events recorded from 43 remote hosts to 47 internal hosts.

The top five source addresses:

Alerts	IP Address	Host
8	207.117.70.5	Does not resolve
8	195.66.170.8	irc.cg.yu
5	216.54.223.198	Does Not Resolve
4	192.216.128.28	protege.linkline.com
4	213.151.16.249	Does not resolve

The top five destination addresses:

Alerts	Host	Port
5	MY.NET.60.8	1080
5	MY.NET.204.102	1080
5	MY.NET.60.11	1080
4	MY.NET.254.10	1080
4	MY.NET.222.10	1080

Correlation
http://www.sans.org/y2k/091400.htm
http://www.sans.org/y2k/practical/JoanneTreurniet.html

06/27-12:16:30.590717 l245ppp151.ksc.net.th:1166 -> MY.NET.97.159:1080
06/27-12:16:31.795642 l245ppp151.ksc.net.th:1166 -> MY.NET.97.159:1080
06/27-12:16:34.149941 l245ppp151.ksc.net.th:1166 -> MY.NET.97.159:1080

http://www.sans.org/y2k/practical/Dale_Ross_GCIA.htm

```
09/07-16:47:09.626314 [**] WinGate 1080 Attempt [**] 151.17.144.96:4401 ->
MY.NET.100.2:1080
09/07-16:47:10.843105 [**] WinGate 1080 Attempt [**] 151.17.144.96:4425 ->
MY.NET.100.2:1080
09/07-16:47:32.494655 [**] WinGate 1080 Attempt [**] 151.17.144.96:4593 ->
MY.NET.100.2:1080
09/07-16:47:37.959851 [**] WinGate 1080 Attempt [**] 151.17.144.96:4641 ->
MY.NET.100.2:1080
09/07-16:47:52.269249 [**] WinGate 1080 Attempt [**] 151.17.144.96:4761 ->
MY.NET.100.2:1080
```

Defensive Recommendation
I would be a good idea to scan the internal network for servers listening on port 1080, as well as inspect the local machines. Review the acceptable use policy for references to proxy services. If possible, disable any WinGate/SOCKS proxy service. External connections to TCP port 1080 should be blocked at the border router or firewall.

Russia Dynamo – SANS Flash 28-jul-00

The Snort ruleset that generated this alert was created in response to notes in the SANS Handlers Diary of 7.29.00 (http://www.sans.org/y2k/072818.htm), and is meant as a general alert on traffic bound to or from the dol.ru network at port 8080.

There were 41 Russia Dynamo alerts from one remote host to one internal host:

Alerts	IP Address	Host
40	194.87.6.189	189.6.87.194.dynamic.dol.ru
1	MY.NET.178.42	Internal

The destination addresses:

Alerts	Host	Port
40	MY.NET.178.42	317
1	194.87.6.189	various

Correlation
http://www.sans.org/y2k/072818.htm
http://www.sans.org/y2k/practical/Crist_Clark_GCIA.html

Defensive Recommendation
Shut off the source at the border router or the firewall. It might also be a good idea to filter out the entire 194.87 address block. TCP port 317 is attributed to ZanNet (http://www.zannet.com), a remote administration tool for Windows 95/98. As well, MY.NET.178.42 may be compromised. Review the acceptable use policy for references to ZanNet and have a look at the local machine.

Tiny Fragments - Possible Hostile Activity

Tiny fragments are just that: IP packets split into segments smaller than the RFC's suggest. Tiny fragments are used as a firewall penetration technique, as a way to evade some IDS's or packet filters, for DoS attacks, or as a scan. Please refer to RFC 1858 (http://www.ietf.org/rfc/rfc1858.txt) for details.

There were 27 Tiny Fragments events generated from 2 remote hosts to 12 internal hosts:

Alerts	IP Address	Host
14	209.39.78.125	125.texasgraphics.com
13	64.168.21.140	adsl-64-168-21-140.dsl.snfc21.pacbell.net

The top five destination addresses:

Alerts	Host
13	MY.NET.203.198
3	MY.NET.212.106
2	MY.NET.208.142
1	MY.NET.218.178
1	MY.NET.224.130

Correlation
http://www.sans.org/y2k/060600-1930.htm
http://www.sans.org/y2k/practical/Guy_Bruneau.doc
[**] Tiny Fragments - Possible Hostile Activity [**]

There was some traffic in "tiny fragments", which means that TCP traffic was fragmented smaller than is normally done by operating systems or network (less than 128 KB). The source hosts detected were host01.quo.jsv.qwest.net (63.236.34.174) with 6 hits on 28

June 2000, **adsl-61-144-55.mia.bellsouth.net (208.61.144.55)** with a single hit on 11 July 2000, and 202.76.177.204 (from Australia) with two hits on 26 July 2000

http://www.sans.org/y2k/practical/JoanneTreurniet.html
Tiny Fragments - Possible Hostile Activity

06/28-06:35:13.540772 host01.quo.jsv.qwest.net: -> MY.NET.1.8:
06/28-06:35:13.540827 host01.quo.jsv.qwest.net: -> MY.NET.1.8:
06/28-06:35:13.540878 host01.quo.jsv.qwest.net: -> MY.NET.1.8:
06/28-06:37:13.538078 host01.quo.jsv.qwest.net: -> MY.NET.1.8:
06/28-06:37:13.538175 host01.quo.jsv.qwest.net: -> MY.NET.1.8:
06/28-06:37:13.538272 host01.quo.jsv.qwest.net: -> MY.NET.1.8:
07/11-03:33:54.281367 adsl-61-144-55.mia.bellsouth.net: -> MY.NET.230.241:
07/26-11:05:01.522342 202.76.177.204: -> MY.NET.70.20:
07/26-13:54:29.666358 202.76.177.204: -> MY.NET.70.20:

Defensive Recommendation
There are a number of reasons to see tiny fragments on a network. For instance, attempting to bypass network security by sending packets of an unusually small size. Unfortunately, that means someone is up to no good. It would be a good idea to start a packet capture on traffic originating from the two remote sources, and to continue monitoring for these events. Keep track of the senders to compare to future alerts. It might also be a good idea to cut off the source addresses at the border router or the firewall.

NMAP TCP Ping!

Nmap TCP Ping! is an OS fingerprinting technique very similar to Queso. Nmap is a port-scanning tool capable of stealth and OS fingerprinting, among other things. The TCP ping option can be selected with the –PT switch (http://www.insecure.org/nmap/nmap_manpage.html).

There were 26 Nmap TCP Ping! events generated from 8 remote hosts to 11 internal hosts. The source addresses generated alerts for only two destination ports:

Alerts	Destination Port	Service
23	53	DNS
3	80	HTTP

The top five source addresses:

Alerts	IP Address	Host
9	192.102.197.234	geo197a.cps.intel.com
7	194.133.58.2	bestroute1-t.alcatel.fr
3	63.119.91.2	Does Not Resolve
2	199.197.130.21	Does Not Resolve
2	194.133.58.129	bestroute2-t.alcatel.fr

The top five destination addresses:

Alerts	Host	Port
6	MY.NET.1.8	53
4	MY.NET.1.3	53
4	MY.NET.1.5	53
3	MY.NET.1.10	53
3	MY.NET.253.125	80

Correlation
http://www.sans.org/y2k/practical/Wei-Chieh_Lim_GCIA.doc
[**] IDS028 - PING NMAP TCP [**]
11/07-09:58:26.646757 192.102.197.234:53 -> server1.my.net:53

TCP TTL:47 TOS:0x0 ID:43478
*****A**** Seq: 0x356 Ack: 0x0 Win: 0x578**

[**] IDS028 - PING NMAP TCP [**]
11/07-09:58:26.599733 192.102.197.234:80 -> server1.my.net:53
TCP TTL:47 TOS:0x0 ID:43476
*****A**** Seq: 0x355 Ack: 0x0 Win: 0x578**

Our firewall, *server1.my.net* queries the intel.com nameservers for www.intel.com and gets the answer:

> **Authoritative answers can be found from:**
> **geo164a.cps.intel.com internet address = 192.198.164.170**
> **geo197a.cps.intel.com internet address = 192.102.197.234**

server1.my.net then queries 192.102.197.234 for the IP address for www.intel.com and receives an answer.

http://www.sans.org/y2k/091700.htm
http://archives.neohapsis.com/archives/snort/2000-08/0040.html

Defensive Recommendation
The correlations point to these events as false positives. It appears the majority of the alerts might have been generated by DNS load balancing. To be sure, it might be a good idea to sniff the traffic bound for the destination addresses. If the traffic indeed turns out to be DNS, adjust the Snort ruleset accordingly. If not, the source addresses should be blocked at the firewall or the border router.

Null scan!
A null scan is a stealth/attack port scanning technique marked by the lack code bits or flags in the TCP header. Any TCP packet of this sort seen on the network should be considered suspicious. It's possible to create such packets using tools like nmap (http://www.insecure.org/nmap/nmap_manpage.html), but poorly written software or faulty networking equipment has been known to create them as well.

There were 23 Null Scan! events generated from 19 remote hosts and 7 internal hosts. Both the source and destination addresses generated alerts for 31 different destination ports, including one to port 0. The top four destination ports were:

Alerts	Destination Port	Service
10	6346	Gnutella
4	6699	Napster
2	29515	Unknown
2	6688	Napster

The Top 5 source addresses:

Alerts	IP Address	Host
4	24.43.241.223	cr937687-a.bawk1.on.wave.home.com
2	24.17.64.12	c117002-a.roalok1.mi.home.com
1	203.54.81.61	ess-p-203-54-81-61.mega.tmns.net.au
1	213.65.88.221	h221n1fls31o834.telia.com
1	209.221.200.17	host17.qnet.com

The Top 5 destination addresses:

Alerts	Host	Port
4	MY.NET.209.30	6346
2	MY.NET.209.154	76

1	MY.NET.220.214	6346
1	MY.NET.97.29	39
1	MY.NET.212.182	6699

Correlation
There are 48 NULL Scans recorded in the scan logs.
Mar 25 05:58:00 194.109.233.100:1109 -> MY.NET.219.30:6346 NULL ********
Mar 25 06:16:24 212.199.104.98:25128 -> MY.NET.222.154:4969 NULL ********
Mar 25 10:26:14 202.77.194.196:0 -> MY.NET.20.10:0 NULL ********
Mar 25 16:43:38 62.254.145.163:6346 -> MY.NET.220.214:1164 NULL ********
Mar 26 06:39:33 24.43.241.223:4148 -> MY.NET.209.30:6346 NULL ********
Mar 26 06:43:06 24.43.241.223:4148 -> MY.NET.209.30:6346 NULL ********
Mar 26 06:44:00 24.43.241.223:4148 -> MY.NET.209.30:6346 NULL ********
http://www.sans.org/y2k/practical/Teri_Bidwell_GCIA.doc

Defensive Recommendation
The majority of the scans appear to be looking for an open Napster or Gnutella port, while the remainder could be regular scans of the internal network. Any machines on the internal network allowing connections to any of the documented ports should be given some scrutiny; especially those connecting to TCP ports 6346, 6688, and 6699. If access doesn't fall under the acceptable use policy, connections to these ports should be blocked. It might also be a good idea to start a packet capture just to find out what's going on.
As well, a review of current security policy for references to Napster and Gnutella is advised. While neither is a destructive application, it does affect productivity and can have an adverse effect on network bandwidth and system storage. Just shutting it off at the router or firewall might be the best course of action.
For more information see http://www.napster.com and http://gnutella.wego.com.

Port 55850 TCP - Possible MyServer Activity - Ref. 010313-1

MyServer is an obscure DDOS tool discovered on some university servers late last summer. The agent works by spoofing its source address and flooding a third target with FIN packets to TCP port 113. It seems to be particular to Linux machines, and is installed through known exploits. The agent has been observed to bind to UDP 55850 (or TCP 55850, depending on who you ask) to listen for control commands.

There were 13 Port 55850 TCP – Possible MyServer Activity events generated by five internal and remote hosts:

Alerts	IP Address	Host
5	MY.NET.253.24	Internal
3	198.81.129.194	relay2.ucia.gov
2	MY.NET.100.230	Internal
2	137.118.131.7	mail.fmtc.com
1	204.68.24.61	aw161.netaddress.usa.net

Correlation
None.

Defensive Recommendation
The traffic recorded in the alerts appears to be legitimate, but it's very had to tell without more information. The best course of action would be to inspect the local machines for evidence of tampering with ls, ps, and netstat. MyServer DDoS tools have been observed to be installed in /lib. As well, servers compromised with MyServer have been observed to probe other networks at TCP ports 21 and 111, although neither of the local machines generated any other alerts. It might

also be a good idea to update the Snort ruleset to alert on traffic to UDP port 55850, and ensure that ingress and egress filtering is configured on the border router.
Please refer to:
http://www.securityfocus.com/archive/75/140891
http://www.incidents.org/archives/y2k/082200.htm

SUNRPC Highport Access!

Sun RPC HPA alerts can be generated when a remote host attempts to connect to an internal host on a port that may be used for Remote Procedure Calls. The port is typically numbered above 32000. Most probes or connection attempts from the Internet to RPC ports are with the intent to compromise a system (see http://www.sans.org/infosecFAQ/cmsd.htm for an example). Ports 111 and 32771 are favorite targets for attackers.

There was ten SUNRPC HPA events generated by one remote host to one internal host.

SnortSnarf synopses:

03/25-14:58:28.952796 [**] SUNRPC highport access! [**] 216.136.171.195:3778 -> MY.NET.100.225:32771
03/25-14:58:28.953834 [**] SUNRPC highport access! [**] 216.136.171.195:3778 -> MY.NET.100.225:32771

Correlation
http://www.sans.org/y2k/021500.htm

Defensive Recommendation
Access from the Internet to RPC ports should not be allowed unless under very controlled conditions, and then monitored carefully. It would be a good idea to have a look at the local machine. If nothing else, start a packet capture to find out what's going on. The machine could be compromised, but it would be worth a look to find out if RPC services are available to the Internet. As well, the firewall rulesets should be reviewed. Any unnecessary traffic to and from the internal network should be blocked.

Connect To 515 From Inside

Port 515 is commonly associated with the Line Printer Daemon (LPD) in Unix. There are known exploits for this port that can allow an attacker to gain root access. Refer to http://www.cert.org/advisories/CA-2000-22.html for details.

This is the only instance of this event.
SnortSnarf synopses:
03/29-15:38:30.344890 [**] connect to 515 from inside [**] MY.NET.60.11:713 -> 198.81.209.16:515

Correlation
http://www.sans.org/y2k/110800-0900.htm
http://www.cert.org/advisories/CA-2000-22.html
http://www.sans.org/newlook/alerts/port515.htm

Defensive Recommendation
It would be a good idea to have a look at the local machine. If nothing else, start a packet capture to find out what's going on. The machine could be compromised, but it would be worth a look to find out if LPD services are available to the Internet. As well, the firewall rulesets should be reviewed. Any unnecessary traffic to and from the internal network should be blocked.

STATDX UDP Attack

The STATDX attack targets Unix systems running the rpc.statd daemon. This is the only instance of this attack.

SnortSnarf synopses:

03/27-13:53:15.045609 [**] STATDX UDP attack [**] 210.117.174.31:1411 -> MY.NET.6.15:32776

Correlation

http://www.sans.org/y2k/120600-1200.htm
http://www.sans.org/y2k/practical/Joseph_Rach.html#DETECT2

Defensive Recommendation

It would be a good idea to have a look at the local machine. If nothing else, start a packet capture to find out what's going on. The machine could be compromised, but it would be worth a look to find out if RPC services are available to the Internet. As well, the firewall rulesets should be reviewed. Any unnecessary traffic to and from the internal network should be blocked.

Scans

SCAN	COUNT
UDP	163324
SYN	64435
SYNFIN	1436
NOACK	72
INVALIDACK	81
UNKNOWN	38
FIN	13
NULL	48
VECNA	19
FULLXMAS	5
XMAS	3
SPAU	5
NMAPID	6

Stats:

Unique Scans detected	2301
Unique Source Hosts	391
Unique Destination Hosts	49532

Top five external scanners

Alerts	IP Address	Host
149	212.144.16.169	16-169.E.dial.o-tel-o.net
123	203.89.246.250	Does Not Resolve
97	129.206.170.20	jupiter.wh.uni-heidelberg.de
33	157.158.46.39	thales.ogi.polsl.gliwice.pl
30	61.11.252.117	Does Not Resolve

Of the top five external scanners:

212.144.16.169

Began SYN scanning 19,589 addresses on the MY.NET network beginning 00:15:55, and ending 05:18:24 on 3.25.01, all to TCP port 21 (FTP).
212.144.16.169 did not generate any other alerts.

203.89.246.250
Began SYN scanning 4836 addresses on the MY.NET network beginning 00:38:53, and ending 02:05:29 on 3.26.01, all to TCP port 25 (SMTP).
203.89.246.250 did not generate any other alerts.

129.206.170.20
Made 97 connection attempts to MY.NET 202.54 at random times beginning 12:09:06, and ending 19:25:02 on 3.25.01, all to TCP port 6346 (Gnutella).
129.206.170.20 did not generate any other alerts.

157.158.46.39
Began SYN scanning 1629 addresses on the MY.NET network beginning 15:56:20, and ending 16:2124 on 3.28.01, all to TCP port 1111 (LMSocialServer).
157.158.46.39 did not generate any other alerts.

61.11.252.117
Began a SYN-FIN scan of 1441 addresses on the MY.NET network beginning 20:02:01, and ending 20:30:35 on 3.29.01, all to TCP port 21.
61.11.252.117 did not generate any other alerts.

Internal Scanners
Particular attention should be paid to scanning internal hosts. Scans from internal machines can mean possible compromise, curious users, or that your IDS ruleset needs adjustment. It's always a good idea to investigate scanning internal hosts.

Top five external scanners

Alerts	Host
109	MY.NET.227.194
87	MY.NET.218.86
63	MY.NET.202.34
62	MY.NET.140.21
53	MY.NET.224.130

Of the top five internal scanners:
MY.NET.227.194
Over the period of 3.26.01 22:14:56 to 3.29.01 18:42:54 MY.NET.227.194 made or attempted to make UDP connections to various hosts on the Internet at well-known Trojan ports. The common source and destination port is 13139. Other ports of interest are 80, 666, 1025, 1045, 4000, 6666, 10000, 11000, 12344, 20000, 24000, 65534, and 65535. Refer to http://www.simovits.com/nyheter9902.html for details on these ports.
The most interesting period ranged from 22:40:21 to 23:21:35 on 3.26, 07:00:32 to 23:56:46 on 3.27 at regular intervals, and 09:50:35 to 18:42:22 on 3.28 at regular intervals. During these periods MY.NET 227.194 attempted to connect from port 13139 to 377 unique Internet addresses at port 13139.
It's possible that MY.NET.227.194 is compromised and warrants investigation.
MY.NET.218.86
Began SYN scanning 1071 addresses on the Internet beginning 01:10:45, and ending 12:58:00 on 3.25.01, all to TCP port 6346 (Gnutella).
MY.NET.218.86did not generate any other alerts.

MY.NET.202.34
Attempted UDP connections to ports 7777, 7778, 10000, and a few other well-known trojan ports to 289 unique Internet addresses beginning 13:03:26 on 3.25.01, and ending 20:09:57 on 3.29.01.

It’s possible that MY.NET.202.34 is compromised and warrants investigation.

MY.NET.140.21
Attempted UDP connections from port 7001 to alternating ports 7000 and 7003 to 29 unique Internet addresses beginning 13:03:26 on 3.25.01, and ending 23:14:05 on 3.29.01.
It’s possible that MY.NET.140.21 is compromised and warrants investigation.

MY.NET.224.130
Attempted UDP connections from various ports to alternating ports 13139 and 27050, as well as a handful of other well-known trojan ports to 289 unique Internet addresses beginning 15:54:41 on 3.25.01, and ending 16:12:51 on 3.29.01.
It’s possible that MY.NET.224.130 is compromised and warrants investigation.

Out of Spec

Out Of Specification (OOS) alerts represent anomalous, strange, don’t-fit-any-other-category traffic. Crafted packets created by various scanning tools (in attempts to evade intrusion detections systems; or used for network and/or OS fingerprinting) can generate OOS alerts. Faulty networking equipment or poorly written software has also been known to create these kinds of packets. Whatever the case, the packets are marked by inconsistencies in the flag field of the TCP header. Since the packets that generate these alerts wouldn’t normally occur, the hosts responsible for OOS entries should bear a little scrutiny.

Of the 6239 total OOS entries recorded in the five day analysis period, two external hosts

Alerts	Source IP	Host
3871	61.11.252.117	Does Not Resolve
2033	211.178.63.4	Does Not Resolve

were responsible for 94%, or 5904, of the OOS alerts. All of these alerts are SYN-FIN scans to almost each and every host on the MY.NET network, with the scans running across ports 21, 53, 109, and 8080; mostly to port 21. The packets from both hosts were crafted as evidenced by the uniform packet ID in the TCP header:

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/30-12:08:13.519728 211.178.63.4:21 -> MY.NET.108.1:21
TCP TTL:24 TOS:0x0 ID:39426
**SF**** Seq: 0x5F9CF69B   Ack: 0x58983421   Win: 0x404
77 0B 04 1B 50 8A                                w...P.
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/30-12:08:16.059625 211.178.63.4:8080 -> MY.NET.104.1:8080
TCP TTL:24 TOS:0x0 ID:39426
**SF**** Seq: 0x3B8C8415   Ack: 0x2BAD205   Win: 0x404
8B 0F C1 B0 AB 5E                                .....^
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/30-12:08:38.531893 211.178.63.4:53 -> MY.NET.111.1:53
TCP TTL:24 TOS:0x0 ID:39426
**SF**** Seq: 0x5AB0B17E   Ack: 0x42B6B509   Win: 0x404
01 01 08 0A 09 47                                .....G
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/30-12:08:43.040697 211.178.63.4:109 -> MY.NET.116.1:109
TCP TTL:24 TOS:0x0 ID:39426
**SF**** Seq: 0x12330AD3   Ack: 0x5666A6EB   Win: 0x404
00 00 00 00 00 00                                ......
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/29-20:30:30.592760 61.11.252.117:21 -> MY.NET.254.244:21
TCP TTL:17 TOS:0x0 ID:39426
```

```
**SF**** Seq: 0x6C7E8E3F   Ack: 0x4D827474   Win: 0x404
00 00 00 00 00 00                                  ......
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/29-20:30:30.694034 61.11.252.117:21 -> MY.NET.254.246:21
TCP TTL:17 TOS:0x0 ID:39426
**SF**** Seq: 0x6C7E8E3F   Ack: 0x4D827474   Win: 0x404
00 00 00 00 00 00                                  ......
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

As a matter of fact, you'd almost think the packets came from the same piece of software. Of note here, traffic from 211.178.63.4 did not trigger the SYN-FIN alert rule in Snort. It might be a good idea to review and test the MY.NET Snort system and ruleset to find out why.

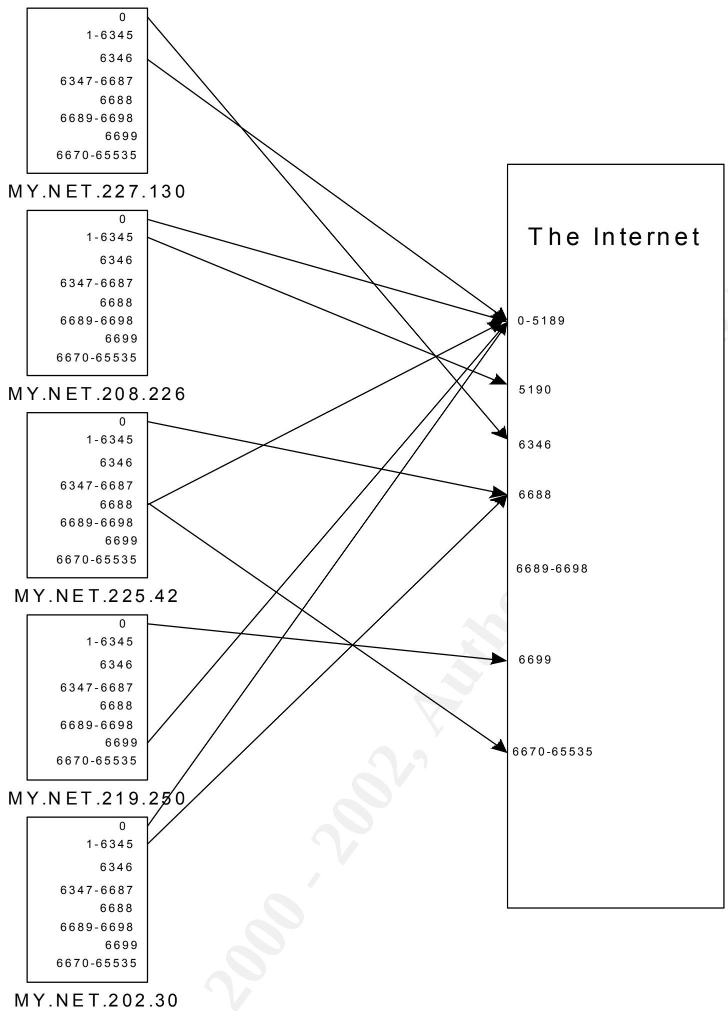
These internal hosts generated Out Of Spec entries:

MY.NET.150.139	MY.NET.201.254	MY.NET.202.30	MY.NET.208.226	MY.NET.219.250
MY.NET.224.66	MY.NET.225.42	MY.NET.227.130	MY.NET.228.86	MY.NET.229.26

The top five internal hosts:

Alerts	IP Address
39	MY.NET.227.130
8	MY.NET.208.226
7	MY.NET.225.42
6	MY.NET.219.250
4	MY.NET.202.30

Each of the top five is attempting to connect to various hosts all over the internet at or from the well-known Napster or Gnutella ports 6346, 6688, and 6699. There is no pattern of connection to any one host. MY.NET.227.130 seems to be the biggest Gnutella user on the network with all of its connections running either from or to port 6346. The link graph below illustrates the connection traffic from internal hosts to the Internet:



Other than the remarkable array of TCP flag combinations, there is nothing about the 335 packets generated by the internal hosts that stands out as either malicious or craft:

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/29-20:44:11.538393 MY.NET.227.130:0 -> 24.24.187.142:6346
TCP TTL:126 TOS:0x0 ID:48281  DF
21*F*PAU Seq: 0x70603EA   Ack: 0xD5C50016   Win: 0x5018
TCP Options => EOL EOL
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/29-21:15:38.085983 MY.NET.227.130:6346 -> 24.11.4.242:3501
TCP TTL:126 TOS:0x0 ID:58215  DF
2*SF**A* Seq: 0x3D2   Ack: 0x7F780EF3   Win: 0x5018
21 52 E4 C1 00 00 34 B5 00 07 F5 62 3E 5A FF F6  !R....4....b>Z..
E3 D9                                            ..
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/26-21:12:18.969510 MY.NET.208.226:84 -> 129.2.249.90:1389
TCP TTL:126 TOS:0x0 ID:63545  DF
21SFRPA* Seq: 0x14460020   Ack: 0x1E2701F5   Win: 0x5010
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

It would also seem as if MY.NET.208.226 spends a lot of time on AOL (TCP 5190). However, the remote host in this case (notregistered-129-2-249-90.student.umd.edu) is a workstation at the University of Maryland. It might be worthwhile to have a look at MY.NET.208.226.

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/26-21:12:15.052044 MY.NET.208.226:1389 -> 129.2.249.90:5190
TCP TTL:126 TOS:0x0 ID:51765  DF
21SFRPA* Seq: 0x3D0020   Ack: 0x1E2701D5   Win: 0x5010
TCP Options => Opt 32 (32): 2020 2000 0402 3031 3233 3435 0000 0000 0000 0000
0000 0000 0000 0000 0000 EOL EOL EOL EOL
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/26-21:12:21.243550 MY.NET.208.226:1389 -> 129.2.249.90:5190
TCP TTL:126 TOS:0x0 ID:65084  DF
21SFRPA* Seq: 0xD00020   Ack: 0x1E27020C   Win: 0x5010
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

As well, all of the top five hosts are generating OOS packets sourced from port 0.

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/29-20:44:11.538393 MY.NET.227.130:0 -> 24.24.187.142:6346
TCP TTL:126 TOS:0x0 ID:48281  DF
21*F*PAU Seq: 0x70603EA   Ack: 0xD5C50016   Win: 0x5018
TCP Options => EOL EOL
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/30-19:32:28.305715 MY.NET.219.250:0 -> 65.80.157.169:6699
TCP TTL:126 TOS:0x0 ID:26504  DF
21*FR*A* Seq: 0x9A102A1   Ack: 0x8FC15E6   Win: 0x5010
TCP Options => EOL EOL
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Port 0 is reserved (unassigned), so any traffic to or from port 0 to the Internet is most out of the ordinary. It could be that these machines are compromised and being used to scan or attack external hosts. Using Napster or Gnutella ports would be a good cover for something like that. It could be that the users responsible for these machines have installed a bad batch of Gnutella clones (there are many). It might be a good idea to set up a sniffer to find out what's going on. It wouldn't hurt to have a look at the boxes, either.

Appendices

Appendix A: Test Questions

What do these log entries represent?

```
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.47/21 flags
SYN
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.47/21 flags
RST
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.252/21
flags SYN
messages.18052630:May 26 17:17:50 pixpublic-active %PIX-2-106001:
Inbound TCP connection denied from 208.15.87.15/21 to a.b.c.252/21
flags RST
```

a) An FTP bounce attack
b) A way to fingerprint a machine
c) Password file theft
d) Frustration at not being able to download the latest N'Sync mp3's

Answer: b) A way to fingerprint a machine.

Stephen Northcutt's Attack Severity formula:

(Criticality + Lethality) – (System + Net Countermeasures) = Severity

a) Measures the effectiveness of Snort rules
b) Measures heartburn if your network has been compromised
c) Is a great way to calculate the ROI of your IDS
d) Helps prioritize your problems

Answer: d) Northcutt's formula is a quick way to determine which problem (or problems) should be given first priority.

What do you do when you see this:

```
May 27 12:08:10 pixpublic-active %PIX-2-106001: Inbound TCP
connection denied from 204.235.225.20/54037 to a.b.c.120/113 flags SYN
on interface outside
May 27 12:08:10 pixpublic-active %PIX-2-106001: Inbound TCP
connection denied from 204.235.225.20/54039 to a.b.c.120/113 flags SYN
on interface outside
May 27 12:08:10 pixpublic-active %PIX-2-106001: Inbound TCP
connection denied from 204.235.225.20/54040 to a.b.c.120/113 flags SYN
on interface outside
```

a) Run for the hills
b) Shun the source at your border router
c) Quit before your boss finds out
d) Check your firewall configuration.

Answer: d) Most commercial firewalls ship with some feature to handle Auth requests from SendMail servers.

What does this guy want?

202.206.240.11	51886	a.b.c.65	80	5/26/2001 13:30	..%c0%af. .*HTTP | GET /scripts/..%c0%af.. /winnt/system32/cmd.exe?/c+dir HTTP\0'
202.206.240.11	50439	a.b.c.173	80	5/26/2001 13:39	'..%c0%af.. *HTTP | GET /scripts/..%c0%af.. /winnt/system32/cmd.exe?/c+dir HTTP\0'
202.206.240.11	50491	a.b.c.173	80	5/26/2001 13:39	'..%c1%9c. .*HTTP | GET /scripts/..%c1%9c.. /winnt/system32/cmd.exe?/c+dir HTTP\0'

a) To see if you've patched your NT server
b) To see if you've patched your Linux server
c) His GIAC momgate login
d) His money back

Answer: a) If you run NT and have kept up with the HotFixes, this guy should get frustrated and go somewhere else. Otherwise, he won't go away.

Sudden heavy scans against your network is an indication of what?

a) A new exploit
b) A hole in your network you didn't know you had
c) Worm infestation
d) An electrolyte imbalance

Answer: a) While b) and c) are possible answers, they almost always seem to stem from a). When new exploits are published to the Internet, it's usually not very long before some begin to try them out. It's a good idea to keep up with Incidents.org Handler's Diary (http://www.incidents.org/diary/diary.php), as well as BugTraq (www.securityfocus.com) and a few others. It's a better idea to learn your network, so when you spot trends or read about new vulnerabilities, you can nip your problems in the butt.

Appendix B: Analysis

There are a lot of great scripts (and tips) in the posted practicals (http://www.sans.org/giactc/gcia.htm), and many were used for my analysis. I found that no one set of scripts or technique worked any better than another, so I used an amalgamation of everything I could find to generate tables, charts, statistics, and correlations. The highlights documented below serve only to underscore hours of trial and error. Eventually, every student develops his own technique. I've come away with some great tools and references to begin to develop my own.

The first step was to sort the alerts and generate some statistics. SnortSnarf, snort_sort.pl, and snort_stats.pl, all available here http://www.snort.org/snort-files.htm, was used. Lenny Zeltser's collection of perl scripts, correlate.tgz, was used to get both the alerts and the OOS logs into a .csv files. The .csv files were later imported into Microsoft Excel for some sorting.

To analyze the OOS alerts, a list of hosts was created:

```
# awk '$2 ~ /MY.NET/ {print $2}' * | awk -F: '{print $1}' | sort -u > ips.out
```

Then, a list of top talkers:

```
# for i in `cat ips.out`;do
awk '$2 ~ /'"$i"'/ {
        sum++
} END {
print sum,"'"$i"'"
}' *
done | sort -k 1n,1
```

The example data for the OOS packets was extracted with:

```
# sed -n "/218.106/,/=+=+/p" OOS.txt | more
```

To find machines looking for access to port 0 and port 1:

```
awk '$2 ~ /MY.NET/ {print $2}' * |awk '/:0$/ {print $0}' | sort -u
awk '$2 ~ /MY.NET/ {print $2}' * |awk '/:1$/ {print $0}' | sort -u
```

Courtesy of P.J. Goodwin (http://www.sans.org/y2k/practical/PJ_Goodwin_GCIA.doc)

To analyze the scan logs, Becky Bogle (http://www.sans.org/y2k/practical/Becky_Bogle_GCIA.doc) provided theses dandies that sort the number of individual IP addresses, calculate ports, get unique ports for a particular address, as well as unique hosts, and generates a list of networks:

#To sort the source ips, dest ips, source ports, and dest ports in an individual scan file by number #of occurrences:

```
#!/usr/bin/perl
use strict;
use warnings;

my(%s_ip);
%s_ip=();
my(%d_ip);
%d_ip=();
```

```
my(%s_port);
%s_port=();
my(%d_port);
%d_port=();
open(INFILE,"c:\\snort\\practical\\scans\\scan-mar");
	while(<INFILE>)
		{
		my($month,$day,$time,$source,$flow,$dest,$scan,$misc)=split(/ /,$_);
		my($source_ip,$source_port)=split(/:/,$source);
		my($dest_ip,$dest_port)=split(/:/,$dest);
		if (exists $s_ip{$source_ip})
			{
			$s_ip{$source_ip}++;
			}
		else {$s_ip{$source_ip}=1};
		if (exists $d_ip{$dest_ip})
			{
			$d_ip{$dest_ip}++;
			}
		else {$d_ip{$dest_ip}=1};
		if (exists $s_port{$source_port})
			{
			$s_port{$source_port}++;
			}
		else {$s_port{$source_port}=1};
		if (exists $d_port{$dest_port})
			{
			$d_port{$dest_port}++;
			}
		else {$d_port{$dest_port}=1};
		}
		open(OUTFILE,">>c:\\snort\\practical\\scans\\source_ip.txt");
		print OUTFILE "\n\nSource IP by Volume\n\n";
		foreach(reverse sort {$s_ip{$a}<=>$s_ip{$b}} keys %s_ip)
			{
			print OUTFILE "$_ $s_ip{$_}\n"
			}
		open(OUTFILE,">>c:\\snort\\practical\\scans\\dest_ip.txt");
		print OUTFILE "\n\nDest IP by Volume\n\n";
		foreach(reverse sort {$d_ip{$a}<=>$d_ip{$b}} keys %d_ip)
			{
			print OUTFILE "$_ $d_ip{$_}\n"
			}
		open(OUTFILE,">>c:\\snort\\practical\\scans\\source_port.txt");
		print OUTFILE "\n\nSource Port by Volume\n\n";
		foreach(reverse sort {$s_port{$a}<=>$s_port{$b}} keys %s_port)
			{
			print OUTFILE "$_ $s_port{$_}\n"
			}
		open(OUTFILE,">>c:\\snort\\practical\\scans\\dest_port.txt");
		print OUTFILE "\n\nDest Port by Volume\n\n";
		foreach(reverse sort {$d_port{$a}<=>$d_port{$b}} keys %d_port)
			{
			print OUTFILE "$_ $d_port{$_}\n"
			}
```

#To get the total counts for each type of scan:

```
#!/usr/bin/perl
use strict;
use warnings;

my($scanfile);
my(%scanhash);
%scanhash=();
opendir(SCANDIR,"c:\\giac\\scans");
while($scanfile=readdir SCANDIR)
	{
	print "$scanfile starting.\n";
	open(INFILE,"c:\\giac\\scans\\$scanfile");
	while(<INFILE>)
		{
		if (m/^Dec|^Nov|^Jan/)
			{
			chomp($_);
			s/MY.NET/123.456/;
			s/  / /;
			my($line)="$_";
			open(OUTFILE,">>c:\\giac\\scans\\newscans\\allscans.txt");
			print OUTFILE "$line\n";
			my($month,$day,$time,$source,$flow,$dest,$scan,$misc)=split(/ /,$_);
			if (exists $scanhash{$scan})
				{
				$scanhash{$scan}++;
				}
			else {$scanhash{$scan}=1};
			}
		}
	print "$scanfile done!\n";
	}
foreach(reverse sort {$scanhash{$a}<=>$scanhash{$b}} keys %scanhash)
	{
	print "$_ $scanhash{$_}";
	}
```

#To divide the scans into individual files by scan type:

```
#!/usr/bin/perl
use strict;
use warnings;

open(INFILE,"c:\\snort\\practical\\scans\\scan-mar");
while(<INFILE>)
		{
		if (m/^Mar/)
			{
			chomp($_);
			my($month,$day,$time,$source,$flow,$dest,$scan,$misc)=split(/ /,$_);
			open(OUTFILE,">>c:\\snort\\practical\\scans\\$scantype.txt");
			print OUTFILE "$_\n";
			}
		}
```

Then, to generate top scans for a particular port (say, port 21)

```
# awk -F, '$6 == 21 {print $3}' SYN-FINscan.csv | sort -u | while read i;do
       awk -F, '$6 == 21 && $3 == "'"$i"'" {
          sum++
       } END {
          print sum",'"$i"' "
       }' SYN-FINscan.csv
       done
```

Other data was gathered using combinations of grep and awk on the alert, scan, and OOS logs.

The scan logs were also analyzed using Perl scripts borrowed from Mike Bell's practical (http://www.sans.org/y2k/practical/Mike_Bell_GCIA.doc).

snort_dest.pl:

```
#!/usr/bin/perl
#

#   Start mainline code
while (<>) {
#
#  Check for blank line, if so process next line
#
    if ( $_ eq "" )  { next };
#
#  Check for spp_portscan, if it is get the next record
#
#   Tokenize the string so we can use it
#
    if ($_ =~ m/^\w{3}\s+\d+\s+\d+\:\d+\:\d+\s+([\w\d\.]+)\:(\d+)\s+\-
\>\s+([\d\w\.]+)\:(\d+)\s+UDP/) {

        $saddr  =       $1;
        $sport  =       $2;
        $daddr  =       $3;
        $dport  =       $4;
        $dest{$daddr}++;
    }  # end if

    if ($_ =~ m/^\w{3}\s+\d+\s+\d+\:\d+\:\d+\s+([\w\d\.]+)\:(\d+)\s+\-
\>\s+([\d\w\.]+)\:(\d+)\s+([-\w]+)\s+[\*1PUSFAR]+\s+/) {

        $saddr  =       $1;
        $sport  =       $2;
        $daddr  =       $3;
        $dport  =       $4;
        $descrp =       $5;
        $dest{$daddr}++;
    }  # end if

}  # while

foreach $num ( sort keys(%dest) ) {
        $strings = $dest{$num};
        foreach $string (split(' ', $strings)) {
                print "$string\t$num\n";
        }
}
```

Appendix C: References

Evaluate an Attack or Present A Challenge
Parsing PIX Firewall Logs

Setting up Snort and ACID on Windows machines
http://www.snort.org/papers/SnortOnAcid_Win32.htm

PIX configuration guide
http://www.cisco.com/univercd/cc/td/doc/product/iaabu/pix/pix_v43/pix43cfg/index.htm

PIX syslog references
http://www.cisco.com/univercd/cc/td/doc/product/iaabu/pix/pix_v43/syslog/index.htm
http://www.cisco.com/univercd/cc/td/doc/product/iaabu/pix/pix_v43/syslog/pixemapa.htm

MySQL configuration guide and FAQ
http://www.mysql.com/documentation/

ACID configuration guide and FAQ
http://www.cert.org/kb/acid/

ActiveState Perl scripting reference for Win32
http://aspn.activestate.com/ASPN/Reference/Products/ActivePerl/lib/Pod/perlwin32.html

ActiveState Perl regular expression tutorial
http://aspn.activestate.com/ASPN/Reference/Products/ActivePerl/lib/Pod/perlretut.html

General References and bibliography

Handy books:

The SANS Institute. TCP/IP for Firewalls and Intrusion Detection. Course Reference, Baltimore SANS, May 2001.

The SANS Institute. Network Traffic Analysis Using TCP Dump. Course Reference, Baltimore SANS, May 2001.

The SANS Institute. Intrusion Detection: Signatures and Analysis Parts 1 & 2. Course Reference, Baltimore SANS, May 2001.

Dougherty, Dale and Arnold Robbins. sed and awk 2nd Edition. Cambridge, MA: O'Reilly, 1997.

Loshin, Pete. TCP/IP 2nd Edition. London, UK: Academic Press, LTD,1995.

Northcutt, Stephen and Judy Novak. Network Intrusion Detection, An Analysts Handbook 2nd Edition. Indianapolis, IN: New Riders Publishing, 2000.

Security Websites:

- http://www.dshield.org
- http://www.mynetwatchman.com
- http://www.incidents.org
- http://www.sans.org
- http://cexx.org/
- http://www.securityfocus.com
- http://www.snort.org/
- http://www.cert.org

http://mitre.cve.org
http://www.whitehats.com/
http://www.robertgraham.com/pubs/firewall-seen.html
http://www.sans.org/giactc/gcia.htm
http://www.enteract.com/~lspitz/forensics/
http://packetstormsecurity.org/index.shtml

Internet search engines:

http://www.google.com/advanced_search

IP address lookup:

http://www.arin.net/whois/index.html
http://www.internic.net/whois.html
http://www.ripe.net/db/whois.html
http://whois.apnic.net/

Port assignments:

http://www.isi.edu/in-notes/iana/assignments/port-numbers
http://www.snort.org/Database/portsearch.asp
http://advice.networkice.com/advice/Exploits/Ports/
http://www.simovits.com/nyheter9902.html
http://www.pointman.org/Docs/port-list
http://www.sans.org/newlook/resources/IDFAQ/oddports.htm
http://www.good-stuff.co.uk/useful/portfull.html
http://www.clic.net/~hello/puppet/nnports.html
http://www.tlsecurity.com/trojanh.htm
http://www.ec11.dial.pipex.com/port-num3.htm