GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permited without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Network Monitoring and Threat Detection In-Depth (Security 503)"
at http://www.giac.org/registration/gcia

GCIA Certification – Practical Assignment
Version 2.8b

Dallas Lone Star – March 2001

Bruno Marien

Assignment 1 – Network Detects

Detect 1

```
05/18-22:55:52.040347  [**] IDS218  - CVE-1999-0070 - TEST-CGI probe
    [**] 212.198.0.93:46387  -> MY.NET.70.15:80
05/18-22:55:52.340767  [**] IDS224  - CVE-1999-0045 - NPH CGI access
    attempt [**] 212.198.0.93:46408  -> MY.NET.70.15:80
05/18-22:55:52.652019  [**] IDS128  - CVE-1999-0067 - CGI phf attempt
    [**] 212.198.0.93:46436  -> MY.NET.70.15:80
05/18-22:55:53.013349  [**] ID S128 - CVE-1999-0067 - CGI phf attempt
    [**] 212.198.0.93:46464  -> MY.NET.70.15:80
05/18-22:55:53.301545  [**] IDS128  - CVE-1999-0067 - CGI phf attempt
    [**] 212.198.0.93:46482  -> MY.NET.70.15:80
05/18-22:55:53.612524  [**] CVE -1999-0196 - WEB-CGI-Websendmai l CGI
    access attempt [**] 212.198.0.93:46502  -> MY.NET.70.15:80
05/18-22:55:54.676771  [**] IDS219  - WEB-CGI-Perl access attempt [**]
    212.198.0.93:46592  -> MY.NET.70.15:80
05/18-22:55:55.014925  [**] CVE -1999-0953 - WEB-MISC - wwwboard.pl
    attempt [**] 212. 198.0.93:46620  -> MY.NET.70.15:80
05/18-22:55:55.387432  [**] WEB -CGI-WWW-SQL CGI access attempt [**]
    212.198.0.93:46648  -> MY.NET.70.15:80
05/18-22:55:56.066554  [**] WEB -CGI-AT-admin CGI access attempt [**]
    212.198.0.93:46694  -> MY.NET.70.15:80
05/18-22:55:56.369899  [**] WEB -CGI-wwwadmin [**] 212.198.0.93:46707
    -> MY.NET.70.15:80
05/18-22:55:56.746623  [**] IDS226  - CVE-1999-0172 - CGI-formmail
    [**] 212.198.0.93:46728  -> MY.NET.70.15:80
05/18-22:55:57.063769  [**] WEB -CGI-sendform.cgi [**]
    212.198.0.93:4 6756 -> MY.NET.70.15:80
05/18-22:55:57.369840  [**] WEB -CGI-Maillist CGI access attempt [**]
    212.198.0.93:46785  -> MY.NET.70.15:80
05/18-22:55:57.699156  [**] IIS -achg.htr Attempt [**]
    212.198.0.93:46805  -> MY.NET.70.15:80
05/18-22:55:58.039772  [**] CAN -1999-0407 - IIS-aexp.htr Attempt [**]
    212.198.0.93:46831  -> MY.NET.70.15:80
05/18-22:55:58.352346  [**] CAN -1999-0407 - IIS-aexp2.htr Attempt
    [**] 212.198.0.93:46865  -> MY.NET.70.15:80
05/18-22:55:58.680886  [**] CAN -1999-0407 - IIS-aexp2b.htr Attempt
    [**] 212.198.0.93:46902  -> MY.NET.70.15:80
05/18-22:55:59.095521  [**] CVE -2000-0304 - IIS-iisadmpwd [**]
    212.198.0.93:46935  -> MY.NET.70.15:80
05/18-22:55:59.397828  [**] CAN -1999-0407 - IIS-aexp4.htr Attempt
    [**] 212.198.0.93:46962  -> MY.NET.70.15:80
05/18-22:55:59.716928  [**] CAN -1999-0407 - IIS-aexp4b.htr Attempt
    [**] 212.198.0.93:46980  -> MY.NET.70.15:80
05/18-22:56:00.082364  [**] CAN -1999-0407 - IIS-anot.htr Attempt [**]
    212.198.0.93:47011  -> MY.NET.70.15:80
05/18-22:56:00.412647  [**] CAN -1999-0407 - IIS-anot3.htr Attempt
    [**] 212.198.0.93:47034  -> MY.NET.70.15:80
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
05/18-22:56:00.714430  [**] CAN -1999-0736 - IIS-showcode [**]
    212.198.0.93:47057  -> MY.NET.70.15:80
05/18-22:56:01.053166  [**] WEB -MISC-AuthChangeUrl [**]
    212.198.0.93:47085  -> MY.NET.70.15:80
05/18-22:56:01.332728  [**] WEB -MISC-AuthChangeUrl [**]
    212.198.0.93:47111  -> MY.NET.70.15:80
05/18-22:56:01.644224  [**] WEB -/.... [**] 212.198.0.93:47151  ->
    MY.NET.70.15:80
05/18-22:56:01.920561  [**] WEB -MISC-AuthChangeUrl [**]
    212.198.0.93:47170  -> MY.NET.70.15:80
05/18-22:56:02.195388  [**] IIS -fpcount [**] 212.198.0.93:47191  ->
    MY.NET.70.15:80
05/18-22:56:02.536365  [**] CAN -2000-0726 - BUGTRAQ ID 1623  - IIS-
    CGImail [**] 212.198.0.93:47213  -> MY.NET.70.15:80
05/18-22:56:02.805143  [**] CVE -1999-0191 - IIS-newdsn [**]
    212.198.0.93:47231  -> MY.NET.70.15:80
05/18-22:56:03.082929  [**] IIS -getdrvs.exe [**] 212.198.0.93:47250  -
    > MY.NET.70.15:80
05/18-22:56:03.379936  [**] CVS -1999-0937 - WEB-CGI-Bnbform CGI
    access attempt [**] 212.198.0.93:47275  -> MY.NET.70.15:80
05/18-22:56:03.646722  [**] CVE -1999-0936 - WEB-CGI-survey [**]
    212.198.0.93:47293  -> MY.NET.70.15:80
05/18-22:56:03.919563  [**] WEB -Domino-domcfg.nsf [**]
    212.198.0.93:47310  -> MY.NET.70.15:80
05/18-22:56:04.188457  [**] CVE -1999-0021 - WEB-count.cgi [**]
    212.198.0.93:47321  -> MY.NET.70.15:80
05/18-22:56:04.553156  [**] IDS228  - CVE-1999-0237 - Guestbook CGI
    access attempt [**] 212.198.0.93:47346  -> MY.NET.70.15:80
05/18-22:56:04.817674  [**] CVE -1999-0147 - WEB-CGI-Aglimpse CGI
    access attempt [* *] 212.198.0.93:47367  -> MY.NET.70.15:80
05/18-22:56:05.095488  [**] IDS221  - CVE-1999-0612 - Finger CGI
    access attempt [**] 212.198.0.93:47390  -> MY.NET.70.15:80
05/18-22:56:05.370415  [**] CVE -1999-0260 - WEB-MISC - /cgi-bin/jj
    attempt [**] 212.198.0.93: 47415 -> MY.NET.70.15:80
05/18-22:56:05.642043  [**] WEB -CGI-CGI Man access attempt [**]
    212.198.0.93:47431  -> MY.NET.70.15:80
05/18-22:56:05.907416  [**] CVE -1999-0039 - WEB-CGI-Webdist CGI
    access attempt [**] 212.198.0.93:47457  -> MY.NET.70.15:80
05/18-22:56:06.768190  [**] WEB -CGI-day5datacopier.cgi [**]
    212.198.0.93:47549  -> MY.NET.70.15:80
05/18-22:56:07.563849  [**] WEB -CGI-day5datanotifier.cgi [**]
    212.198.0.93:47610  -> MY.NET.70.15:80
05/18-22:56:07.851510  [**] CVE -1999-0270 - WEB-CGI-CGI pf displa y
    access attempt [**] 212.198.0.93:47633  -> MY.NET.70.15:80
05/18-22:56:08.148265  [**] WEB -CGI-Files CGI access attempt [**]
    212.198.0.93:47654  -> MY.NET.70.15:80
05/18-22:56:08.420452  [**] CVE -1999-0175 - WEB-MISC-convert.bas
    Attempt [**] 212.198.0.93:4 7676 -> MY.NET.70.15:80
05/18-22:56:08.682308  [**] WEB -CGI-dumpenv.pl [**]
    212.198.0.93:47696  -> MY.NET.70.15:80
05/18-22:56:08.953991  [**] WEB -CGI-upload.pl [**] 212.198.0.93:47720
    -> MY.NET.70.15:80
05/18-22:56:09.686269  [**] CVE -1999-0146 - WEB-CGI-Campas CGI access
    attempt [**] 212.198.0.93:47768  -> MY.NET.70.15:80
05/18-22:56:10.021781  [**] WEB -CGI-Textcounter CGI access attempt
    [**] 212.198.0.93:47793  -> MY.NET.70.15:80
05/18-22:56:10.623911  [**] CVE -1999-0176 - WEB-CGI-Webgais CGI
    access attempt  [**] 212.198.0.93:47838  -> MY.NET.70.15:80
05/18-22:56:10.896206  [**] CVE -1999-0264 - WEB-CGI-Htmlscript CGI
    access attempt [**] 212.198.0.93:47860  -> MY.NET.70.15:80
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
05/18-22:56:11.170409  [**] CVE -1999-0177 - WEB-CGI-Upload CGI access
    attempt [**] 212. 198.0.93:47886  -> MY.NET.70.15:80
05/18-22:56:11.506478  [**] CVE -1999-0177 - WEB-CGI-Upload CGI access
    attempt [**] 212.198.0.93:47898  -> MY.NET.70.15:80
05/18-22:56:12.124093  [**] WEB -CGI-Args CGI access attempt [**]
    212.198.0.93:47937  -> MY.NET.70.15:8 0
05/18-22:56:12.400777  [**] WEB -CGI-NPH-publish CGI access attempt
    [**] 212.198.0.93:47953  -> MY.NET.70.15:80
05/18-22:56:12.715290  [**] CVE -1999-0262 - WEB-CGI-Faxsurvey probe
    [**] 212.198.0.93:47973  -> MY.NET.70.15:80
05/18-22:56:13.014374  [**] WEB -~root [**] 212.198.0.93:48002  ->
    MY.NET.70.15:80
05/18-22:56:13.305709  [**] FrontPage -users.pwd [**]
    212.198.0.93:48026  -> MY.NET.70.15:80
05/18-22:56:13.584418  [**] BUGTRAQ ID 1205  - FrontPage -
    administrators.pwd [**] 212.198.0.93:48047  -> MY.NET.70.15:80
05/18-22:56:15.638842  [**] BUGTRAQ ID 162  - SCAN - Whisker Stealth -
    IIS search97 access attempt [**] 212.198.0.93:48199  ->
    MY.NET.70.15:80
05/18-22:56:15.918174  [**] CVE -1999-0269 - WEB-PageService [**]
    212.198.0.93:48223  -> MY.NET.70.15:80

[**] IDS218  - CVE-1999-0070 - TEST-CGI probe [**]
05/18-22:55:52.040347 0:B0:64:12:8F:60  -> 8:0:20:A0:11:63 type:0x800
len:0xD3
212.198.0.93:46387  -> MY.NET.70.15:80 TCP TTL:42 TOS:0x0 ID:56679
IpLen:20 DgmLen:197 DF
***AP*** Seq: 0x9AD9465B  Ack: 0x52B9844F  Win: 0x  2238  TcpLen: 20
47 45 54 20 2F 63 67 69 2D 62 69 6E 2F 74 65 73  GET /cgi  -bin/tes
74 2D 63 67 69 20 48 54 54 50 2F 31 2E 31 0D 0A  t  -cgi HTTP/1.1..
48 6F 73 74 3A 20 XX XX XX XX XX XX XX XX XX XX  Host: XXXXXXXXXX
XX XX XX XX XX XX 0D 0A 43 6F 6E 6E 65 63  74 69  XXXXXX..Connecti
6F 6E 3A 20 6B 65 65 70 2D 61 6C 69 76 65 0D 0A  on: keep  -alive..
56 69 61 3A 20 31 2E 30 20 63 75 72 69 65 20 28  Via: 1.0 curie (
4E 65 74 43 61 63 68 65 20 4E 65 74 41 70 70 2F  NetCache NetApp/
35 2E 30 2E 31 52 32 29 0D 0A 58  2D 46 6F 72 77  5.0.1R2)..X -Forw
61 72 64 65 64 2D 46 6F 72 3A 20 32 31 32 2E 31  arded  -For: 212.1
39 38 2E 32 33 31 2E 36 39 0D 0A 0D 0A           98.231.69....
```

1. Source of Trace:

An IDS at our network, just in front of the corporate firewall.

2. Detect was generated by:

Snort IDS (version 1.7).

3. Probability the source address was spoofed:

It is quite unlikely that the source address was spoofed. All these attacks require an established connection, so spoofing the source address wouldn't work. Wha t might be possible though, is that the host (of which we see the source address) is compromised and the real attacker is hiding his real address by using this compromised host. Another possibility is that the source we see is an HTTP proxy server, which w ould automatically hide the attacker's real IP address.

When looking at the actual packet dump, we see the *X-Forwarded-For* field, informing us that the original source address actually was 212.198.231.69.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

4. Description of attack:

Someone was looking fo r possible exploits for our webserver (vulnerability scanning). (At least) 54 different kind of known vulnerabilities were tried.

What follows is an overview of the different exploits/vulnerabilities that were tried, with a little explanation. If there wa s a CVE classification, I used it; Bugtraq classification was used next; finally, unclassified alerts are given.

Common Vulnerabilities and Exposures classification:

CAN-1999-0509	Perl, sh, csh, or other shell interpreters are installed in the cgi -bin directory on a WWW site, which allows remote attackers to execute arbitrary commands.
CAN-1999-0736	The showcode.asp sample file in IIS and Site Server allows remote attackers to read arbitrary files.
CVE-1999-0021	Arbitrary command execution via buffer over flow in Count.cgi (wwwcount) cgi-bin program.
CVE-1999-0039	Arbitrary command execution using webdist CGI program in IRIX.
CVE-1999-0045	List of arbitrary files on Web host via nph -test-cgi script
CVE-1999-0067	CGI phf program allows remote command executi on through shell metacharacters.
CVE-1999-0070	test-cgi program allows an attacker to list files on the server
CVE-1999-0146	The campas CGI program provided with some NCSA webservers allows an attacker to read arbitrary files.
CVE-1999-0147	The aglimpse CG I program of the Glimpse package allows remote execution of arbitrary commands
CVE-1999-0172	FormMail CGI program allows remote execution of commands.
CVE-1999-0175	The convert.bas program in the Novell webserver allows a remote attackers to read any file on the system that is internally accessible by the webserver.
CVE-1999-0176	The Webgais program allows a remote user to execute arbitrary commands.
CVE-1999-0177	The uploader program in the WebSite webserver allows a remote attacker to execute arbitrary p rograms.
CVE-1999-0191	IIS newdsn.exe CGI script allows remote users to overwrite files.
CVE-1999-0196	The websendmail program in the Webgais program allows a remote user to access arbitrary files.
CVE-1999-0237	Remote execution of arbitrary commands throu gh Guestbook CGI program.
CVE-1999-0260	The jj CGI program allows command execution via shell metacharacters.
CVE-1999-0262	faxsurvey CGI script on Linux allows remote command execution via shell metacharacters.
CVE-1999-0264	htmlscript CGI program allows remote read access to files.
CVE-1999-0269	Netscape Enterprise servers may list files through the PageServices query.
CVE-1999-0270	pfdispaly CGI program for SGI's Performer API Search Tool allows read access to files.
CVE-1999-0407	By default, IIS 4.0 ha s a virtual directory /IISADMPWD which contains files that can be used as proxies for brute force password attacks, or to identify valid users on the system.
CVE-1999-0612	A version of finger is running that exposes valid user information to any entity on the network.
CVE-1999-0936	BNBSurvey survey.cgi program allows remote attackers to execute commands via shell metacharacters.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

CVE-1999-0937	BNBForm allows remote attackers to read arbitrary files via the automessage hidden form variable.
CVE-1999-0953	WWWBoard stores encrypted passwords in a password file that is under the web root and thus accessible by remote attackers.
CVE-2000-0304	Microsoft IIS 4.0 and 5.0 with the IISADMPWD virtual directory installed allows a remote attacker to cause a denial of service via a malformed request to the inetinfo.exe program, aka the "Undelimited .HTR Request" vulnerability.
CVE-2000-0726	CGIMail.exe CGI program in Stalkerlab Mailers 1.1.2 allows remote attackers to read arbitrary files by specifying the file in the $Attach$ hidden form variable.

Bugtrack classification:

162	Whisker Stealth - IIS search97 access attempt
1205	FrontPage-administrators.pwd
2110	IIS-achg.htr Attempt

Unclassified:

FrontPage	users.pwd
IIS	fpcount
IIS	getdrvs.exe
WEB	/....
WEB	~root
WEB-CGI	Args CGI access attempt
WEB-CGI	AT-admin CGI access attempt
WEB-CGI	day5datacopier.cgi
WEB-CGI	day5datanotifier.cgi
WEB-CGI	dumpenv.pl
WEB-CGI	Files CGI access attempt
WEB-CGI	Maillist CGI access attempt
WEB-CGI	NPH-publish CGI access attempt
WEB-CGI	sendform.cgi
WEB-CGI	Textcounter CGI access attempt
WEB-CGI	upload.pl
WEB-CGI	wwwadmin
WEB-CGI	WWW-SQL CGI access attempt
WEB-Domino	domcfg.nsf
WEB-MISC	AuthChangeUrl

5. Attack mechanism:

What matters in this attack, isn't that the attacker tried a known exploit, but that he tried 63 exploits (of which 54 different ones) in 22 seconds! Most likely this is a vulnerability scan. Looking at the actual packets confirms this: the packet dump shows " *GET /cgi-bin/test.cgi HTTP/1.0"* as get request and not for instance " *GET /cgi-bin/test.cgi?* HTTP/1.0"* which would mean actually (ab)using the exploit, not just testing for the presence of it.

This is most probably some kind of script/tool (s)he wrote or found somewhere. The script looks for several vulnerabilities. It was written to be usable against a bunch of webservers, because you see vulnerabilities for different types of webservers (unix webservers (using CGI scripting), IIS, Netscape Enterprise server and others). Maybe there is a tool that generates exactly all the attacks seen here, but I'm not sure which. There was an alert that mentions *whisker*, so it might be whisker, but this could also just be inserted to misguide you.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

When looking at the (sanitized) packet dump of the first exploit, we see that the HTTP traffic is actually coming from 212.198.231.69, using *curie* (the name of the machine) as some kind of proxy.

Further investigation reveals that NetCache from NetworkAppliance is indeed some kind of proxy. Actually, it delivers content closer to the end users by mirroring data as necessary. Big companies (such as ISP's) that require very high performance use it. And indeed, 212.198.231.69 is part of 212.198.0.0/16, which is owned by Lyonnaise Communications, an ISP (and TV distributor) in France. It uses DHCP for IP address allocation, so pointing out the culprit won't be easy. ISPs however (should) keep a log of all their connections, so it should be possible to find out exactly who did it.

6. Correlations:

Looking at our systems, it couldn't be correlated to any other events (e.g.: reconnaissance), but then again, it's not that hard to find a webserver. Maybe (s)he just came to our site and thought how nice it would be to be able to show off to its friends that (s)he hacked our site (we are an IT security company —using IIS? yes, such things happen, mostly for stupid reasons, but we're working on it —which is always a little bit more impressive when being able to compromise their systems). Or (s)he was just interested to see how secure *our* servers are (being a security firm).

It might be interesting to mention that there were no other alerts seen (from anywhere) during this attack than the ones shown above, so this wasn't a decoy to cover up a *real* attack.

Finding this exact combination is a little bid hard. Finding references of occurrences in the past of the individual alerts is easy. Here are some:

- CAN-1999-0509 is no. 2 in The Ten Most Critical Internet Security Threats (http://www.sans.org/topten.htm)
- CVE-1999-0407 is mentioned in http://www.sans.org/y2k/031401.htm .

7. Evidence of active targeting:

This was very active targeting. There was no scan preceding the first alert. Immediately the webserver was targeted and nothing else. However, the guy/girl is just looking for vulnerabilities, not abusing them. But unless you are hired to do so, this probably means you want to accomplish something (hack the server).

8. Severity:

Criticality	: 5	this is our corporate webserver; as a security firm we would lose credibility if it was compromised, which could harm us quite a lot
Lethality	: 4	not a very specific attack, more looking around (although quite aggressively)
System Countermeasures	: 5	all patches applied, OS completely hardened
Network Countermeasures	: 4	latest version of FW-1, good rule base (only inbound HTTP traffic allowed, nothing outbound allowed), but http content isn't stripped, so it will pass

Severity = (Criticality + Lethality) – (System Countermeasures + Network Countermeasures)
= (5 + 4) – (5 + 4)
= 0

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

9. Defensive recommendation:

Defenses are fine. Attacks will not likely be successful if we always apply the very latest patches on our webserver and also update the firewall regularly.

10. Multiple choice test question:

When looking carefully at the packet dump sho wn above, would you say the sender is:

A. searching for confidential files
B. searching for normally accessible files
C. searching for vulnerabilities
D. requesting a file from your webserver, by browsing the site

The correct answer is C: the sender is just looking for the presence of the /cgi -bin/test-cgi script, without giving any parameters to it (by appending for instance '*?*'). The latter would indicate abusing the script.

Detect 2

internal sensor (inside DMZ)

```
2001/05/27 08:59:28 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.3:80 3045
2001/05/27 08:59:39 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.11:80 3045
2001/05/27 08:59:28 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.4:80 3045
2001/05/27 08:59:38 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.10:80 3045
2001/05/27 08:59:38 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.9:80 3045
2001/05/27 09:00:37 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.18:80 3045
2001/05/27 09:00:35 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.14:80 3045
2001/05/27 09:00:29 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.13:80 3045
2001/05/27 09:00:38 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.19:80 3045
2001/05/27 09:00:39 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.21:80 3045
2001/05/27 09:00:29 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.12:80 3045
2001/05/27 09:00:36 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.17:80 3045
2001/05/27 09:00:32 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.20:80 3045
2001/05/27 09:01:38 sen_dmz_internal 38.33.20.159:3751 -> CLIENT.NET.4.26:80 3200
2001/05/27 09:01:37 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.31:80 3045
2001/05/27 09:01:44 sen_dmz_internal 38.33.20.159:3854 -> CLIENT.NET.4.26:80 3226
2001/05/27 09:01:34 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.25:80 3045
2001/05/27 09:01:37 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.27:80 3045
2001/05/27 09:01:41 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.29:80 3045
2001/05/27 09:01:48 sen_dmz_internal 38.33.20.159:4076 -> CLIENT.NET.4.30:80 3217
2001/05/27 09:00:44 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.24:80 3045
2001/05/27 09:01:50 sen_dmz_internal 38.33.20.159:4083 -> CLIENT.NET.4.31:80 3217
2001/05/27 09:01:36 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.30:80 3045
2001/05/27 09:01:38 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.32:80 3045
2001/05/27 09:01:43 sen_dmz_internal 38.33.20.159:3822 -> CLIENT.NET.4.30:80 3200
2001/05/27 09:01:44 sen_dmz_internal 38.33.20.159:3855 -> CLIENT.NET.4.31:80 3200
2001/05/27 09:01:44 sen_dmz_internal 38.33.20.159:3856 -> CLIENT.NET.4.32:80 3200
2001/05/27 09:01:50 sen_dmz_internal 38.33.20.159:4085 -> CLIENT.NET.4.32:80 3217
2001/05/27 09:01:40 sen_dmz_internal 38.33.20.159:3782 -> CLIENT.NET.4.26:80 3217
2001/05/27 09:02:08 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.35:80 3045
2001/05/27 09:03:14 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.53:80 3045
2001/05/27 09:02:57 sen_dmz_internal 38.33.20.159:1831 -> CLIENT.NET.4.48:80 3217
2001/05/27 09:02:56 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.49:80 3045
2001/05/27 09:02:26 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.40:80 3045
2001/05/27 09:02:13 sen_dmz_internal 38.33.20.159:4703 -> CLIENT.NET.4.35:80 3200
2001/05/27 09:02:00 sen_dmz_internal 38.33.20.159:4371 -> CLIENT.NET.4.30:80 5054
2001/05/27 09:02:05 sen_dmz_internal 38.33.20.159:4502 -> CLIENT.NET.4.33:80 3200
2001/05/27 09:02:41 sen_dmz_internal 38.33.20.159:1297 -> CLIENT.NET.4.41:80 3226
2001/05/27 09:03:06 sen_dmz_internal 38.33.20.159:2061 -> CLIENT.NET.4.51:80 3226
2001/05/27 09:02:55 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.51:80 3045
2001/05/27 09:02:27 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.41:80 3045
2001/05/27 09:02:57 sen_dmz_internal 38.33.20.159:1813 -> CLIENT.NET.4.48:80 3213
2001/05/27 09:02:26 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.39:80 3045
2001/05/27 09:02:19 sen_dmz_internal 38.33.20.159:4776 -> CLIENT.NET.4.36:80 3226
2001/05/27 09:02:39 sen_dmz_internal 38.33.20.159:1289 -> CLIENT.NET.4.40:80 3226
2001/05/27 09:02:41 sen_dmz_internal 38.33.20.159:1315 -> CLIENT.NET.4.39:80 5040
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
2001/05/27 09:02:09 sen_dmz_internal 38.33.20.159:4608 -> CLIENT.NET.4.33:80 3217
                          (242 entries omitted)
2001/05/27 09:25:43 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.251:80 3045
2001/05/27 09:25:38 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.249:80 3045
2001/05/27 09:25:41 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.250:80 3045
2001/05/27 09:26:25 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.253:80 3045
2001/05/27 09:26:20 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.252:80 3045
```

external sensor (outside DMZ)

```
2001/05/27 08:59:26 sen_dmz_external 38.33.20.159:4336 -> CLIENT.NET.4.1:79 3002
2001/05/27 08:59:27 sen_dmz_external 38.33.20.159:4352 -> CLIENT.NET.4.2:79 3002
2001/05/27 08:59:27 sen_dmz_external 38.33.20.159:4384 -> CLIENT.NET.4.4:79 3002
2001/05/27 08:59:27 sen_dmz_external 38.33.20.159:4368 -> CLIENT.NET.4.3:79 3002
2001/05/27 08:59:37 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.9:80 3045
2001/05/27 08:59:37 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.10:80 3045
2001/05/27 08:59:38 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.11:80 3045
2001/05/27 09:00:27 sen_dmz_external 38.33.20.159:1652 -> CLIENT.NET.4.13:79 3002
2001/05/27 09:00:27 sen_dmz_external 38.33.20.159:1636 -> CLIENT.NET.4.12:79 3002
2001/05/27 09:00:27 sen_dmz_external 38.33.20.159:1696 -> CLIENT.NET.4.14:79 3002
2001/05/27 09:00:29 sen_dmz_external 38.33.20.159:1766 -> CLIENT.NET.4.17:79 3002
2001/05/27 09:00:29 sen_dmz_external 38.33.20.159:1796 -> CLIENT.NET.4.18:79 3002
2001/05/27 09:00:30 sen_dmz_external 38.33.20.159:1838 -> CLIENT.NET.4.19:79 3002
2001/05/27 09:00:31 sen_dmz_external 38.33.20.159:1872 -> CLIENT.NET.4.21:79 3002
2001/05/27 09:00:31 sen_dmz_external 38.33.20.159:1854 -> CLIENT.NET.4.20:79 3002
2001/05/27 09:00:32 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.20:80 3045
2001/05/27 09:00:33 sen_dmz_external 38.33.20.159:1936 -> CLIENT.NET.4.22:79 3002
2001/05/27 09:00:35 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.14:80 3045
2001/05/27 09:00:36 sen_dmz_external 38.33.20.159:2034 -> CLIENT.NET.4.23:79 3002
2001/05/27 09:00:36 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.18:80 3045
2001/05/27 09:00:36 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.17:80 3045
2001/05/27 09:00:37 sen_dmz_external 38.33.20.159:2051 -> CLIENT.NET.4.24:79 3002
2001/05/27 09:00:38 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.19:80 3045
2001/05/27 09:00:44 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.24:80 3045
2001/05/27 09:01:27 sen_dmz_external 38.33.20.159:3502 -> CLIENT.NET.4.25:79 3002
2001/05/27 09:01:30 sen_dmz_external 38.33.20.159:3534 -> CLIENT.NET.4.26:79 3002
2001/05/27 09:01:30 sen_dmz_external 38.33.20.159:3550 -> CLIENT.NET.4.27:79 3002
2001/05/27 09:01:32 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.26:80 3045
2001/05/27 09:01:33 sen_dmz_external 38.33.20.159:3603 -> CLIENT.NET.4.28:79 3002
2001/05/27 09:01:34 sen_dmz_external 38.33.20.159:3633 -> CLIENT.NET.4.29:79 3002
2001/05/27 09:01:34 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.25:80 3045
2001/05/27 09:01:34 sen_dmz_external 38.33.20.159:3663 -> CLIENT.NET.4.30:79 3002
2001/05/27 09:01:40 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.29:80 3045
2001/05/27 09:01:43 sen_dmz_external 38.33.20.159:3822 -> CLIENT.NET.4.30:80 3200
2001/05/27 09:01:44 sen_dmz_external 38.33.20.159:3854 -> CLIENT.NET.4.26:80 3226
2001/05/27 09:01:46 sen_dmz_external 38.33.20.159:3951 -> CLIENT.NET.4.30:80 3213
2001/05/27 09:01:48 sen_dmz_external 38.33.20.159:4076 -> CLIENT.NET.4.30:80 3217
2001/05/27 09:01:48 sen_dmz_external 38.33.20.159:4044 -> CLIENT.NET.4.31:80 3213
2001/05/27 09:01:48 sen_dmz_external 38.33.20.159:4052 -> CLIENT.NET.4.32:80 3213
2001/05/27 09:01:49 sen_dmz_external 38.33.20.159:4083 -> CLIENT.NET.4.31:80 3217
2001/05/27 09:01:49 sen_dmz_external 38.33.20.159:4085 -> CLIENT.NET.4.32:80 3217
2001/05/27 09:01:50 sen_dmz_external 38.33.20.159:4103 -> CLIENT.NET.4.26:80 5040
2001/05/27 09:01:51 sen_dmz_external 38.33.20.159:4110 -> CLIENT.NET.4.26:80 5054
2001/05/27 09:01:53 sen_dmz_external 38.33.20.159:4153 -> CLIENT.NET.4.31:80 3226
2001/05/27 09:01:57 sen_dmz_external 38.33.20.159:4286 -> CLIENT.NET.4.33:79 3002
2001/05/27 09:01:59 sen_dmz_external 38.33.20.159:4358 -> CLIENT.NET.4.31:80 5040
2001/05/27 09:01:59 sen_dmz_external 38.33.20.159:4317 -> CLIENT.NET.4.30:80 5040
2001/05/27 09:01:59 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.33:80 3045
2001/05/27 09:02:00 sen_dmz_external 38.33.20.159:4370 -> CLIENT.NET.4.32:80 5040
2001/05/27 09:02:04 sen_dmz_external 38.33.20.159:4502 -> CLIENT.NET.4.33:80 3200
2001/05/27 09:02:04 sen_dmz_external 38.33.20.159:4509 -> CLIENT.NET.4.34:79 3002
2001/05/27 09:02:05 sen_dmz_external 38.33.20.159:4557 -> CLIENT.NET.4.35:79 3002
2001/05/27 09:02:06 sen_dmz_external 38.33.20.159:4573 -> CLIENT.NET.4.36:79 3002
2001/05/27 09:02:11 sen_dmz_external 38.33.20.159:4683 -> CLIENT.NET.4.33:80 3226
2001/05/27 09:02:11 sen_dmz_external 38.33.20.159:4685 -> CLIENT.NET.4.34:80 3200
2001/05/27 09:02:13 sen_dmz_external 38.33.20.159:4703 -> CLIENT.NET.4.35:80 3200
2001/05/27 09:02:13 sen_dmz_external 38.33.20.159:4711 -> CLIENT.NET.4.36:80 3200
2001/05/27 09:02:13 sen_dmz_external 38.33.20.159:4706 -> CLIENT.NET.4.34:80 3213
2001/05/27 09:02:16 sen_dmz_external 38.33.20.159:4738 -> CLIENT.NET.4.33:80 5054
2001/05/27 09:02:16 sen_dmz_external 38.33.20.159:4742 -> CLIENT.NET.4.36:80 3217
2001/05/27 09:02:17 sen_dmz_external 38.33.20.159:4764 -> CLIENT.NET.4.34:80 3226
2001/05/27 09:02:18 sen_dmz_external 38.33.20.159:4771 -> CLIENT.NET.4.35:80 3226
2001/05/27 09:02:19 sen_dmz_external 38.33.20.159:4776 -> CLIENT.NET.4.36:80 3226
2001/05/27 09:02:19 sen_dmz_external 38.33.20.159:4820 -> CLIENT.NET.4.37:79 3002
2001/05/27 09:02:20 sen_dmz_external 38.33.20.159:4834 -> CLIENT.NET.4.34:80 5040
2001/05/27 09:02:20 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.37:80 3045
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
2001/05/27 09:02:21 sen_dmz_external 38.33.20.159:4838 -> CLIENT.NET.4.34:80 5054
2001/05/27 09:02:21 sen_dmz_external 38.33.20.159:4840 -> CLIENT.NET.4.35:80 5040
2001/05/27 09:02:25 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.39:80 3045
2001/05/27 09:02:26 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.40:80 3045
2001/05/27 09:02:26 sen_dmz_external 38.33.20.159:4970 -> CLIENT.NET.4.37:80 3200
2001/05/27 09:02:26 sen_dmz_external 38.33.20.159:4981 -> CLIENT.NET.4.41:79 3002
2001/05/27 09:02:31 sen_dmz_external 38.33.20.159:1110 -> CLIENT.NET.4.39:80 3200
2001/05/27 09:02:32 sen_dmz_external 38.33.20.159:1116 -> CLIENT.NET.4.38:80 3217
2001/05/27 09:02:32 sen_dmz_external 38.33.20.159:1130 -> CLIENT.NET.4.40:80 3200
                          (388 lines omitted)
2001/05/27 09:25:41 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.250:80 3045
2001/05/27 09:25:43 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.251:80 3045
2001/05/27 09:26:13 sen_dmz_external 38.33.20.159:3633 -> CLIENT.NET.4.252:79 3002
2001/05/27 09:26:19 sen_dmz_external 38.33.20.159:3721 -> CLIENT.NET.4.253:79 3002
2001/05/27 09:26:20 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.252:80 3045
2001/05/27 09:26:25 sen_dmz_external 38.33.20.159:5 -> CLIENT.NET.4.253:80 3045
```

1. Source of Trace:

Network of one of our clients for whom we monitor their network(s).

2. Detect was generated by:

Cisco Secure IDS sensors at the client's premises.

The fields you see in the log are the following:

- date
- time
- sensor
- source address : source port
- destination address : destination port
- exploit signature number (see §4: description of attack)

3. Probability the source address was spoofed:

Not likely. The HTTP exploits need an established connection and t here are only few scanning tools that can spoof their source address. Besides that, the host, port and vulnerability scans all come from the same source.

4. Description of attack:

This is a combination of host scans (looking for hosts that are up), port scans (looking for ports that are listening) and vulnerability scans (looking for usable exploits). It even might include actually using an exploit, but that is difficult to tell without actual packet dumps.

The used techniques are the following:

3002		TCP SYN Port Sweep
3040		NULL TCP Packet
3045	CAN-1999-0454	Queso Sweep
3200	CVE-1999-0067	WWW Phf Attack
3213	CVE-1999-0070	WWW TEST -CGI Attack
3217	CAN-1999-0238	WWW php View File Attack
3226	CVE-1999-0039	WWW Webdist Bug
5040	CAN-1999-0509	WWW Perl Interpreter Attack
5054	CVE-1999-0953	WWWBoard Password

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

5. Attack mechanism:

A combination of different techniques is used here. Looking at the speed, we can be sure that it is some kind of script.

At the highest level, we see that the script is doing a host scan against the complete range (it tries every host; there are some hosts missing, but probably the sensor (or the analyzer) had troubles keeping up).
For each host, the script does a SYN port scan (signature 3002): the external logs show destination port 79 (finger), but much more ports are tried. Remark that these alerts aren't seen behind the firewall (in the DMZ) —the firewall blocks them.

If it gets some kind of response (with other words: if it knows the host is up), it will do a Queso fingerprint (signature 3045). This is done at destination port 80. Notice that these packets pass the firewall, regardless whether an HTTP server is listening or not. I don't think this is due to a bad rulebase, but to a firewall that doesn't strictly follow the RFC.

If it gets response on port 80 (HTTP), the script checks several vulnerabilities, all targeting UNIX webservers —which isn't that bad, knowing that the webservers run Solaris. Remark though that it doesn't do a perfect job, because the webdist vulnerability is only applicable to IRIX systems.

6. Correlations:

The different alerts could not be related to any other event for this client in the past.

For the CGI vulnerability, I refer to the detect 1.
Queso is quite popular and can easily be found on the web. If you want some, try the following URL: http://www.sans.org/searchsans?p=1&lang=en&mode=all&q=queso (this is actually a search for Queso on the SANS site).

7. Evidence of active targeting:

Not really. The attacker is looking for vulnerable webservers: (s)he sweeps a complete network to find them and when (s)he does, (s)he tries some vulnerabilities.

8. Severity:

Criticality	: 3	really a big company; they have many websites (mirrored over the world) and if one gets defaced, it won't put them out of business
Lethality	: 1	just looking around and playing a little bit; even if (s)he was really trying to bring damage upon the company, it wouldn't matter: they aren't using any of the programs to make it work
System Countermeasures	: 3	patches applied, not sure whether the OS is hardened
Network Countermeasures	: 3	there are firewalls, but some packets got through although they shouldn't have!

Severity = (Criticality + Lethality) – (System Countermeasures + Network Countermeasures)
= (3 + 1) – (3 + 3)
= –2

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

9. Defensive recommendation:

It is remarkable that the firewall allows inbound traffic to port 80 to machines that aren't running an HTTP server. Besides that, Queso is using mostly flag settings that aren't allowed by the RFC. A firewall shouldn 't let them through.
I recommend taking a look at the firewall's rule base and check whether or not it is possible to force RFC compliant behavior (which would make the OS fingerprinting a lot more difficult).

10. Multiple choice test question:

```
2001/05/27 08:59:28 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.3:80
2001/05/27 08:59:39 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.11:80
2001/05/27 08:59:28 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.4:80
2001/05/27 08:59:38 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.10:80
2001/05/27 08:59:38 sen_dmz_internal 38.33.20.159:5 -> CLIENT.NET.4.9:80
```

Seeing the above output, which tool most likely generated this traffic?

A. telnet
B. queso
C. idlescan
D. all of the above

The correct answer is B: telnet would us e another source port per connection (and it would normally not be lower than 1024). Using Idlescan, the source and destination ports would be the same.

Detect 3

```
05/20-21:44:45.389134  [**] IDS181  - MISC - Shellcode X86 NOPS [**]
    212.142.9.23:2214  -> MY.NET.68.170:80

[**] IDS181  - MISC - Shellcode X86 NOPS [**]
05/20-21:44:45.389134 0:B0:64:12:8F:60  -> 8:0:20:A7:10:E3 type:0x800
len:0x4D4
212.142.9.23:2214  -> MY.NET.68.170:80 TCP TTL:44 TOS:0x10 ID:19068
IpLen:20 DgmLen:1222 DF
***AP*** Seq: 0x182BDF9  A ck: 0xCDE014F  Win: 0x7D78  TcpLen: 20
47 45 54 20 2F 4E 55 4C 4C 2E 70 72 69 6E 74 65  GET /NULL.printe
72 20 48 54 54 50 2F 31 2E 30 0D 0A 42 65 61 76  r HTTP/1.0..Beav
75 68 3A 20 90 90 90 90 90 90 90 90 90 90 90 90  uh: ............
90 90 90 90 90 90 9 0 90 EB 03 5D EB 05 E8 F8 FF  .........].....
FF FF 83 C5 15 90 90 90 8B C5 33 C9 66 B9 D7 02  ..........3.f...
50 80 30 95 40 E2 FA 2D 95 95 64 E2 14 AD D8 CF  P.0.@..  -..d.....
05 95 E1 96 DD 7E 60 7D 95 95 95 95 C8 1E 40 14  .....~`}......@.
7F 9A 6B 6A  6A 1E 4D 1E E6 A9 96 66 1E E3 ED 96  ..kjj.M....f....
66 1E EB B5 96 6E 1E DB 81 A6 78 C3 C2 C4 1E AA  f....n....x.....
96 6E 1E 67 2C 9B 95 95 95 66 33 E1 9D CC CA 16  .n.g,....f3.....
52 91 D0 77 72 CC CA CB 1E 58 1E D3 B1 96 56 44  R..wr....X....VD
74 96 54 A6 5C F3 1E 9D 1E D3 89 96 56 54 74 97  t.T.  \.......VTt.
96 54 1E 95 96 56 1E 67 1E 6B 1E 45 2C 9E 95 95  .T...V.g.k.E,...
95 7D E1 94 95 95 A6 55 39 10 55 E0 6C C7 C3 6A  .}.....U9.U.l..j
C2 41 CF 1E 4D 2C 93 95 95 95 7D CE 94 95 95 52  .A..M,....}.  ...R
D2 F1 99 95 95 95 52 D2 FD 95 95 95 95 52 D2 F9  ......R......R..
94 95 95 95 FF 95 18 D2 F1 C5 18 D2 85 C5 18 D2  ................
81 C5 6A C2 55 FF 95 18 D2 F1 C5 18 D2 8D C5 18  ..j.U...........
D2 89 C5 6A C2 55 52 D2 B5 D1 95 95 95 18 D2 B5  ...j  .UR.........
C5 6A C2 51 1E D2 85 1C D2 C9 1C D2 F5 1E D2 89  .j.Q............
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
1C D2 CD 14 DA D9 94 94 95 95 F3 52 D2 C5 95 95  ...........R....
18 D2 E5 C5 18 D2 B5 C5 A6 55 C5 C5 C5 FF 94 C5  .........U......
C5 7D 95 95 95 95 C8 14 78 D5 6B 6A 6A C0 C5   6A  .}......x.kjj..j
C2 5D 6A E2 85 6A C2 71 6A E2 89 6A C2 71 FD 95  .]j..j.qj..j.q..
91 95 95 FF D5 6A C2 45 1E 7D C5 FD 94 94 95 95  .....j.E.}......
6A C2 7D 10 55 9A 10 3F 95 95 95 A6 55 C5 D5 C5  j.}.U..?....U...
D5 C5 6A C2 79 16 6D 6A 9A 11 02 95 9 5 95 1E 4D  ..j.y.mj.......M
F3 52 92 97 95 F3 52 D2 97 95 80 52 D2 91 41 1B  .R....R....R..A.
9C 82 FF 85 18 92 C5 C6 6A C2 61 FF A7 6A C2 49  ........j.a..j.I
A6 5C C4 C3 C4 C4 C4 6A E2 81 6A C2 59 10 55 E1  .  \.....j..j.Y.U.
F5 05 05 05 05 15 AB 95 E1 BA  05 05 05 05 FF 95  ................
C3 FD 95 91 95 95 C0 6A E2 81 6A C2 4D 10 55 E1  .......j..j.M.U.
D5 05 05 05 05 FF 95 6A A3 C0 C6 6A C2 6D 16 6D  .......j...j.m.m
6A E1 BB 05 05 05 05 7E 27 FF 95 FD 95 91 95 95  j......~'.......
C0 C6 6A C2 69 10 55  E9 8D 05 05 05 05 E1 09 FF  ..j.i.U.........
95 C3 C5 C0 6A E2 8D 6A C2 41 FF A7 6A C2 49 7E  ....j..j.A..j.I~
1F C6 6A C2 65 FF 95 6A C2 75 A6 55 39 10 55 E0  ..j.e..j.u.U9.U.
6C C4 C7 C3 C6 6A 47 CF CC 3E 77 7B 56 D2 F0 E1  l....jG..>w{V...
C5 E7 FA F6 D 4 F1 F1 E7 F0 E6 E6 95 D9 FA F4 F1  ................
D9 FC F7 E7 F4 E7 EC D4 95 D6 E7 F0 F4 E1 F0 C5  ................
FC E5 F0 95 D2 F0 E1 C6 E1 F4 E7 E1 E0 E5 DC FB  ................
F3 FA D4 95 D6 E7 F0 F4 E1 F0 C5 E7 FA F6 F0 E6  ................
E6 D4 95 C5 F0 F0 FE DB F4 F8 F0 F1 C5 FC E5 F0  ................
95 D2 F9 FA F7 F4 F9 D4 F9 F9 FA F6 95 C2 E7 FC  ................
E1 F0 D3 FC F9 F0 95 C7 F0 F4 F1 D3 FC F9 F0 95  ................
C6 F9 F0 F0 E5 95 D0 ED FC E1 C5 E7 FA F6 F0 E6  ..............  ..
E6 95 D6 F9 FA E6 F0 DD F4 FB F1 F9 F0 95 C2 C6  ................
DA D6 DE A6 A7 95 C2 C6 D4 C6 E1 F4 E7 E1 E0 E5  ................
95 E6 FA F6 FE F0 E1 95 F6 F9 FA E6 F0 E6 FA F6  ................
FE F0 E1 95 F6 FA FB FB F0 F6 E1 95 E6 F0 FB F1  ......  ........
95 E7 F0 F6 E3 95 F6 F8 F1 BB F0 ED F0 95 0D 0A  ................
48 6F 73 74 3A 20 90 90 90 90 90 90 90 90 90 90  Host: ..........
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90    ..............
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90   90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 9  0 90  90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90  90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90  90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 33  ...............3
C0 B0 90 03 D8 8B 03 8B 40 60 33 DB B3 24 03 C3  ........@`3..$..
FF E0 EB B9 90 90 05 31 8C 6A 0D 0A 0D 0A        .......1.j....
```

1. Source of Trace:

An IDS at our network, just in front of the corporate firewall.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

2. Detect was generated by:

Snort IDS (version 1.7).

3. Probability the source address was spoofed:

It is quite unlikely that the source address was spoofed. This attack requires an established connection, so spoofing the source address wouldn't work. What might be possible though, is that the host (of which we see the source address) is compromised and the real attacker is hiding his real address by using this compromised host. Another possibility is that the source we see is an HTTP proxy server, which would automatically hide the attacker's real IP address. However, looking at the actual packet, there is no indication that a proxy is in use, but you can configure proxies not to show for whom it is proxying (no "*X-Forwarded-For*" field).

4. Description of attack:

This is a buffer overflow exploit for MS IIS 5.0 running on MS Windows 2000.

CAN-2001-0241 Buffer overflow in Internet Printing ISAPI extension in Windows 2000 allows remote attackers to gain root privileges via a long print request that is passed to the extension through IIS 5.0.

5. Attack mechanism:

The attacker is trying to send a specially crafted packet that will result in a buffer overflow on an MS IIS 5.0 running on a Windows 2000 server, which could give him system level privileges.

This vulnerability stems from an unchecked buffer that exists in the ISAPI extension that deals with IPP (Internet Printing Protocol). The IPP extension will allow for example remote users to send print jobs to the webserver to be printed over the Internet using either HTTP or HTTPS. There exists however an unchecked buffer that is used to contain the users' print requests. Therefore a malicious user can simply craft a specially formatted request to the webserver and when processed, overflow the allocated buffer and have the ability to run arbitrary code of his/her own choosing.

It is worth mentioning that an alert was triggered, not because of the ".printer" string in the request (which is the most recognizable item for this exploit), but because of the many NOPs present in this packet. We are running an IIS 4.0 on a hardened Windows NT 4.0, for which this exploit wouldn't work, so it wasn't necessary to add this specific alert.

6. Correlations:

This event couldn't be correlated to any other event seen by our Snort IDS.

I didn't find any references on the Web to someone using this exploit against their server, but I found many references about the exploit, so I'm pretty sure someone tried this exploit against someone else. It is pretty recent (less than a month old), but also a very dangerous exploit and the code to exploit it, can be found easily. You can find the exploit and some more information on the following sites:

http://www.securiteam.com/exploits/IIS_5_0___printer__Exploit_Code_Released.html
http://www.sans.org/newlook/digests/SAC/windows.htm

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

7. Evidence of active targeting:

This is extreme active tar geting. The attacker is trying to exploit our webserver, using one very specific and very dangerous exploit.

8. Severity:

Criticality	: 5	this is our corporate webserver; as a security firm we would lose credibility if it was compromised, which could ha rm us quite a lot
Lethality	: 1	a very specific and dangerous attack, but harmless if you are running IIS 4.0 on NT.
System Countermeasures	: 5	all patches applied, OS completely hardened
Network Countermeasures	: 4	latest version of FW -1, good rule base (only inbound HTTP traffic allowed, nothing outbound allowed), but http content isn't stripped, so it will pass

Severity = (Criticality + Lethality) – (System Countermeasures + Network Countermeasures)
= (5 + 1) – (5 + 4)
= –3

9. Defensive recommendati on:

Defenses are fine. Attacks will not likely be successful if we always apply the very latest patches on our webserver and also update the firewall regularly.

10. Multiple choice test question:

Give the most correct answer. The packet dump shown above indicates:

A. a NOP attack
B. the .printer IIS 5.0 attack
C. regular (harmless) HTTP traffic
D. regular port 2214 traffic

The correct answer is B.

Detect 4

```
May 22 11:40:54 CLIENT iss: sensor_1  Glacier CUSTOMER.NET.229.21 80
    161.142.2.10 7718

                              XXXXXXXXXXXXX                          </td>
</tr>                <tr>           <td bgcolor="#DDE7D6"
align="center" valign="top" width="20">5.</td>           <td
bgcolor="#DDE7D6" valign="top">             <a
href="http://XXXXXXXXXXXXXXXXXXXXXXXXX  XXXXXXXXXXXXXXXXXXXXX">XXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX  </a>
<br><b>XXXXXXXXX</b> XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX   XXX
XXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX   .
          <br><b>Type:</b> HTML
<br><b>XXXXX</b>                                  XXXXXXXXXXXXX
    </td>    </tr>                <tr>           <td
bgcolor="#FFFFFF" align="center" valign="top" width="20">6.</td>
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
    <td bgcolor="#FFFFFF" valign="top">            <a
href="http://XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX">XXXXXXXXX
XXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX </a>
<br><b>XXXXXXXXXXX</b> XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXX .                    <br><b>XXXXX</b>
HTML                          <br><b>XXXXX</b>
```

1. Source of Trace:

Network of one of our customers for whom we monitor their network(s).

2. Detect was generated by:

RealSecure (version 5.0 X -Press Update 2.3).

3. Probability the s ource address was spoofed:

It is quite unlikely that the source address was spoofed. This attack requires an established connection, so spoofing the source address wouldn't work.

4. Description of attack:

This is regular HTTP traffic, not an attack.

5. Attack mechanism:

During a normal web query the client (normally using a port above 1023) will contact a webserver on port 80 (HTTP) or 443 (HTTPS) with a SYN packet, complete the3 way handshake by receiving a SYN -ACK and sending an ACK and then send the query and receive the answer.

6. Correlations:

We get false positives all the time for RealSecure and not only for this type of alert. Many alerts in RealSecure are only based on the destination port, which is prone to produce many false positives.

You can find lots of false positive detects on the Web. One, comparable to the one shown above, can be found at the top of http://www.sans.org/y2k/070800.htm .

7. Evidence of active targeting:

Yes, it was sen d to a specific host, although no harm was intended (the client just took as source port the destination port used for a Glacier attack (7718)).

8. Severity:

Criticality	: 2	if this would be an attack, it is outbound traffic, so none of our servers are at risk; however, in that case, the targeted company might take some actions against us

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Lethality	: 0	it's not an attack
System Countermeasures	: 5	all patches applied, OS completely hardened; this specific client is really paranoid (as it should be)
Network Countermeasures	: 4	knowing what we know about their webservers, I guess their firewall (FW-1) is very restrictive, hardened and completely patched, but we don't have access, so I couldn't check it and hence I am not 100% sure

Severity = (Criticality + Lethality) – (System Countermeasures + Network Countermeasures)
= (3 + 0) – (5 + 4)
= –6

9. Defensive recommendation:

Not needed. There wasn't an attack and their countermeasures are very good.

10. Multiple choice test question:

Seeing only the following, what could you conclude knowing that CUSTOMER.NET.229.21 is a webserver?

```
May 22 11:40:54 CLIENT iss: sensor_1  Glacier CUSTOMER.NET.229.21 80
    161.142.2.10 7718
```

A. most likely a glacier attack
B. most likely a false positive
C. most likely a false negative
D. none of the above

The correct answer is B: a connection to destination port 7718 wouldn't use port 80 as source port when a service is listening on it (webserver).

Detect 5

```
05/21-10:54:18.221703  [**] Napster Client Data [**]
    MY.NET.80.1:46054 -> 66.26.171.113:6699
05/21-10:55:16.557973  [**] Napster Client Data [**]
    MY.NET.80.1:46132  -> 64.255.196.237:6699

[**] Napster Client Data [**]
05/21-10:54:18.221703 0:4:27:B4:EB:0  -> 0:B0:64:12:8F:60 type:0x800
len:0x7A
MY.NET.80.1:46054  -> 66.26.171.113:6699 T CP TTL:126 TOS:0x0 ID:39374
IpLen:20 DgmLen:108 DF
***AP*** Seq: 0x2F4B3D44  Ack: 0x951BEE  Win: 0x446F  TcpLen: 20
XX XX XX XX XX XX XX 20 22 43 3A 5C 50 72 6F 67  XXXXXXX "C:  \Prog
72 61 6D 20 46 69 6C 65 73 5C 4E 61 70 73 74 65  ram Files  \Napste
72 5C 4D 79 20 46 69 6C 65 73 5C 44 61 76 69 64  r  \My Files\David
20 42 6F 77 69 65 20 2D 20 46 61 6D 65 2E 6D 70   Bowie   - Fame.mp
33 22 20 30                                      3" 0

 [**] Napster Client Data [**]
05/21-10:55:16.557973 0:4:27:B4:EB:0  -> 0:B0:64:12:8F:60 type:0x800
len:0x8B
MY.NET.80.1:46132  -> 64.255.196.237:6699 TCP TTL:126 TOS:0x0 ID:40556
IpLen:20 DgmLen:125 DF
***AP*** Seq: 0x2FCDE418  Ack: 0x288360EE  Win: 0x40E7  TcpLen: 20
7A 75 6D 62 69 6B 65 20 22 63 3A 5C 70 72 6F 67  XXXXXXX "c:  \prog
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
72 61 6D 20 66 69 6C 65 73 5C 6E 61 70 73 74 65  ram files  \napste
72 5C 6D 79 20 66 69 6C 65 73 5C 44 61 76 69 64  r  \my files\David
20 42 6F 77 69 65 20 46 61 6D 65 20 28 6F 72 69   Bowie Fame (ori
67 69 6E 61 6C 20 76 65 72 73 69 6F 6E 29 2E 6D  ginal ve  rsion).m
70 33 22 20 30                                   p3" 0
```

1. Source of Trace:

An IDS at our network, just in front of the corporate firewall.

2. Detect was generated by:

Snort IDS (version 1.7).

3. Probability the source address was spoofed:

It is quite unlikely that the source address was spoofed. This attack requires an established connection, so spoofing the source address wouldn't work.

4. Description of attack:

This is not an attack, but is not risk free traffic either. There are known vu lnerabilities for Napster (CAN-2000-0281 and CAN-2000-0412) and the file sharing in itself could be dangerous. There exist a program, called wrapster, which can pack every file in a wrapped file which looks like an mp3 file (having a bit rate of 32kps and a frequency of 32kHz). In this way, you could share every file.

5. Attack mechanism:

For a complete description on how Napster works, see http://david.weekly.org/code/napster.php3 . The "Requesting a File" part of this description is shown here for your c onvenience:

```
        SENT
            2A 00 CB 00 username
                "C:\MP3\REM - Everybody Hurts.mp3"
        RECEIVED
            5D 00 CC 00 username
                2965119704 (IP-address backward-form = A.B.C.D)
                6699 (port)
                "C:\MP3\REM - Everybody Hurts.mp3" (song)
                (32-byte checksum)
                (line speed)
        [connect to A.B.C.D:6699]
        RECEIVED from client
            31 00 00 00 00 00
        SENT to client
            GET
        RECEIVED from client
            00 00 00 00 00 00
*       SENT to client
*           Myusername
*           "C:\MP3\REM - Everybody Hurts.mp3"
*           0 (port to connect to)
        RECEIVED from client
            (size in bytes)
        SENT to server
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

00 00 DD 00 (give the go -ahead thru server)
RECEIVED from client
[DATA]

The part with the asterisks in front is the packet seen by the Snort IDS.

6. Correlations:

There were two more alerts in the snort output I've analyzed from our ne twork.

For related incidents, just go to
http://www.sans.org/searchsans/perlfect/search/search.pl?lang=en&mode=all&q=napster .

7. Evidence of active targeting:

Yes. This traffic was specifically targeted at the destination host, because that was the hos t that had the file the source host was looking for.

8. Severity:

Criticality	: 1	if this would be an attack, it is outbound traffic, so none of our servers are at risk
Lethality	: 2	it's not an attack and he is not sending a file out
System Countermea sures	: 5	all patches applied, OS completely hardened
Network Countermeasures	: 4	latest version of FW -1, good rule base (only inbound HTTP traffic allowed, nothing outbound allowed), but http content isn't stripped, so it will pass

Severity = (Criticality + Lethality) – (System Countermeasures + Network Countermeasures)
= (5 + 4) – (5 + 4)
= 0

Criticality	: 2	if this would be an attack, it is outbound traffic, so none of our servers are at risk; however, in that case, the targeted company might take so me actions against us
Lethality	: 0	it's not an attack
System Countermeasures	: 5	all patches applied, OS completely hardened; this specific client is really paranoid (as it should be)
Network Countermeasures	: 4	knowing what we know about their webserver s, I guess their firewall (FW -1) is very restrictive, hardened and completely patched, but we don't have access, so I couldn't check it and hence am not 100% sure

Severity = (Criticality + Lethality) – (System Countermeasures + Network Countermeasures)
= (3 + 0) – (5 + 4)
= –6

9. Defensive recommendation:

It would be wiser to disallow P2P traffic (napster, gnutella, etc…) because of the inherent risk (file sharing, which could mean any file when using wrapster) and the known vulnerabilities, which allow denial of service attacks and reading arbitrary (confidential) files on the client system by specifying the full pathname for the files. As a side effect, you might also see a (tremendous) decline in bandwidth usage.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

10. Multiple choice test question:

Using Napster is:

A. a security risk
B. possibly bandwidth consuming
C. cheap compared to actually buying the records
D. all of the above

The correct answer is D.

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Assignment 2 – Describe the State of Intrusion Detection

Give the URL, location or command that you acquired the attack from.

I got this from a friend, who got it from Arthur Donkers, the person who discovered this new exploit. It is extremely recent (discovered only one week ago, on Wednesday May 22, 2001).

Arthur Donkers is the founder and president of Le Reseau Netwerksystemen B.V. (*Le Reseau* is French for *The Network*; *Netwerksystemen* is Dutch for *Networking Systems*; *B.V.* is something like *Inc.*). It is a Dutch company that offers security services, especially for the high-end market. For more info, I refer to http://www.reseau.nl , but unless you speak Dutch, you won't get much wiser…

Arthur Donkers is also one of the writers of *SysAdmin*, a monthly magazine that concentrates on Unix and applications for it…

He has his own honeypot and it is there that he saw someone using this new, advanced worm doing its work…

Describe the attack, including how it works.

This Internet worm uses the recently discovered bind TSIG vulnerability (CVE -2001-0010) to compromise systems. This vulnerability affects many versions of bind and also many different platforms. More information about this vulnerability can be found at:

- http://www.cert.org/advisories/CA-2001-02.html
- http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0010
- http://www.securityfocus.com/bid/2302

I will now describe what this specific worm does exactly…

Once a new system has been compromised, using the exploit mentioned above, the worm will download itself from the system it has been scanning from. This is done trough a small, by the worm installed, HTTP server that listens on TCP port 12321. Before downloading the files the worm first checks if wget or lynx have been installed. If so then one of these applications is used to retrieve the files. If these files are not installed the worm will create a program called '*get*' and use this program to download the binaries.

Three packages have to be downloaded: *stuff.tgz*, *sshd.tgz* and *named.tgz*. The first package that the worm downloads is *stuff.tgz*. This is the main package of the worm, containing the spreading tools and a rootkit. The rootkit includes *adore* which is a kernel module package that can hide processes and files. For more information about the adore rootkit, I refer to http://www.sans.org/y2k/practical/Michael_Reiter_GCIH.zip . This file contains a document by Michael Reiter which describes the working of the adore rootkit.

After the stuff package has been unpacked, a script *'tr0jan'* is executed which will:

- Create a temporary backdoor on port 1524 TCP
- Download and install the *sshd.tgz* and the *named.tgz* package.
- Compile, install and activate the rootkit with the adore kernel modules.
- Activate the spreading mechanism (ip generator, scanner and exploit).
- Replace the vulnerable bind binary with a patched version.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

- Remove the temporary backdoor and installs an SSH server in /tmp/.ssh/.

The SSH backdoor listens on port 12345 TCP, which is a well known trojan port on which also e.g. NetBus and Ashley listen. The SSH server has a hard coded password that the creators of the worm can use to gain root level access on the compromised system. This password is *'h4ck3d'*.

Now the worm replaces several system files for in case that adore does not work properly, it can reveal its existence. The replaced files are *du*, *ps*, *ls* and *klogd*. And as last action the worm adds itself to the system startup files to make sure it will be activated at boot time.

The methods used to propagate and compromise are very affective and that makes this worm very dangerous. Even tough the bind vulnerability used by the worm is well known (as mentioned before, CVE-2001-0010), there are still many vulnerable systems on the Internet and they form a very easy target.

Interesting to notice is that the adore package used by the worm, originally was created for the whitehat community (the good guys). Unfortunately blackhats (the bad guys) are very aware of its capabilities resulting that this package has been includes in several rootkit, auto rooters and worms lately.

Provide an annotated network trace of the attack in action.

What I would like to show here is how an infected system infects another not already infected system. For this, I will use 2 simultaneous tcpdump sessions: one on the target machine and one on the already infected system (the attacker).

Both machines have the following configuration:

- intel architecture
- RedHat 7.0 basic install with Bind DNS server version 8.2.2-p5 running (this version is vulnerable for vulnerablility CVE-2001-0010, which is necessary because this worm uses it).
- Modular kernel version 2.2.16

Unfortunately, I couldn't get the first system infected, so I am not able to show a net trace. Maybe I did something wrong with the scripts, or there actually was something wrong with the scripts, but I didn't have enough time anymore to get it working...

If I could have shown a working net trace, this attacking mechanism would have shown how great a threat it really is. Adore is known to be a very good mechanism of hiding your trojan, and the trojan gives root access to your machine (this worked indeed, but there were some other problems), so it might take a very long time before you notice your DNS server is compromised…

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Assignment 3 – Analyze This Scenario

Introduction

We were asked to provide an intrusion analysis for GIAC Enterprises. Therefor we have been given one month's worth of data from a Snort Intrusion Detection System with a fairly standard rulebase.

Snort output

We do not have all the data, because now and then there were power failures or the disks were full. A complete analysis of all events is thus not possible, but because of the amount of data received, we will be able to give a good impression of hostile activity.

Three data files were found twice. They were removed so they wouldn't falsify statistical information. There were also some files, which appeared to be generated in the year 2000. Because the snort output doesn't include a year (only month and day and the hour, up to the microsecond) and the dates (when not considering the year) fell between the other dates, we presumed the data was gathered in 2001. Those files were included in the analysis.

The data contains information between January 20, 2001 and March 12, 2001, but as mentioned before, there are gaps.

Three different kinds of files need to be considered:

- alert files
- scan files
- packet logs

The alert files give an overview of all the events that triggered an alert based on the used snort rulebase. This includes some general scan information, but for the scan attempt details you need to consult the scan files, which only contain scan information. The packet logs are dumps of the actual network traffic, seen and interpreted by the snort system.

Statistical analysis of the alert files

First we will give some statistical information concerning the alert files.

Table 3.1 shows the distribution of the alerts by their type. This is an exhaustive enumeration of all the kind of alerts that were detected by the Snort IDS during the monitoring period.

Table 3.1: Distribution of alerts by alert type

# alerts	percent	alert type
436882	73.10	UDP SRC and DST outside network
105880	17.72	portscan related
15021	2.51	Watchlist 000220 IL-ISDNNET-990517
11608	1.94	SYN-FIN scan!
9914	1.66	Possible RAMEN server activity
5728	0.96	Watchlist 000222 NET-NCFC
4818	0.81	NMAP TCP ping!
1722	0.29	TCP SRC and DST outside network
1517	0.25	External RPC call

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

1155	0.19	SNMP public access
729	0.12	SMB Name Wildcard
591	0.10	connect to 515 from inside
543	0.09	Attempted Sun RPC high port access
499	0.08	WinGate 1080 Attempt
469	0.08	Queso fingerprint
229	0.04	Tiny Fragments - Possible Hostile Activity
135	0.02	Null scan!
112	0.02	SUNRPC highport access!
81	0.01	ICMP SRC and DST outside network
25	< 0.01	Back Orifice
8	< 0.01	STATDX UDP attack
4	< 0.01	TCP SMTP Source Port traffic
4	< 0.01	Security 000516 -1
2	< 0.01	Probable NMAP fingerprint attempt
1	< 0.01	SITE EXEC - Possible wu-ftpd exploit - GIAC000623
1	< 0.01	Russia Dynamo - SANS Flash 28-jul-00

One should be aware though that the percentages can't be considered absolutely correct, because of the slightly distorted image created by the way Snort registers scan related alerts. Normally, for each port scan, three or more entries in the alert logs can be found. When considering only the "PORTSCAN DETECTED" entries, there were 11656 port scans.

When observing Table 3.1 you should keep in mind that there were only 16 Snort alert files, corresponding to 16 days of monitoring. This means that at average, there were more than 37 thousand alerts a day! This corresponds to an alert every 2.3 seconds. This is huge, but fortunately, most of the alerts don't indicate a possibly compromised system.

One kind of alert that really stands out is the "UDP SRC and DST outside network" alert. This indicates that Snort saw a UDP packet from which (he thinks) the source and destination address are not part of your infrastructure. It is related to the "TCP SRC and DST outside network" alert, which is the same for TCP packets. These alerts indicate that you are routing traffic through your network for others. For most companies, this behavior isn't appropriate. We will go in depth into this later.

Table 3.2 shows the distribution by source IP address. Only the top 30 is shown. For your convenience, the hostname corresponding to the IP address is shown for each entry (*nslookup* information). If the hostname couldn't be retrieved, the name of the network block to which the host belongs is shown (*whois* information).

Table 3.2: Distribution of alerts by source IP address (top 30)

source IP address	# alerts	percent	host or network information
155.101.21.38	81304	16.54	bonfire.crsim.utah.edu
171.69.248.71	29058	5.91	tower-u1.cisco.com
140.142.19.72	28117	5.72	netfx.uwtv.washington.edu
206.190.54.67	20713	4.21	<< NET-NETBLK1-YAHOOBS -- Yahoo! Broadcast Services, Inc. >>
129.116.65.3	17549	3.57	vbrick1.ots.utexas.edu
63.250.208.169	17397	3.54	<< NETBLK-NETBLK2-YAHOOBS -- Yahoo! Broadcast Services, Inc. >>
128.223.83.33	17270	3.51	iptvhost.uoregon.edu
152.1.1.79	16121	3.28	ping.cc.ncsu.edu

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

130.235.13 3.92	15824	3.22	IPTVserver.ldc.lu.se
130.240.64.20	15736	3.20	fix.cdt.luth.se
171.68.98.109	13083	2.66	ottawa-dhcp-109.cisco.com
130.161.180.141	12346	2.51	tweetie.oli.tudelft.nl
171.68.43.192	9411	1.91	waukesha-dhcp-192.cisco.com
130.234.184.112	9336	1.90	termos.keltti.jyu.fi
128.171.104.147	8982	1.83	<< NET-HAWAII -- University of Hawaii >>
128.223.83.35	8884	1.81	icecast.uoregon.edu
130.225.127.87	8734	1.78	pc-127-087.adm.ku.dk
128.178.10.2	7685	1.56	fbpc1.epfl.ch
128.249.104.243	6559	1.33	bcm00392.ccit. bcm.tmc.edu
128.249.104.246	6487	1.32	bcm03409.ccit.bcm.tmc.edu
171.69.33.40	6315	1.28	<< NETBLK-NETBLK-CISCO-BBLOCK -- Cisco Systems, Inc. >>
159.226.81.1	5362	1.09	<< NET-NCFC -- The Computer Network Center Chinese Academy of Sciences >>
206.190.54.131	5300	1.08	<< NET-NETBLK1-YAHOOBS -- Yahoo! Broadcast Services, Inc. >>
MY.NET.70.38	4788	0.97	
137.224.18.7	4073	0.83	HKlompmaker2.lenD.wau.nl
212.179.41.169	4061	0.83	fr-c41169.bezeqint.net
63.105.122.6	3932	0.80	morrison.multicasttech.com
10.0.0.1	3867	0.79	[[RESERVED-6]]
130.240.4.100	3582	0.73	salmis.anl.luth.se
203.178.137.220	2618	0.53	palette.kyoto.wide.ad.jp

This table shows you who's interested in your systems. Sometimes you can see that some countries are particularly interested (e.g.: after a p olitical incident between two countries, it is possible that hackers of one country attack computer systems in the other one, cfr. the recent difficulties between the USA and China).

Most of the activity comes from universities:

- all .edu domains
- lu.se is the University of Lund (Sweden)
- luth..se is the Technical University of Lund (Sweden)
- jyu.fi is the University of Jyväskylä (Finland)
- tudelft.nl is the University of Technology in Delft (The Netherlands)
- ku.dk is the University of Copenhagen (Denmark)
- epfl.ch is the Federal Polytechnic School from Lausanne (Switzerland)
- wau.nl is the University for Life Sciences in Wageningen (The Netherlands)
- The Computer Network Center Chinese Academy of Sciences

The two most active companies are Cisco Systems, Inc., the number one supplier for network devices and Yahoo! Broadcast Services, Inc., well known for their Instant Messenger product.

Notice also that there is one server from your domain (MY.NET.70.38) in the list. Maybe this system is compromised, but it might a s well be a server which triggers a lot of false positives.

We will look further into this later.

Table 3.3 shows the distribution by destination IP address. Only the top 30 is shown. For your convenience, the hostname corresponding to the IP address is shown for each entry (*nslookup*

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

information). If the hostname couldn't be retrieved, the name of the network block to which the host belongs is shown (*whois* information).

Table 3.3: Distribution of alerts by destination IP address (top 30)

dest. IP address	# alerts	percent	host or network information
224.2.127.254	360172	73.28	SAP.MCAST.NET
233.28.65.197	20713	4.21	<< NET-MCAST-NET >>
233.28.65.255	17397	3.54	<< NET-MCAST-NET >>
224.0.1.41	12036	2.45	GATEKEEPER.MCAST.NET
MY.NET.6.47	5339	1.09	
233.40.70.199	5300	1.08	<< NET-MCAST-NET >>
MY.NET.213.250	4069	0.83	
10.255.255.255	3867	0.79	[[RESERVED-6]]
224.0.1.1	3703	0.75	NTP.MCAST.NET
169.254.255.255	2950	0.60	<< NETBLK-LINKLOCAL -- For use with Link Local Networks >>
MY.NET.207.226	2186	0.44	
233.28.65.223	2175	0.44	<< NET-MCAST-NET >>
MY.NET.209.114	1599	0.33	
1.1.1.1	1533	0.31	<< RESERVED-9 >>
MY.NET.207.126	1451	0.30	
24.67.186.244	1309	0.27	<< NETBLK-FIBERLINK-CABLE -- Shaw Fiberlink ltd. >>
192.168.0.255	1306	0.27	[[IANA-CBLK-RESERVED]]
24.48.226.183	1074	0.22	pa-southhills2a-695.pit.adelphia.net
MY.NET.100.99	872	0.18	
MY.NET.220.42	792	0.16	
MY.NET.201.146	730	0.15	
162.129.112.40	656	0.13	jhwins01.jhoc1.jhmi.edu
MY.NET.222.2	619	0.13	
233.28.65.62	610	0.12	<< NET-MCAST-NET >>
216.181.129.185	573	0.12	<< NET-PRIMUSDSL-BLK1 -- PrimusDSL, Inc. >>
128.138.2.112	553	0.11	aden2-112-dhcp.resnet.Colorado.edu
172.16.1.103	534	0.11	[[IANA-BBLK-RESERVED]]
MY.NET.210.34	436	0.09	
MY.NET.217.42	413	0.08	
MY.NET.225.50	408	0.08	

This table gives an overview of the destination addresses of the traffic seen by the Snort IDS as alerting. It tells you in what systems (potential) hackers are interested.

Some remarks can be made already:

- We see a lot of multicast destinations (they take up more than 86% of all destinations)
- 1.1.1.1 is a very peculiar address to come along, because it is part of a reserved network (1.0.0.0/8 is reserved by the Internet Corporation for Assigned Names and Numbers) and shouldn't be used

Table 3.4 shows the distribution of alerts by destination IP address of which the destination belongs to the MY.NET network or a local network (the private IP address ranges 10.0.0.0 – 10.255.255.255, 172.16.0.0 – 172.31.255.255 and 192.168.0.0 – 192.168.255.255). Because we don't have host information for these networks, no hostnames can be given.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Table 3.4: Distribution of alerts by destination IP address, belonging to MY.NET (top 30)

dest. IP address	# alerts	percent
MY.NET.6.47	5339	1.09
MY.NET.213.250	4069	0.83
10.255.255.255	3867	0.79
MY.NET.207.226	2186	0.44
MY.NET.209.114	1599	0.33
MY.NET.207.126	1451	0.30
192.168.0.255	1306	0.27
MY.NET.100.99	872	0.18
MY.NET.220.42	792	0.16
MY.NET.201.146	730	0.15
MY.NET.222.2	619	0.13
172.16.1.103	534	0.11
MY.NET.210.34	436	0.09
MY.NET.217.42	413	0.08
MY.NET.225.50	408	0.08
MY.NET.217.206	403	0.08
MY.NET.223.254	362	0.07
MY.NET.222.94	321	0.07
MY.NET.211.74	304	0.06
MY.NET.60.11	301	0.06
MY.NET.202.246	296	0.06
MY.NET.204.22	273	0.06
MY.NET.224.34	263	0.05
10.1.11.101	217	0.04
MY.NET.217.98	209	0.04
MY.NET.6.7	207	0.04
MY.NET.225.186	154	0.03
MY.NET.223.214	146	0.03
MY.NET.100.206	144	0.03
MY.NET.221.246	135	0.03

This is a very interesting table, because it gives you an idea of which of *your* servers are targeted (or at best, involved in a lot of activity). Keep in mind that the percentages given are quite low because of the distorted picture that was created by the multicast traffic. If you would consider only your network (and the private networks), the percentages would be approximately nine times the percentage shown in table 3.4.

Maybe some of these systems are compromised by the attacks they were involved in, so they (certainly the top 10) should be investigated further.

Detailed analysis of the alert files

We will now analyze in more detail, the different alerts by alert type, in the same order as table 3.1.

UDP SRC and DST outside network

As mentioned before, this is UDP traffic with source and destination addresses considered outside your network(s) by the Snort IDS. Snort uses a variable HOME_NET in its configuration file which tells him which networks he can consider as local. Normally, all

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

other networks are considered outside. It is good security practice to keep real external traffic out of your networks.

Tables 5 through 8 give you information concerni ng the traffic that caused these alerts.

Table 3.5: Distribution of "UDP DST and SRC outside network" alerts by destination port (all)

destination port	# alerts	percent	service	service description
9875	327798	75.03		
5779	47153	10.79		
9880	32372	7.41		
1718	12036	2.75	h323gatedisc	
137	7581	1.74	netbios-ns	NETBIOS Name Service
67	3868	0.89	bootps	Bootstrap Protocol Server
123	3703	0.85	ntp	Network Time Protocol
138	1325	0.30	netbios-dgm	NETBIOS Datagram Service
53	660	0.15	domain	Domain Name Server
30011	162	0.04		
38293	93	0.02		
39213	67	0.02		
9004	33	0.01		Asherons Call
9000	15	< 0.01	cslistener	CSlistener
514	11	< 0.01	syslog	
0	2	< 0.01	(reserved)	
4160	2	< 0.01	jini-discovery	Jini Discovery
162	1	< 0.01	snmptrap	SNMPTRAP

Remark: in the service column, you can find the official service name, as given by the Internet Assigned Numbers Authority (IANA). The service description column gives a description of this official service or mentions services, not found on the IANA site (www.iana.org). If both columns are emp ty, the service is unassigned (not registered) and unknown.

Table 3.6: Distribution of "UDP DST and SRC outside network" alerts by destination IP address (top 10)

dest. IP address	# alerts	percent	host or network information
224.2.127.254	360172	82.44	SAP.MCAST.NET
233.28.65.197	20713	4.74	<< NET-MCAST-NET >>
233.28.65.255	17397	3.98	<< NET-MCAST-NET >>
224.0.1.41	12036	2.75	GATEKEEPER.MCAST.NET
233.40.70.199	5300	1.21	<< NET-MCAST-NET >>
10.255.255.255	3867	0.89	[[RESERVED-6]]
224.0.1.1	3703	0.85	NTP.MCAST.NET
169.254.255.255	2950	0.68	<< NETBLK-LINKLOCAL -- For use with Link Local Networks >>
233.28.65.223	2175	0.50	<< NET-MCAST-NET >>
192.168.0.255	1306	0.30	[[IANA-CBLK-RESERVED]]

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Table 3.7: "UDP DST and SRC outside network" connections, real external addresses (top 10)

connection			# alerts	percent
155.101.21.38	→	224.2.127.254:9875	81304	18.61
171.69.248.71	→	224.2.127.254:9875	29058	6.65
140.142.19.72	→	224.2.127.254:9880	28117	6.44
206.190.54.67	→	233.28.65.197:5779	20713	4.74
129.116.65.3	→	224.2.127.254:9875	17549	4.02
63.250.208.169	→	233.28.65.255:5779	17397	3.98
128.223.83.33	→	224.2.127.254:9875	17270	3.95
152.1.1.79	→	224.2.127.254:9875	16121	3.69
130.235.133.92	→	224.2.127.254:9875	15824	3.62
130.240.64.20	→	224.2.127.254:9875	15736	3.60

Table 3.8: "UDP DST and SRC outside network" connections, private addresses (top 15)

connection			# alerts	percent
10.0.0.1	→	10.255.255.255:67	3867	0.89
192.168.0.2	→	192.168.0.255:137	1028	0.24
172.16.3.213	→	172.16.1.103:137	534	0.12
192.168.0.2	→	192.168.0.255:138	277	0.06
10.3.41.11	→	10.1.11.101:137	217	0.05
10.1.202.19	→	10.2.0.11:137	82	0.02
10.10.10.1	→	10.17.220.10:138	9	0.00
10.10.10.1	→	10.74.97.7:138	8	0.00
10.10.10.1	→	10.17.220.11:138	8	0.00
10.10.10.1	→	10.17.220.15:138	8	0.00
10.10.10.1	→	10.17.220.16:138	8	0.00
63.248.202.90	→	172.16.192.13:53	7	0.00
10.10.10.1	→	10.17.220.14:138	6	0.00
10.10.10.1	→	10.74.96.11:138	6	0.00
10.10.10.1	→	10.74.96.10:138	6	0.00

What can we conclude out of all this?

To begin with, there is really a lot of UDP traffic to 224.2.127.254 (82%), a multicast address used by SAP, on either port 9875 or 9880. Both ports are unregistered by the IANA. By analyzing all the data, we found out that only destination address 224.2.127.254 was using these two ports. There was traffic from 163 different sources, so most likely those port numbers are really used by SAP and were just not handed over to the IANA. We mention the number of hosts using this destination address because it is very unlikely that all those 163 servers were compromised and are using the multicast destination address 224.2.127.254 to connect to other machines, looking for trojans, listening on port 9875 or 9880 (there is a known trojan, Portal of Doom, which uses port 9875, but only for TCP traffic, not UDP).

If there are servers or clients in your company which use this kind of SAP (UDP) multicast traffic, you might consider putting '224.2.127.254' in the HOME_NET variable. It will considerably reduce the amount of false positives for "*UDP DST and SRC outside network*" alerts…

Almost 11% of the traffic is using port 5779 as destination. When analyzing all the data, we see that they all are going to multicast addresses. This is an unregistered port, but all the IP

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

addresses of the transmitting hosts are part of the Yahoo! Broadcast Services, Inc. networks, so most likely this is some kind of Yahoo Messenger traffic.

When analyzing the traffic concerning private addresses, we see a lot of bootps traffic from 10.0.0.1, which seems logical (using 10.0.0.1 as a bootstrap server) and thus normal. NETBIOS (ports 137-139) is known to be noisy, so that traffic is also quite normal.

Maybe the only bizarre thing is the domain (port 53) traffic to 172.16.192.13 from 63.248.202.90. When seeing the actual data, we see that the source port is 137, which indicates that it is a MS Windows system which is probably looking up a host which isn't part of his local (NT) domain. Probably 172.16.192.13 is the internal IP address of one of your DNS servers and the Snort IDS saw the packets after NAT (Network Address Translation).

We see also broadcast traffic from 169.254.0.0/16 addresses. This is only a little piece of this traffic (in total there were 2950 comparable packets seen):

```
02/20-16:35:00.645164 169.254.238.176:  137 -> 169.254.255.255:137
02/20-16:35:29.364499 169.254.238.176:137  -> 169.254.255.255:137
02/20-16:35:33.119821 169.254.238.176:137  -> 169.254.255.255:137
02/20-16:37:22.842139 169.254.238.176:137  -> 169.254.255.255:137
02/20-16:37:23.593285 169.254.238. 176:137 -> 169.254.255.255:137
02/20-16:37:33.878043 169.254.238.176:138  -> 169.254.255.255:138
02/20-16:39:06.077218 169.254.252.132:137  -> 169.254.255.255:137
02/20-16:39:54.604950 169.254.221.37:137  -> 169.254.255.255:137
02/20-16:40:45.265420 169.254.2 52.132:137 -> 169.254.255.255:137
```

We want to remark here that everywhere in this report, the snort alert types ("[**] UDP SRC and DST outside network [**]" for the extract above) were removed from the alert output.

This network (169.254.0.0/16) is reserved by IANA for Link Local Networks. This network is used for IP autoconfiguration, which uses broadcasts to autoconfigure IP addresses in the absence of a DHCP server. It might be that the local net where your Snort IDS is located has DHCP or congestion problems, and WIN98 (and some WIN95 and Mac) machines are negotiating addresses.

Recommendations:

- You might consider adding the private IP addresses (10.0.0.0 – 10.255.255.255, 172.16.0.0 – 172.31.255.255 and 192.168.0.0 – 192.168.255.255) to the HOME_NET variable of your Snort IDS configuration. Snort will generate less false positives for this kind of alert.
- Check whether you are using SAP. If you are, consider adding 224.2.127.254 to the HOME_NET variable. Snort will generate much lesser false positives.
- Consider buying a firewall —or if you have one, using a stricter policy. Besides the traffic explained above, we also saw traffic really going from an external address to another external address. Snort saw this, so it passed your network. This traffic should not be there (unless you are an ISP or route traffic for a living).
- You might have a problem with your DHCP server or congestion problems in the network were the Snort IDS is located.
- Most probably, some routers are wrongly configured. The most outward router(s) should only send packets with an internal destination address into the internal network.

Portscan related alerts

We will consider these in the next part when analyzing the scan files.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Watchlist 000220 IL -ISDNNET-990517

All this traffic originates from the IL-ISDNNET-990517 network, being 212.179.0.0/16. This network is owned by ISDN Net Ltd., an Israeli ISP, and is known to show suspicious activities.

Table 3.9 gives an overview of which services the people of this Israeli network are using from your network.

Table 3.9: Distribution of "Watchlist 000220 IL -ISDNNET-990517" alerts by destination port (top 10)

destination port	# alerts	percent	service	service description
6688	7043	46.89		
6699	4174	27.79		Napster
4718	1451	9.66		
6346	549	3.65	gnutella-svc	
4074	464	3.09		
41003	436	2.90		
4222	207	1.38	vrml-multi-use	VRML Multi User System
2610	187	1.24	versa-tek	VersaTek
4971	154	1.03		
2609	134	0.89	system-monitor	System Monitor

The mostly used service uses an unknown port (6688) and we don't have packet dumps to get deeper into this, so it remains unclear what exactly is going on. Several machines of your network are contacted (most importantly: MY.NET.213.250 and MY.NET.209.114) for this service.

Napster (port 6699) is a well-known application, as is Gnutella (port 6346). There are however security risks involved using these programs because of their nature: they share files (multimedia or other) between your machine and the community —which is nice from a philosophical perspective, but not from a security viewpoint. In addition to possible security holes in the software, which —in the worst case —might give complete access to your system, it is equally thinkable that a user (one of your employees/colleagues) might drop (critical) business related documents in the shared directory, without intent.

Recommendations:

- Find out what service is running on port 6688 on MY.NET.213.250 and MY.NET.209.114 (and others; for a complete list, contact us).
- Consider blocking all Napster, Gnutella and related traffic on your firewall(s). This traffic shouldn't be allowed because of their inherent security risk (file sharing).

SYN-FIN scan!

A SYN-FIN scan is a form of scanning hosts (networks) by sending a packet with the SYN and FIN flag set. Depending on the answer you get, you know whether or not a specific service is listening on the machine you scanned. The advantage of setting both the SYN and FIN flag (the RFC states that only the SYN flag is set to initiate a connection) is that several perimeter devices (firewalls, routers, etc…) let them pass. It is also called a stealth scan.

Table 3.10 gives you all the hosts who performed a SYN-FIN scan. Table 3.11 shows in what services they were interested.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Table 3.10: Distribution of "SYN -FIN scan!" alerts by source IP address (all)

source IP address	# alerts	percent	host or network information
130.234.184.112	9336	80.43	termos.keltti.jyu.fi
128.61.136.233	1158	9.98	tann6233.mse.gatech.edu
211.248.112.67	1108	9.55	<< KAWON-M -- Kawon Middle School, Seoul >>
4.35.4.244	1	0.01	evrtwa1-ar4-004-244.elnk.dsl.gtei.net
24.50.25.5	1	0.01	fl-wellu1-c6-5.pbc.adelphia.net
63.252.15.242	1	0.01	A010-0496.ABDN.splitrock.net
66.25.174.123	1	0.01	cs6625174-123.austin.rr.com
128.206.176.25	1	0.01	mu-176025.dhcp.missouri.edu
209.255.180.130	1	0.01	A010-0384.LAUR.splitrock.net

Table 3.11: Distribution of "SYN -FIN scan!" alerts by destination port (all)

destination port	# alerts	percent	service	service description
21	10494	90.40	ftp	File Transfer [Control]
53	1108	9.55	domain	Domain
412	1	0.01	synoptics-trap	Trap Convention Port
6346	1	0.01	gnutella-svc	
443	1	0.01	https	http protocol over TLS/SSL
1415	1	0.01	dbstar	DBStar
259	1	0.01	esro-gen	Efficient Short Remote Operations
1700	1	0.01	mps-raft	

Let's take a look at host 130.234.184.112:

```
02/25-04:50:13.630822 130.234.184.112:21  -> MY.NET.1.17:21
02/25-04:50:13.690765 130.234.184.112:21  -> MY.NET.1.20:21
02/25-04:50:13.850140 130.234.184.112:21  -> MY.NET.1.28:21
02/25-04:50:14.030527 130.234.184.112:21  -> MY.NET.1.37:21
02/25-04:50:14.111125 130.234.184.112:21  -> MY.NET.1.41:21
02/25-04:50:14.151043 130.234.184.112:21  -> MY.NET.1.43:21
02/25-04:50:14.210940 130.234.184.112:21  -> MY.NET.1.46:21
…               (9322 entries omitted)
02/25-05:24:27.923716 130.234.184.112:21  -> MY.NET.254.221:21
02/25-05:24:28.143017 130.234.184.112:21  -> MY.NET.254.232:21
02/25-05:24:28.302113 130.234.184.112:21  -> MY.NET.254.240:21
02/25-05:24:28.324129 130.234.184.112:21  -> MY.NET.254.241:21
02/25-05:24:28.342488 130.234.184.112:21  -> MY.NET.254.242:21
02/25-05:24:28.462988 130.234.184.112:21  -> MY.NET.254.248:21
02/25-05:24:28.482939 130.234.184.112:21  -> MY.NET.254.249:21
```

This is quite serious. This guy (girl) isn't doing half work. Not only is (s)he scanning all your hosts for FTP servers (which are known to have vulnerabilities), (s)he has also done a reconnaissance sweep somewhere in the past, because (s)he knows the IP addresses of your systems ((s)he doesn't try the complete B-class, but uses specific IP addresses only, totaling 9336 hosts). Obviously, some kind of script is used (9336 hosts scanned in less than 35 minutes). The source is a system at the University of Jyväskylä in Finland.

Let's take a look at the most interesting one:

```
02/06-16:58:47.639057 211.248.112.67:53  -> MY.NET.1.29:53
02/06-16:58:48.039145 211.248.112.67:53  -> MY.NET.1.130:53
02/06-16:58:48.118237 211.248.112.67:53  -> MY.NET.1.134:53
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
02/06-16:58:48.246195 211.248.112.67:53  -> MY.NET.1.67:53
02/06-16:58:48.384843 211.248.112.67:53  -> MY.NET.1.74:53
02/06-16:58:48.439284 211.248.112.67:53  -> MY.NET.1.150:53
02/06-16:58:48.520306 211.248.112.67:53  -> MY.NET.1.154:53
…               (1094 entries omitted)
02/06-17:20:15.906113 211.248.112.67:53  -> MY.NET.254.2:53
02/06-17:20:16.266869 211.248.112.67:53  -> MY.NET.254.20:53
02/06-17:20:16.285449 211.248.1 12.67:53 -> MY.NET.254.21:53
02/06-17:20:16.485735 211.248.112.67:53  -> MY.NET.254.31:53
02/06-17:20:18.266746 211.248.112.67:53  -> MY.NET.254.120:53
02/06-17:20:20.459291 211.248.112.67:53  -> MY.NET.254.172:53
02/06-17:20:21.100779 211.248.112.67:53  -> MY.NET.254.215:53
```

We consider this one even more dangerous than the two above, because (s)he is searching for an exploitable dns server (some versions of BIND have well known vulnerabilities). If (s)he could be able to compromise your DNS servers —considering the worst case scenario —you could be out of business for a while: imagine all your customers, unable to reach your transaction site… Not a pretty picture. By the way, the hacker is probably 16 years old or so: IP address 211.248.112.67 belongs to the Kaw on Middle School, somewhere in Seoul. Unless the source address was spoofed, but this isn't that logical for a scan (because you need the response coming back to you), but not impossible (we'll explain later)!

There was also a dump file for this SYN -FIN scan, so we were able to investigate this event even further. We show the first three packets as given by the Snort IDS:

```
02/06-16:58:53.282171 211.248.112.67:53  -> MY.NET.1.29:53
TCP TTL:20 TOS:0x0 ID:39426
**SF**** Seq: 0x56572831   Ack: 0x62C5EC3E   Win:   0x404
00 00 00 00 00 00                                ......

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/06-16:58:53.682381 211.248.112.67:53  -> MY.NET.1.130:53
TCP TTL:20 TOS:0x0 ID:39426
**SF**** Seq: 0x31CA253C   Ack: 0x4C54  4802   Win: 0x404
00 00 00 00 00 00                                ......

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
02/06-16:58:53.761434 211.248.112.67:53  -> MY.NET.1.134:53
TCP TTL:20 TOS:0x0 ID:39426
**SF**** Seq: 0x31CA253C    Ack: 0x4C544802   Win: 0x404
00 00 00 00 00 00                                ......
```

The first thing (except for both SYN and FIN flags set) that hits the eye is that the IP Identification number is always the same (39426). This indicates packet crafting, because normally this number should increment. The SYN and FIN flag set, the window size 0x404, the same source and destination ports (53) and IP Identification number 39426 indicate that some scanning tool is used. There are at least two scanning tools that generate this kind of traffic:

- *synscan*, created by *psychoid*; more specifically version 1.6 of this tool shows this behavior
- *Idlescan*, created by *liquidK*

In the latter case, the source address was spoofed (we told you it was possible). It relies on the predictability of the IP ID number generator (most operating systems increment it by one, every time a packet is sent). It works as follows:

- Idlescan (*HACKER*) sends an ICMP echo -request to some host it will use (*ABUSED*). It receives an ICMP echo -reply and remembers its IP ID.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

- *HACKER* then sends a SYN packet to the *TARGET*, using the IP address of ABUSED as source (spoofed packet).
- *TARGET* will send either a SYN-ACK or RESET back to *ABUSED* (he thinks *ABUSED* wants to initiate a connection).
- *ABUSED* will send a RESET back to *TARGET* if he received a SYN-ACK (he didn't initiate a connection!) or nothing in case he received a RESET.
- *HACKER* sends again an ICMP echo-request to *ABUSED* and checks the IP ID. If it is exactly one more than the remembered one, *TARGET* has sent a RESET and the port isn't open. If the difference is more than one, it might indicate that *TARGET* has send a SYN-ACK and that *TARGET* is listening on the chosen port.

For this to work however, *ABUSED* needs to be a very quiet host, needs to respond to ICMP echo-request and must have a predictable IP ID generator (Idlescan expects an increment by one, but the principle remains valid for every predictable IP ID generator). We tried sending echo-requests to 211.248.112.67, but it didn't respond. This doesn't mean that Idlescan hasn't been used. It merely means that 211.248.112.67 isn't responding to echo-requests *now* (but might have been in the past, e.g. during the scan).

Finally, host 128.61.136.233 (the second entry in table 3.10) used exactly the same technique as described above, but is looking for FTP servers. When we look at the actual packet dumps, we see that there are even more hosts scanned (2967) than that there are entries in the alert file (1158). Maybe the analyzer couldn't keep up or some packets were lost. The scan took place between 03/06-16:07:53 and 03/06-16:29:23.

Recommendations:

- You might consider blocking (manually) (any traffic from) the IP address when seeing this kind of traffic (Snort can write alerts to a unix socket, on which some kind of script can be listening, sending you a mail when it receives this kind of alert). It might just be a script kiddy fooling around, but it also might be somebody who means business.

  However, this might not be as easy as it looks, because IP addresses can be changed very quickly (just use another PC/workstation or reconnect with your modem). You might be forced to block a complete network, which is —socially speaking—not that obvious if we are talking about a complete A- or B-class network.

Possible RAMEN server activity

This alert is triggered when traffic from/to port 27374 is seen. This port is used by many trojans and worms, including Bad Blood, Ramen, Seeker, SubSeven, SubSeven 2.1 Gold, Subseven 2.1.4 DefCon 8, SubSeven Muie and Ttfloader. Ramen is a worm for RedHat Linux, the others target MicroSoft Windows systems.

Table 3.12 gives you an idea of which hosts are involved in possible ramen/subseven traffic.

Table 3.12: Distribution of "Possible RAMEN server activity" alerts by source IP address (top 10)

source IP address	# alerts	percent	host or network information
24.67.186.244	2438	24.59	<< NETBLK-FIBERLINK-CABLE -- Shaw Fiberlink ltd. >>
24.48.226.183	1819	18.35	pa-southhills2a-695.pit.adelphia.net
128.138.2.112	728	7.34	aden2-112-dhcp.resnet.Colorado.EDU
MY.NET.201.146	553	5.58	

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

MY.NET.253.12	530	5.35	
MY.NET.97.154	330	3.33	
MY.NET.60.11	326	3.29	
148.129.143.2	210	2.12	inet-gw.census.gov
MY.NET.225.66	60	0.61	
MY.NET.217.202	30	0.30	

Host 24.67.186.244 did an extensive scan for trojans li stening on port 27374 between 02/23 -22:57:57 and 02/23 -23:17:01. In those 20 minutes (s)he scanned 2438 hosts for these trojans. Host 24.48.226.183 did also an extensive scan between 02/11 -23:03:19 and 02/11 -23:20:55. On those 17 minutes, 1819 hosts were s canned.

Host 128.138.2.112 however is not an aggressor, but a victim:

```
02/23-03:15:50.748824 128.138.2.112:27374  -> MY.NET.201.146:4781
…
02/23-03:15:51.768541 128.138.2.112:27374  -> MY.NET.201.146:4781
02/23-03:15:51.881273 128.138.2.112:27374  -> MY.NET. 201.146:4781
02/23-03:15:51.893841 MY.NET.201.146:4781  -> 128.138.2.112:27374
02/23-03:15:51.893891 MY.NET.201.146:4781  -> 128.138.2.112:27374
02/23-03:15:51.893936 MY.NET.201.146:4781  -> 128.138.2.112:27374
02/23-03:15:52.068139 128.138.2.112:27374  -> MY.NET.201.146:4781
02/23-03:15:52.192246 128.138.2.112:27374  -> MY.NET.201.146:4781
02/23-03:15:52.193248 128.138.2.112:27374  -> MY.NET.201.146:4781
02/23-03:15:52.203371 MY.NET.201.146:4781  -> 128.138.2.112:27374
02/23-03:15:52.203416 MY.NET.201.146:4781  -> 128.138.2.112:27374
02/23-03:15:52.253343 128.138.2.112:27374  -> MY.NET.201.146:4781
02/23-03:15:52.313648 128.138.2.112:27374  -> MY.NET.201.146:4781
02/23-03:15:52.375203 128.138.2.112:27374  -> MY.NET.201.146:4781
02/23-03:15:52.496483 128.138.2.112:2737 4 -> MY.NET.201.146:4781
02/23-03:15:52.507778 MY.NET.201.146:4781  -> 128.138.2.112:27374
…
02/23-03:17:51.381172 MY.NET.201.146:4781  -> 128.138.2.112:27374
```

You can see packets going in both directions between 128.138.2.112 and MY.NET.201.146. It is howev er the first host who is using port 27374, so apparently someone at MY.NET.201.146 was (ab)using a trojan, installed at 128.138.2.112.

We see that the hosts given in table 3.12 are not necessarily aggressors; they could be victims as well, because the ale rt is triggered if the destination *or* the source port is equal to 27374. To find out which of your hosts might be compromised, we look at table 3.13.

Table 3.13: MY.NET hosts, possibly compromised by Ramen/SubSeven trojan (top 5)

source IP address & port	# alerts	percent
MY.NET.225.66:27374	60	0.61
MY.NET.217.202:27374	30	0.30
MY.NET.223.42:27374	10	0.10
MY.NET.227.94:27374	9	0.09
MY.NET.60.8:27374	7	0.07

When checking out the first one, we saw the following traffic:

```
02/04-07:45:38.425550 MY.NET.225.6 6:27374 -> 24.48.121.105:4840
02/04-07:45:38.935861 MY.NET.225.66:27374  -> 24.48.121.105:4840
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
02/04-08:14:37.174375 MY.NET.225.66:27374  -> 24.48.121.105:1952
02/04-08:14:37.668263 24.48.121.105:1952  -> MY.NET.225.66:27374
02/04-08:14:37.668410 MY.NET.225.6 6:27374 -> 24.48.121.105:1952
02/04-08:49:53.976126 24.48.121.105:2878  -> MY.NET.225.66:27374
02/04-08:49:53.976330 MY.NET.225.66:27374  -> 24.48.121.105:2878
02/04-08:49:54.514896 24.48.121.105:2878  -> MY.NET.225.66:27374
02/04-08:49:54.514942 MY.NET.225.6 6:27374 -> 24.48.121.105:2878
02/04-08:49:55.022287 24.48.121.105:2878  -> MY.NET.225.66:27374
02/04-09:04:46.762121 24.48.121.105:3064  -> MY.NET.225.66:27374
02/04-09:04:47.210992 24.48.121.105:3064  -> MY.NET.225.66:27374
02/04-09:04:47.211087 MY.NET.225.6 6:27374 -> 24.48.121.105:3064
02/04-09:04:48.225217 24.48.121.105:3064  -> MY.NET.225.66:27374
02/04-09:04:48.225270 MY.NET.225.66:27374  -> 24.48.121.105:3064
02/04-09:11:45.429933 24.48.121.105:3293  -> MY.NET.225.66:27374
02/04-09:39:22.444555 24.48.121.10 5:3966 -> MY.NET.225.66:27374
02/04-09:39:22.444653 MY.NET.225.66:27374  -> 24.48.121.105:3966
02/04-09:41:17.102241 MY.NET.225.66:27374  -> 24.48.121.105:4017
02/04-09:41:17.607801 24.48.121.105:4017  -> MY.NET.225.66:27374
02/04-09:41:17.607846 MY.NET.225.6 6:27374 -> 24.48.121.105:4017
02/04-09:47:07.795336 24.48.121.105:4103  -> MY.NET.225.66:27374
02/04-09:47:07.795436 MY.NET.225.66:27374  -> 24.48.121.105:4103
02/04-09:47:08.297056 MY.NET.225.66:27374  -> 24.48.121.105:4103
02/04-09:49:51.072432 MY.NET.225.6 6:27374 -> 24.48.121.105:4154
02/04-09:49:51.636933 24.48.121.105:4154  -> MY.NET.225.66:27374
02/04-10:06:23.823078 24.48.121.105:4419  -> MY.NET.225.66:27374
02/04-10:06:23.823548 MY.NET.225.66:27374  -> 24.48.121.105:4419
```

and also

```
02/04-09:03:01.685988 M Y.NET.225.66:27374  -> 203.106.99.237:1189
02/04-09:03:03.827397 203.106.99.237:1189  -> MY.NET.225.66:27374
02/04-09:04:47.708466 203.106.99.237:1212  -> MY.NET.225.66:27374
02/04-09:08:11.541729 203.106.99.237:1309  -> MY.NET.225.66:27374
02/04-09:08:11.5418 67 MY.NET.225.66:27374  -> 203.106.99.237:1309
02/04-09:08:15.004230 MY.NET.225.66:27374  -> 203.106.99.237:1309
02/04-09:11:07.039604 203.106.99.237:1416  -> MY.NET.225.66:27374
02/04-09:38:37.242414 MY.NET.225.66:27374  -> 203.106.99.237:1799
02/04-09:38:40. 331294 203.106.99.237:1799  -> MY.NET.225.66:27374
02/04-09:38:40.331381 MY.NET.225.66:27374  -> 203.106.99.237:1799
02/04-09:49:47.493556 203.106.99.237:1928  -> MY.NET.225.66:27374
02/04-09:49:47.493657 MY.NET.225.66:27374  -> 203.106.99.237:1928
02/04-09:49:50.519066 MY.NET.225.66:27374  -> 203.106.99.237:1928
02/04-10:06:26.261572 203.106.99.237:2106  -> MY.NET.225.66:27374
02/04-10:06:26.261657 MY.NET.225.66:27374  -> 203.106.99.237:2106
02/04-10:06:28.387394 MY.NET.225.66:27374  -> 203.106.99.237:2106
02/04-10:10:38.539370 203.106.99.237:2178  -> MY.NET.225.66:27374
02/04-10:10:38.539416 MY.NET.225.66:27374  -> 203.106.99.237:2178
```

There were also other addresses than 24.48.121.105 and 203.106.99.237 communicating with MY.NET.225.66, but these two were the most prominent. Most likely the host MY.NET.225.66 is compromised with some trojan listening on port 27374.

For the same reason, we think MY.NET.217.202 is compromised:

```
02/04-09:09:37.697409 203.79.69.182:4189  -> MY.NET.217.202:27374
02/04-09:09:37.697454 MY. NET.217.202:27374  -> 203.79.69.182:4189
02/04-09:09:40.927410 MY.NET.217.202:27374  -> 203.79.69.182:4189
02/04-09:27:31.194526 203.79.69.182:4588  -> MY.NET.217.202:27374
02/04-09:27:31.197085 MY.NET.217.202:27374  -> 203.79.69.182:4588
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
02/04-07:25:31.59494 3 MY.NET.217.202:27374  -> 212.71.36.71:2478
02/04-08:08:40.586717 MY.NET.217.202:27374   -> 212.71.36.71:3441
02/04-08:08:49.727481 212.71.36.71:3441   -> MY.NET.217.202:27374
02/04-08:08:50.826597 212.71.36.71:3441   -> MY.NET.217.202:27374
02/04-08:21:22.53730 3 212.71.36.71:3922  -> MY.NET.217.202:27374
02/04-08:21:22.538955 MY.NET.217.202:27374   -> 212.71.36.71:3922

02/04-09:27:17.483578 64.228.227.4:1932   -> MY.NET.217.202:27374
02/04-09:27:18.171023 MY.NET.217.202:27374   -> 64.228.227.4:1932
02/04-09:27:18.8710 54 64.228.227.4:1932  -> MY.NET.217.202:27374
02/04-09:27:18.872391 MY.NET.217.202:27374   -> 64.228.227.4:1932
02/04-09:27:19.579350 64.228.227.4:1932   -> MY.NET.217.202:27374
```

MY.NET. 223.42 and MY.NET.227.94 might also be compromised. We don't think MY.NET.60.8 is compromised.

Recommendations:

- Investigate MY.NET.225.66, MY.NET.217.202, MY.NET.223.42 and MY.NET.227.94. They might be compromised with the Ramen or SubSeven trojan (or other trojans using port 27374). It is also possible that there are more compromised hosts in your network. The easiest way to remove the SubSeven trojan is by using an antivirus program (McAfee, Norton, etc…). You can find more info about the ramen worm on http://whitehats.com/print/library/worms/ramen/index.html.

Watchlist 0002 22 NET-NCFC

All this traffic originates from the NET -NCFC network, being 159.226.0.0/16. This network is owned by the Computer Network Center Chinese Academy of Sciences and is known to show suspicious activities.

Tables 3.14 through 3.16 give you some in sight in the traffic.

Table 3.14: Distribution of "Watchlist 000222 NET -NCFC" alerts by source IP address (top 5)

source IP address	# alerts	percent	host or network information
159.226.81.1	5362	93.61	pa.ivpp.ac.cn
159.226.45.204	170	2.97	dos204.iphy.ac .cn
159.226.45.108	111	1.94	dos108.iphy.ac.cn
159.226.39.4	35	0.61	mail.ict.ac.cn
159.226.210.6	10	0.17	wwwserver.camc.com.cn

Table 3.15: Distribution of "Watchlist 000222 NET -NCFC" alerts by destination port (top 3)

destination port	# alerts	percent	service	service description
25	4744	82.82	smtp	Simple Mail Transfer
23	281	4.91	telnet	Telnet
113	7	0.12	ident	
			auth	Authentication Service

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Table 3.16: "Watchlist 000222 NET -NCFC" connections (top 5)

connection			# alerts	percent
159.226.81.1	→	MY.NET.6.47:25	4658	81.32
159.226.45.204	→	MY.NET.6.7:23	170	2.97
159.226.45.108	→	MY.NET.60.11:23	80	1.40
159.226.39.4	→	MY.NET.100.230:25	33	0.58
159.226.45.108	→	MY.NET.6.7:23	31	0.54

All the traffic seems legitimate traffic. We tried to connecting to 159.226.81.1, 159.226.39.4 and 159.226.210.6 on port 25 and they all responded, so they are mail servers and it is normal that they communicate with your mail servers (We couldn't check whether MY.NET.6.47 and MY.NET.100.230 are mail servers, but we guess so because of the connections to port 25).

The telnet sessions also seem completely normal. If you are really paranoid, you could check if 159.226.45.204 and 159.226.45.108 should have telnet access to MY.NET.6.7 and MY.NET.100.230 (it is always possible that someone used a brute-force attack against those machines in the past and has gained unauthorized access).

Recommendations:

- You might consider using SSH instead of telnet to access your machines. Stronger authentication is possible and traffic is encrypted.
- If you are really paranoid, you could check if 159.226.45.204 and 159.226.45.108 should have (telnet) access to MY.NET.6.7 and MY.NET.100.230.

NMAP TCP ping!

NMAP TCP ping is scan to see which hosts are up, by sending TCP ACK packets, by default to port 80 (http). A machine that is up sends a RESET packet back. You can invoke this behavior for nmap by invoking it as follows: `nmap -sP -PT`. If you want to use a different port, just add it after the '-PT', e.g. `nmap -sP -PT666`.

Table 3.17 shows the most interested hosts.

Table 3.17: Distribution of "NMAP TCP ping!" alerts by source IP address (top 3)

source IP address	# alerts	percent	host or network information
MY.NET.70.38	4786	99.34	
192.102.197.234	12	0.25	geo197a.cps.intel.com
63.119.91.2	5	0.10	<< NETBLK-UUNET63 -- UUNET Technologies, Inc. >>

192.102.197.234 and 63.119.91.2 have done some pings, but there are days between two attempts and only one or two destination hosts are involved.

MY.NET.70.38 is a different story. Between 02/20-00:00:46 and 02/23-13:56:55, 3814 hosts are scanned. Remark that there is a gap for February 21 and also a gap for the targeted hosts:

```
02/20-23:50:29.669835 MY.NET.70.38:36339  -> MY.NET.137.147:34181
02/20-23:51:15.038619 MY.NET.70.38:36339  -> MY.NET.137.150: 37213
02/20-23:51:26.005949 MY.NET.70.38:36339  -> MY.NET.137.150:37989
02/20-23:51:57.228618 MY.NET.70.38:36339  -> MY.NET.137.152:40883
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
02/22-00:52:29.643684 MY.NET.70.38:36339  -> MY.NET.212.255:41046
02/22-01:46:37.014547 MY.NET.70.38:36339  -> MY.NET.216. 173:33511
02/22-01:46:59.351011 MY.NET.70.38:36339  -> MY.NET.216.175:39515
02/22-01:47:17.162828 MY.NET.70.38:36339  -> MY.NET.216.176:30144
```

Probably the disk of the snort system was full and we can assume that your complete network was scanned. Remark als o that the destination ports are different each time, instead of using port 80 (which is the default behavior) or some other (but always the same) port.

Recommendations:

- You might consider checking out why MY.NET.70.38 was doing a complete NMAP TCP ping s weep against your complete network between 02/20 -00:00:46 and 02/23 - 13:56:55. Maybe it was just some System Operator checking which machines were still up and running, but better be safe than sorry…

TCP SRC and DST outside network

This alert is comparabl e to the "UDP SRC and DST outside network" alert, but in stead of UDP traffic, here we have TCP traffic. Again, for most companies, you shouldn't see this kind of traffic.

Let's take a look at tables 3.18 and 3.19.

Table 3.18: Distribution of "TCP SRC an d DST outside network" alerts by source IP address (top 5)

source IP address	# alerts	percent	host or network information
127.0.0.1	1533	89.02	[[LOOPBACK]]
169.254.101.152	42	2.44	<< NETBLK-LINKLOCAL -- For use with Link Local Networks >>
192.168.1.51	18	1.05	[[IANA-CBLK-RESERVED]]
10.3.41.11	17	0.99	[[RESERVED-6]]
0.0.0.0	12	0.70	

Table 3.19: Distribution of "TCP SRC and DST outside network" alerts by destination IP address (top 5)

source IP address	# alerts	percent	host or network information
1.1.1.1	1533	89.02	<< RESERVED-9 >>
192.168.0.92	11	0.64	[[IANA-CBLK-RESERVED]]
4.0.0.3	7	0.41	l0.washdc3-cmb1.bbnplanet.net
64.4.13.210	7	0.41	msgr-ns33.msgr.hotmail.com
3.0.0.2	6	0.35	<< NET-GE-INTERNET -- General Electric Company >>

Some remarkable t hings are going on…

To begin with, 127.0.0.1, the loopback address shouldn't be seen on the net. It is only used for communication between local processes on one machine (e.g. X -Windows needs it). Let's take a look at the traffic, seen by Snort IDS:

```
02/20-03:13:22.022820 127.0.0.1:110  -> 1.1.1.1:18539
02/20-03:13:22.024137 127.0.0.1:115  -> 1.1.1.1:18544
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
02/20-03:13:22.027876 127.0.0.1:128  -> 1.1.1.1:18557
02/20-03:13:22.027923 127.0.0.1:129  -> 1.1.1.1:18558
02/20-03:13:22.029320 127.0.0.1:131  -> 1.1.1.1:1 8560
02/20-03:13:22.061151 127.0.0.1:153  -> 1.1.1.1:18582
02/20-03:13:22.062039 127.0.0.1:155  -> 1.1.1.1:18584
…         (1519 entries omitted)
02/24-18:20:00.486856 127.0.0.1:16969  -> 1.1.1.1:17486
02/24-18:20:00.486900 127.0.0.1:17007  -> 1.1.1.1:17487
02/24-18:20:00.487029 127.0.0.1:17300  -> 1.1.1.1:17488
02/24-18:20:00.487132 127.0.0.1:18000  -> 1.1.1.1:17489
02/24-18:20:00.489824 127.0.0.1:22222  -> 1.1.1.1:17498
02/24-18:20:00.494883 127.0.0.1:26274  -> 1.1.1.1:17517
02/24-18:20:00.497811 127.0.0.1:31336  -> 1.1.1.1:17529
```

At first sight, this looks like a scan (a port scan for host 1.1.1.1). Maybe someone spoofed the 127.0.0.1 address, but then again, what's the point of scanning for open ports if you won't receive the results. So perhaps this was some kind of De nial of Service attack since the reply packets would all be sent to the localhost. The speed is impressive (only a few milliseconds between two packets) and the host could get into trouble. But then again, the destination address (1.1.1.1) is reserved by t he IANA and not routable, so the attacker wouldn't exactly be the smartest guy/girl in the world…

What probably is going on is that some machine is acting bizarre. It could be broken hardware…

The second source address we find in table 3.18 is 169.254.10 1.152, which is part of the 169.254.0.0/16 network, reserved by IANA for Link Local Networks. As mentioned before, this network is used for IP autoconfiguration, which uses broadcasts to autoconfigure IP addresses in the absence of a DHCP server. The probl em is that, when looking at the actual traffic, we see that they are not broadcasts, but unicasts, mostly towards the OAL Network (205.188.0.0/16):

```
01/30-01:18:49.213789 169.254.101.152:2715  -> 205.188.48.152:5190
01/30-01:19:37.279909 169.254.101.152:271 5 -> 205.188.48.152:5190
01/30-02:13:08.439529 169.254.101.152:2703  -> 205.188.49.112:5190
01/30-10:12:28.529264 169.254.101.152:1986  -> 205.188.49.176:5190
…              (34 entries omitted)
02/23-10:28:14.121906 169.254.148.188:1029  -> 64.4.13.210:1863
02/23-11:23:37.897572 169.254.101.152:1301  -> 205.188.48.230:5190
02/24-13:14:29.274961 169.254.101.152:3874  -> 205.188.49.25:5190
02/24-22:52:30.702497 169.254.101.152:3524  -> 205.188.49.24:5190
```

Maybe some of your colleagues/employees use AOL's instant messaging so ftware. I'm not sure how reliable that software is, but we've seen strange behavior of other IM software.

We are not concerned about the private addresses seen by the Snort IDS. The use of address 0.0.0.0 is however bizarre:

```
01/30-18:09:50.215686 0.0.0.0 :1543 -> 64.12.24.228:5190
02/04-22:35:02.935504 0.0.0.0:1030  -> 64.12.24.229:5190
02/06-00:25:30.113493 0.0.0.0:2489  -> 205.188.6.217:5190
02/06-01:24:28.317096 0.0.0.0:2489  -> 205.188.6.217:5190
02/20-01:33:01.689835 0.0.0.0:2978  -> 24.9.220.69:1029
02/20-09:42:00.740454 0.0.0.0:3343  -> 24.9.220.69:1034
02/20-09:42:03.485905 0.0.0.0:3343  -> 24.9.220.69:1034
02/23-21:32:11.302188 0.0.0.0:1091  -> 64.12.25.101:8633
02/24-23:42:10.534096 0.0.0.0:3206  -> 65.4.225.188:6688
02/24-23:42:10.534260 0.0.0.0:3205  -> 24.153.20.112:6699
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
02/25-14:44:10.389478 0.0.0.0:1626  -> 24.88.51.246:1615
03/06-01:30:09.044119 0.0.0.0:2652  -> 64.12.24.71:5190
```

64.12.0.0/16 is also part of the OAL network. 24.9.0.0./16 is part of the @Home network. We think these alerts are just glitc hes by some software products. Nevertheless, this traffic shouldn't be there…

Recommendations:

- You might have broken hardware. However, it is very difficult to trace it, because the source address is the loopback address. If you are able to sniff all your (sub)networks, you could point it out, because of the reserved destination address that it is using (1.1.1.1).
- You might consider adding the private IP addresses (10.0.0.0 – 10.255.255.255, 172.16.0.0 – 172.31.255.255 and 192.168.0.0 – 192.168.255.255) to the HOME_NET variable of your Snort IDS configuration. Snort will generate less false positives for this kind of alert.
- Some of your colleagues/employees might be using instant messaging (AOL). This software is probably causing some of the false positives seen for this kind of alert. It might be hard to filter them out, even if you want to. IM software could be a security risk, because of possibly known security vulnerabilities in the software.
- Consider blocking all (real) external TCP traffic on the perim eter firewall(s), if you have one (any).
- Most probably, some routers are wrongly configured. The most outward router(s) should only send packets with an internal destination address into the internal network.

External RPC call

Remote Procedure Call (RPC) provides transparent services for programmers by calling a procedure on a remote machine on your behalf. It makes many things possible (NFS, NIS, Yellow Pages,…), but is also a major source of abuse.

Table 3.20 gives you an overview of the people who wer e using/trying RPC from outside your company to your machines.

Table 3.20: Distribution of "External RPC call" alerts by source IP address (all)

source IP address	# alerts	percent	host or network information
171.65.61.201	1267	83.52	psych-3365-PC.Stanford.EDU
129.105.107.190	245	16.15	dhcp107190.sesp.nwu.edu
209.88.124.3	4	0.26	<< NETBLK-MAURITANIA-TELECOM -- Mauritanian Telecommunication Company >>
199.174.56.66	1	0.07	user-v3qse22.biz.mindspring.com

Host 171.65.61.201 is searching/scanning for hosts listening on port 111 (Portmapper):

```
02/20-19:41:05.730067 171.65.61.201:1464  -> MY.NET.1.15:111
02/20-19:41:05.731385 171.65.61.201:1462  -> MY.NET.1.13:111
02/20-19:41:06.172737 171.65.61.201:1455  -> MY.NET.1.6:111
02/20-19:41:07.758966 171.65.61.201:2214  -> MY.NET.4.0:111
02/20-19:41:07.759014 171.65.61.201:2215  -> MY.NET.4.1:111
…              (1257 entries omitted)
02/20-19:50:27.762190 171.65.61.201:3566  -> MY.NET.253.125:111
02/20-19:50:27.801089 171.65.61.201:3827  -> MY.NET.254.131:111
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
02/20-19:50:27.802801 171 .65.61.201:3837  -> MY.NET.254.141:111
02/20-19:50:27.841864 171.65.61.201:3558   -> MY.NET.253.117:111
02/20-19:50:27.841918 171.65.61.201:3559   -> MY.NET.253.118:111
```

The scan took 9 minutes for 1267 hosts. Host 129.105.107.190 also did a network scan for Portmapper, but not as elaborate (3 minutes for 245 hosts).

SNMP public access

This alert is triggered when the string 'public' is encountered in SNMP traffic. Simple Network Management Protocol (SNMP) is used to monitor and administer network connected devices (printers, computers, routers, etc…). SNMP uses an unencrypted community string (this is actually a password) and many devices use the default community string of 'public' or 'private', which is very dangerous, because someone might then be able to ea sily change the configuration of those devices. Imagine your external router being changed so that all traffic to your webserver is dropped…

Table 3.21 gives you an overview of the people who tried the 'public' string to gain access (legitimate or not) to your machines. Table 3.22 gives you the actual connections.

Table 3.21: Distribution of "SNMP public access" alerts by source IP address (all)

source IP address	# alerts	percent	host or network information
128.46.156.197	1140	98.70	ece156-dhcp-28.ecn.purdue.edu
128.183.38.30	10	0.87	cesdis6.gsfc.nasa.gov
MY.NET.70.42	3	0.26	
MY.NET.111.156	2	0.17	

Table 3.22: "SNMP public access" connections (all)

connection			# alerts	percent
128.46.156.197	→	MY.NET.100.99:161	872	75.50
128.46.156.197	→	MY.NET.100.206 :161	144	12.47
128.46.156.197	→	MY.NET.100.143:161	121	10.48
128.183.38.30	→	MY.NET.154.26:161	10	0.87
MY.NET.70.42	→	MY.NET.50.154:161	3	0.26
MY.NET.111.156	→	MY.NET.50.154:161	2	0.17
128.46.156.197	→	MY.NET.100.160:161	1	0.09
128.46.156.197	→	MY.NET.100. 45:161	1	0.09
128.46.156.197	→	MY.NET.100.205:161	1	0.09

This is no laughing matter. These are not scans, they represent actual connections (or at least attempts): 872 to MY.NET.100.99, 144 to MY.NET.100.206 and 121 to MY.NET.100.143; all of them from 12 8.46.156.197:

```
02/22-11:56:55.782297 128.46.156.197:1191  -> MY.NET.100.99:161
02/22-11:58:09.934277 128.46.156.197:1200  -> MY.NET.100.206:161
02/22-11:59:46.118246 128.46.156.197:1232  -> MY.NET.100.99:161
02/22-12:00:54.380538 128.46.156.197:1238  -> MY.NET .100.206:161
02/22-12:00:55.027408 128.46.156.197:1242  -> MY.NET.100.99:161
02/22-12:01:08.363542 128.46.156.197:1251  -> MY.NET.100.143:161
02/22-12:06:36.137040 128.46.156.197:1330  -> MY.NET.100.99:161
…               (1126 entries omitted)
02/28-06:56:09.874132 128. 46.156.197:2804 -> MY.NET.100.99:161
02/28-06:56:14.337870 128.46.156.197:2805  -> MY.NET.100.99:161
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
02/28-06:56:16.280050 128.46.156.197:2808  -> MY.NET.100.143:161
02/28-06:56:16.327532 128.46.156.197:2809  -> MY.NET.100.143:161
02/28-08:08:42.555437 128.46 .156.197:3843 -> MY.NET.100.206:161
02/28-08:08:47.549512 128.46.156.197:3848  -> MY.NET.100.99:161
02/28-08:08:55.876824 128.46.156.197:3855  -> MY.NET.100.99:161
```

If I was a hacker, I wouldn't bother coming back to the same server when my first attempt wasn't successful. Most likely (s)he gained access by trying the default community string and then started playing with them. Most likely these three hosts are compromised! They should be checked out.

The following traffic is a little bizarre, because of the source port (they are all the same, normally they should increment or be random):

```
02/20-10:33:55.951000 128.183.38.30:1030  -> MY.NET.154.26:161
02/20-14:29:33.326891 128.183.38.30:1030  -> MY.NET.154.26:161
02/20-14:30:03.368514 128.183.38.30:1030  -> MY.NET.154.26:161
02/20-14:32:33.607755 128.183.38.30:1030  -> MY.NET.154.26:161
02/20-14:35:03.889327 128.183.38.30:1030  -> MY.NET.154.26:161
02/20-14:36:34.025095 128.183.38.30:1030  -> MY.NET.154.26:161
02/20-14:51:04.459589 128.183.38.30:1030  -> MY.NET.154.2 6:161
02/20-17:41:04.832151 128.183.38.30:1030  -> MY.NET.154.26:161
02/27-12:12:31.744265 128.183.38.30:1030  -> MY.NET.154.26:161
02/27-16:52:32.386968 128.183.38.30:1030  -> MY.NET.154.26:161
```

This is however not a scan. Notice also that (s)he tries again one week later. Did (s)he really try 10 times the string 'public' and didn't get access? We think that (s)he did get access and something is wrong with the kernel's choosing of source ports. We don't have enough information to be sure, but we don't think that someone who works for NASA (128.183.38.30 resolves to cesdis6.gsfc.nasa.gov) is that silly that (s)he thinks that when something doesn't work the first seven times, maybe it will work the eighth time…

Recommendations:

- Try gaining access to MY.NET.100 .99, MY.NET.100.206, MY.NET.100.143 and MY.NET.154.26 by using the default community string. If you can gain access to them, *consider these hosts compromised* and take appropriate actions. Don't consider them safe if you can't gain access with the default c ommunity string (maybe someone—fiend or foe—changed the community string in the meanwhile). We recommend investigating these hosts thoroughly.

SMB Name Wildcard

This alert is triggered when the string 'SKAA…AA' (containing 30 A's) is encountered in traffic to port 137 (NETBIOS Name Service). Using the program 'nbtstat' a user may gain lots of information about (remote) MS Windows hosts. This is a serious security risk. All incoming NETBIOS traffic (ports 137 -139) should be blocked at the perimeter of your network.

305 external sources try to gain information in this way. There were also 2 internal sources, but that could be legitimate traffic.

Recommendations:

- Consider blocking all incoming (and outgoing) NETBIOS traffic on your firewall(s). This traffic really shouldn't be allowed because of their inherent security risk.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

connect to 515 from inside

Port 515 is used by the printer spooler. People shouldn't be contacting a print server outside the company. This might be an indication that some hosts is compromised and is sending (possibly sensitive) data to a printer outside your company.

Table 3.23 shows all outbound connections to port 515.

Table 3.23: "connect to 515 from inside" connections (all)

connection			# alerts	percent	host or network information
MY.NET.98.190	→	216.181.129.185:515	514	86.97	<< NET-PRIMUSDSL-BLK1 -- PrimusDSL, Inc. >>
MY.NET.97.88	→	216.181.129.185:515	59	9.98	<< NET-PRIMUSDSL-BLK1 -- PrimusDSL, Inc. >>
MY.NET.7.20	→	216.88.97.58:515	15	2.54	bb2h58.coserv.net
MY.NET.162.71	→	209.249.182.79:515	1	0.17	hmotteler.dsl.patriot.net
MY.NET.201.170	→	209.50.66.2:515	1	0.17	pickle.noinfo.com
MY.NET.179.78	→	24.13.123.8:515	1	0.17	cc61691-a.abdn1.md.home.com

Analyzing the complete log we see that on February 2 between 16h25 and 17h56 (print spooler) packets were send from MY.NET.97.88 to 216.181.129.185 and on February 11 between 8h54 and 17h46 from MY.NET.98.190 to the same destination. They always use source port 1025, which is strange. We recommend checking these hosts (MY.NET.98.190 and MY.NET.97.88) because they might be compromised.

MY.NET.7.20 makes 15 connections to 216.88.97.58 on February 3 between 5h27 and 5h39. Besides the fact that it is sending print spooler packets to the outside world, isn't that a little early to be working at the office? Another remarkable thing is that it always uses source port 22. We recommend checking this host also.

Recommendations:

- Check hosts MY.NET.98.190, MY.NET.97.88 and MY.NET.7.20. They might be compromised with some kind of daemon which might be sending classified information to the outside world (more specifically to printers in the outside world).
- Consider blocking all outgoing traffic on port 515 on your firewall(s). Nobody should be sending print jobs to the outside world. While you're at it, consider also blocking all incoming traffic on port 515, because there are known vulnerabilities for several printer spoolers.

Attempted Sun RPC high port access

This alert is triggered on destination port 32771, which is a high port used by Portmapper.

Table 3.24 shows the connections.

Table 3.24: "Attempted Sun RPC high port access" connections (all)

connection			# alerts	percent
64.244.10.40:7777	→	MY.NET.223.254:32771	362	66.67
205.188.153.97:4000	→	MY.NET.221.246:32771	134	24.68
205.188.153.98:4000	→	MY.NET.224.230:32771	20	3.68
205.188.153.105:4000	→	MY.NET.223.70:32771	13	2.39

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

205.188.153.108:4000 →	MY.NET.105.115:32771	6	1.10
205.188.153.107:4 000 →	MY.NET.97.217:32771	5	0.92
205.188.153.109:4000 →	MY.NET.97.207:32771	3	0.55

Most likely these are all false positives. Port 4000 is used by ICQ, instant messaging software (one of the first IM products in the market and currently owned by AOL, of which 205.188.0.0/16 is one of their networks), port 7777 is used by Napster, the (in)famous music (mp3) sharing software. One remarkable fact is the name to which 64.244.10.40 resolves: y0u.g0t.sh0t.sh00ting.0n.d0t.net.

Recommendations:

- As mentioned befo re, consider blocking IM traffic because of well -known vulnerabilities.
- Also mentioned before: consider blocking all Napster (and comparable) traffic because of their inherent security risk (file sharing).

WinGate 1080 Attempt

Port 1080 is used by the So cks/WinGate proxy server. This proxy has known vulnerabilities so scanning your network for proxy servers could be interesting for someone with malicious intent.

Tables 3.25 and 3.26 give you the mostly used source and destination IP addresses.

Table 3.25: Distribution of "WinGate 1080 Attempt" alerts by source IP address (top 5)

source IP address	# alerts	percent	host or network information
199.173.178.2	111	22.24	proxy.monitor.twisted.ma.us.dal.net
63.53.52.128	47	9.42	pool-63.53.52.128.irvn.grid.net
204.117.70.5	44	8.82	security.enterthegame.com
24.1.201.200	29	5.81	<< NETBLK-ATHOME -- @Home Network >>
212.73.162.30	26	5.21	<< PTP-LUND >>

Table 3.26: Distribution of "WinGate 1080 Attempt" alerts by destination IP address (top 5)

dest. IP address	# alerts	percent
MY.NET.98.188	38	7.62
MY.NET.97.80	34	6.81
MY.NET.221.30	29	5.81
MY.NET.15.178	21	4.21
MY.NET.98.118	14	2.81

The above tables give you only the top of the iceberg. Many hosts are involved, both as source and destination (105 sources, 229 destinations), so it is not always clear what exactly is going on.

Host 63.53.52.128 clearly performed a scan. Host 199.173.178.2 however did some scanning before and after February 20, but on that day, (s)he contacted only hosts MY.NET.98.188 and MY.NET.97.80 as shown in table 3.26 (it were the only contacts for those hosts) and did no scanning. Did (s)he find what (s)he wanted and was (s)he successful in compromising those

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

hosts? We don't have enough information to conclude this, but we recommend checkin g those two hosts.

Recommendations:

- Check the hosts in table 3.26 (MY.NET.98.188, MY.NET.97.80, MY.NET.221.30, MY.NET.15.178 and MY.NET.98.118) to find out whether they were compromised or not. All these hosts were contacted several times in a row by only one host (not the same one however for all incidents).

Queso fingerprint

The 'Queso fingerprint' is triggered when Snort sees an inbound packet with the SYN flag set and the two least significant bits of the Reserved Bits set (seen as '21S*****'). In the old days, this wasn't to be seen, but nowadays, due to ECN (Explicit Congestion Notification, RFC 2481), you can encounter legitimate traffic that triggers this alert. Queso uses several flagsettings in trying to determine what OS is running on the target ed system. 'nmap' can produce similar output.

Tables 3.27 and 3.28 give you the mostly used source and destination IP addresses. Table 3.29 gives you the most common destination ports.

Table 3.25: Distribution of "Queso fingerprint" alerts by source IP address (top 5)

source IP address	# alerts	percent	host or network information
194.51.109.194	66	14.07	<< FR-BTPI-ABLEWOOD -- Reseaux d'Acces a l'Internet >>
209.85.60.183	31	6.61	<< NETBLK-SOFTAWARE-BLK3 -- SoftAware, Inc. >>
141.30.228.134	29	6.18	x13l3b.wh2.tu-dresden.de
141.30.228.43	26	5.54	x07r5c.wh2.tu-dresden.de
141.30.228.189	22	4.69	x04m3a.wh2.tu-dresden.de

Table 3.26: Distribution of "Queso fingerprint" alerts by destination IP address (top 5)

dest. IP address	# alerts	percent
MY.NET.229.242	62	13.22
MY.NET.229.158	39	8.32
MY.NET.203.50	25	5.33
MY.NET.162.200	22	4.69
MY.NET.206.30	21	4.48

Table 3.27: Distribution of "Queso fingerprint" alerts by destination port (top 5)

destination port	# alerts	percent	service	service description
6346	271	57.78	gnutella-svc	
6355	81	17.27		
25	20	4.26	smtp	Simple Mail Transfer
113	14	2.99	ident	
			auth	Authentication Service
6347	5	1.07	gnutella-rtr	

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

When analyzing the traffic towards MY.NET.229.242, all coming from 194.51.109.194, using the alert log and packet dump files, this just seems legitimate traffic, using ECN. This was all Gnutella traffic.

The traffic towards MY.NET.229.158 however, looks like a port scan:

```
02/20-03:55:55.715780 209.85.60.179:1978  -> MY.NET.229.158:2715
02/23-04:12:33.271588 209. 85.60.183:1978 -> MY.NET.229.158:1971
02/23-04:44:44.230721 209.85.60.183:1978  -> MY.NET.229.158:2252
02/23-04:47:50.657389 209.85.60.183:1978  -> MY.NET.229.158:2284
02/23-05:58:36.984479 209.85.60.183:1978  -> MY.NET.229.158:2853
…                     (29 entries omitted)
02/27-20:09:08.769685 209.85.60.179:1978  -> MY.NET.229.158:1095
02/27-20:13:39.832795 209.85.60.179:1978  -> MY.NET.229.158:1118
02/27-20:23:07.487946 209.85.60.179:1978  -> MY.NET.229.158:1189
02/27-20:24:03.200493 209.85.60.179:1978  -> MY.NET.229.158:1205
02/27-20:59:20.206533 209.85.60.179:1978  -> MY.NET.229.158:1557
```

The aggressing hosts shown above were the only two involved in this scan/OS fingerprinting. Remark also that it is a slow scan (mostly tens of minutes between to attempts).

Some of the alerts are triggered correctly; others are false positives. When you do have a scan, it is only a reconnaissance sweep and not an actual attempt to compromise your system. It might however been an indication that an attempt will take place…

Tiny Fragments - Possible Hostile Activity

This alert might be an indication that someone is trying to flood some of your systems with lots of tiny IP fragments (some network-connected devices might have a hard time reassembling all those tiny fragments). It is often used in combination with other attacks to evade IDS detection.

Table 3.28 gives you an overview of who targeted which of your servers.

Table 3.28: "Tiny Fragments – Possible Hostile Activity" connections (top 5)

connection			# alerts	percent	host or network information
212.89.165.5	→	MY.NET.223.42	116	50.66	eptanissa.altec.gr
64.80.90.36	→	MY.NET.98.117	53	23.14	<< NETBLK-PAETECCOMM -- PaeTec Communications, Inc. >>
64.80.90.36	→	MY.NET.97.231	20	8.73	<< NETBLK-PAETECCOMM -- PaeTec Communications, Inc. >>
202.205.5.10	→	MY.NET.1.8	6	2.62	<< TNTS-CN -- Tsinghua Network Training & Services >>
64.80.88.99	→	MY.NET.206.254	5	2.18	<< NETBLK-PAETECCOMM -- PaeTec Communications, Inc. >>

The possible attacks were done in each case spread over a few minutes. This traffic might be legitimate or malicious. It is hard to tell without further information. We would suggest malicious intent was in mind, because of the several minutes between the first arrival and last arrival, but that could also happen in normal traffic.

No correlation to other (kind of) attacks could be made when looking at the aggressing hosts. There were some other possible attacks against the targets, but most of them were just scans.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Recommendations:

- You might investigate how well MY.NET.223.42, MY.NET.98.117 and MY.NET.97.23 1 can handle tiny fragments, but for now, I wouldn't consider them threatened.

Null scan!

Null scan alerts indicate an inbound packet with no flags set at all. This is never normal behavior, but could be caused by many different things (malfunctioning ha rdware, software bugs, hacking tools, etc…).

Nothing sensible could be concluded from these alerts. There were in total 135 different alerts, using 118 different source addresses and 90 different destination addresses. 61 different destination ports were used of which the most common were 0 (Reserved, shouldn't be used anywhere, 32 alerts), 6346 (gnutella -svc, 21 alerts) and 21504 (unassigned, 17 alerts).

The only sensible thing we can say is that it shouldn't occur, but it does; which is not that comforting.

SUNRPC highport access!

This alert is triggered on destination port 32771, which is a high port used by Portmapper.

Table 3.29 shows all connections to destination port 32771 (which isn't shown in the table for reasons of space).

Table 3.29: "SUNR PC highport access!" connections (all)

connection			# alerts	percent	host or network information
4.9.158.233:22	→	MY.NET.163.17	101	90.18	cc916074-a.catv1.md.home.com
152.163.241.90:5190	→	MY.NET.98.122	3	2.68	<< NET-ANS-BNET8 -- America Online >>
205.188.5.157:1501	→	MY.NET.98.227	2	1.79	<< NETBLK-AOL-DTC -- America Online, Inc >>
216.136.171.195:5190	→	MY.NET.100.225	2	1.79	usw-sf-ftp1.sourceforge.net
MY.NET.70.38:36340	→	MY.NET.103.112	1	0.89	
MY.NET.70.38:36338	→	MY.NET.103.112	1	0.89	
24.9.203.188: 6120 7	→	MY.NET.165.129	1	0.89	c322300-a.moline1.il.home.com
200.233.81.13:13765	→	MY.NET.60.17	1	0.89	<< Not assigned >>

Most likely, all the alerts for the first entry in table 3.29 are false positives (they are the return packets for the ssh (port 22) conn ection from MY.NET.163.17 to 4.9.158.233). All the alerts for the second entry are probably return packets for an AOL IM connection. Both internal accesses are most likely real RPC calls.

This is the traffic from the others:

```
01/30-14:34:29.280204 200.233 .81.13:13765  -> MY.NET.60.17:32771
01/30-19:19:16.387947 24.9.203.188:61207   -> MY.NET.165.129:32771
02/03-22:17:09.957552 205.188.5.157:5190   -> MY.NET.98.227:32771
02/03-22:17:10.679807 205.188.5.157:5190   -> MY.NET.98.227:32771
03/06-01:53:39.846281 216.13 6.171.195:1501  -> MY.NET.100.225:32771
03/06-01:53:39.923576 216.136.171.195:1501   -> MY.NET.100.225:32771
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

These might be all real sunrpc calls, so we recommend checking them out.

Recommendations:

- Consider verifying whether RPC calls to MY.NET.60.17, MY.NET.165.129, MY.NET.98.227 and MY.NET.100.225 from the outside could be legitimate.
- Consider blocking all external RPC access to your servers, or, if necessary, only allowing access from certain external hosts to certain internal hosts. There might be circumstances where inbound RPC calls are necessary.

ICMP SRC and DST outside network

This is ICMP traffic with source and destination addresses considered outside your network(s) by the Snort IDS. It is related to the "UDP SRC and DST outside network" and "TCP SRC and DST outside network" alerts. ICMP stands for Internet Control Message Protocol and it communicates error messages and other conditions that require attention. It might be necessary to allow certain ICMP traffic into your network, but if source and destination are external, for most companies, this kind of traffic can be blocked.

Tables 3.30 and 3.31 give you an impression of this ICMP traffic.

Table 3.30: Distribution of "ICMP SRC and DST outside network" alerts by source IP address (top 5)

source IP address	# alerts	percent	host or network information
10.3.41.11	26	32.10	[[RESERVED -6]]
10.0.0.1	21	25.93	[[RESERVED -6]]
128.249.101.1	4	4.94	ben101.bcm.tmc.edu
128.249.104.1	3	3.70	ben104.bcm.tmc.edu
10.10.5.3	3	3.70	[[RESERVED -6]]

Table 3.31: Distribution of "ICMP SRC and DST outside network" alerts by destination IP address (top 5)

source IP address	# alerts	percent	host or network information
10.1.40.102	26	32.10	[[RESERVED -6]]
209.143.81.2	20	24.69	third.greenmount.org
224.2.127.254	16	19.75	SAP.MCAST.net
192.63.42.145	3	3.70	<< NETBLK-UNISYS-NET2 -- UNISYS >>
24.228.9.100	2	2.47	<< NETBLK-CVISP -- Cablevision Systems >>

Again, we see some private addresses (10.0.0.0/8 addresses) and also a multicast address (224.2.127.254). There is traffic from internal (private) addresses to external addresses and vice versa, which is seen by Snort as external traffic because of the missing private address range to its NET_HOME variable. This traffic is most probably normal. The pure external traffic however shouldn't pass your network.

Most likely your most external routers are malfunctioning or wrongly configured and sending packets destined to the outside world in your network.

Recommendations:

- Consider blocking all (real) external ICMP traffic on the perimeter firewall(s), if you have one (any).

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

- Most probably, some routers are wrongly configured. The most outward router(s) should only send packets with an internal destination address into the internal network. You might consider checking them .

Back Orifice

Back Orifice is a trojan developed by the hacking group "Cult of the Dead Cow" and released in August 1998. Systems are infected in the normal Trojan Horse manner: a person downloads or is sent an executable from the Internet. Once the exe cutable runs, it invisibly runs on the system, providing full access to outside hackers. This alert triggers when it sees a packet with destination port 31337, which is hackers spelling for 'elite' (meaning 'elite hackers').

Tables 3.32 tells you who was looking for Back Orifice on your systems.

Table 3.32: Distribution of "Back Orifice" alerts by source IP address (all)

source IP address	# alerts	percent	host or network information
203.170.152.87	16	64.00	<< CSC -- C.S.Communications Co., Ltd. >>
63.10.224.59	9	36.00	1Cust59.tnt2.tacoma.wa.da.uu.net

Looking at the logs, we clearly saw they performed a host scan against some hosts of the MY.NET.97.0/24 and MY.NET.98.0/24 subnets. 25 hosts were scanned in total.

STATDX UDP attack

Due to a format stri ng vulnerability in a call to syslog() within its logging module, rpc.statd on various Linux systems can be exploited remotely. statd is an ONC RPC server that implements the Network Status Monitor RPC protocol to provide reboot notification. The NFS file locking service (rpc.lockd) uses it when performing lock recovery. This alert triggers on a string specifically for this exploit.

For more information about this vulnerability, see: http://www.security focus.com/bid/1480 .

This is the complete log of these events:

```
02/20-19:35:35.660074 129.105.107.190:859  -> MY.NET.60.75:798    *
02/20-19:41:44.749045 171.65.61.201:809  -> MY.NET.60.75:1007
02/20-19:41:51.812847 171.65.61.201:833  -> MY.NET.60.58:800
02/20-19:42:33.320412 171.65.61.201:871  -> MY.NET.105.91:798    **
02/20-19:42:33.683596 171.65.61.201:873  -> MY.NET.105.169:32774  *
02/20-19:42:51.102298 171.65.61.201:891  -> MY.NET.130.81:941
02/20-19:43:04.460052 171.65.61.201:904  -> MY.NET.140.29:797
02/20-19:45:33.132877 171.65.61.201:936  -> MY.NET.181.127:910
```

I refer to table 3.20 (*Distribution of "External RPC call" alerts by source IP address*) to see the following correlation:

```
02/20-19:35:34.663346 129.105.107.190:4801  -> MY.NET.60.191:111
02/20-19:35:35.975389 129.105.107.190:4685  -> MY.NET.60.75:111   *
02/20-19:35:37.222154 129.105.107.190:2726  -> MY.NET.68.2:111
…
02/20-19:42:31.051543 171.65.61.201:3320  -> MY.NET.99.176:111
02/20-19:42:32.945358 171.65.61.201:4792  -> MY.NET.105.91:111   **
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
02/20-19:42:34.124788 171.65.61.201:4894   -> MY.NET.105.169:111    *
02/20-19:42:34.142670 171.65.61.201:1274   -> MY.NET.106.218:111
```

The lines marked with an asterisk are the important ones. One would say that after the scan they tried an exploit (probably after a successful response), but the timings aren't always correct. It is only correct for host MY.NET.105.91 (two asterisks). The others are off by one second. Maybe the timing isn't 100% reliable.

We can't be sure that these hosts were compromised, but better safe than sorry…

Recommendations:

- It would be wise to verify whether these hosts (MY.NET.60.75, MY.NET.60.75, MY.NET.60.58, MY.NET.105.91, MY.NET.105.169, MY.NET.130.81, MY.NET.140.29 and MY.NET.181.127) were compromised or not.
- As mentioned before, consider blocking all inbound RPC access to machines that run the statd daemon (or any daemon, for that matter) or put some kind of filtering device in front of it. Even better would be to block all inbound RPC access, but this isn't always feasible.

TCP SMTP Source Port traffic

This alert apparently is triggered when an inbound TCP packet is seen with source port 25 (SMTP) and a well known port as destination port. We say 'apparently', because we haven't found the exact signature that triggers this alert, so we are not sure if there are more requirements.

Only four alerts were triggered:

```
01/30-14:31:36.054897 11.125.218.156:25   -> MY.NET.60.17:274
01/30-14:34:09.165435 17.135.218.56:25   -> MY.NET.60.17:979
02/03-05:46:31.726285 195.211.49.18:25   -> MY.NET.139.54:1007
02/04-05:37:48.374429 200.251.185.30:25   -> MY.NET.158.238:399
```

Table 3.33 shows the host or network information for the source addresses.

Table 3.33: Distribution of "TCP SMTP Source Port traffic" alerts by source IP address (all)

src. IP address	# alerts	percent	host or network information
11.125.218.156	1	25	<< NET-DODIIS -- DoD Intel Information Systems >>
17.135.218.56	1	25	<< NET-APPLE-WWNET -- Apple Computer, Inc. >>
195.211.49.18	1	25	ipanema.homeshopping.com.br
200.251.185.30	1	25	ist.leidenschaftlich.gern.chanop.de

Only host 195.211.49.18 seems to be an SMTP server, which indicates that this traffic might be a return packet (1007 is a reserved port, but some operating systems use those ports if the host isn't listening on them). Of the destination ports, only port 399 is assigned (to iso-tsap-c2 or ISO Transport Class 2 Non-Control over TCP). Because there are no packet dumps, there is not much more we are able to deduct…

Security 000516-1

We didn't find any information about this kind of alerts, but seeing the alerts (all alerts are shown), most likely this is Napster traffic (napster uses a/o. port 6699):

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
02/23-17:27:15.666379 140.247.187.110:6699   -> MY.NET.206.74:1699
02/23-17:27:16.186863 140.247.187.110:6699   -> MY.NET.206.74:1699
02/23-17:27:16.188285 MY.NET.206.74:1699   -> 140.247.187.110:6699
02/23-17:27:16.234242 140.247.187.110:6699   -> MY.NET.206.74:1699
```

Recommendations:

- As mentioned before, consider blocking all Napster, Gnutella and related traffic on your firewall(s). This traffic shouldn't be allowed because of their inherent security risk (file sharing).

Probable NMAP fingerprint attempt

This alert is triggered when an inbound TCP packet with the SYN, FIN, ACK and PUSH flags set, is seen. This packet is in violation with the RFC and indicates most likely the use of nmap, one of the more popular reconnaissance tools. It can perform OS fingerprinting and all different kinds of port and hosts scans.

These were the logged alerts:

```
02/27-06:49:52.479962 24.169.163.127:0   -> MY.NET.227.78:6346
03/07-06:40:45.127533 24.240.49.169:6699   -> MY.NET.207.150:3061
```

It is interesting to see that we can correlate the first alert to the following alert from the "Null Scan!" alert file:

```
02/27-06:49:24.322292 24.169.163.127:6346   -> MY.NET.227.78:4669
```

There is almost half a minute between the null scan and the nmap fingerprint attempt. Beside the SYN-FIN-PUSH-ACK combination, source port 0 also indicates that something's fishy. The packet dump also showed quite some EOL fields in the TCP options. This packet was definitely crafted…

Recommendations:

- A good firewall can protect you against this kind of reconnaissance sweeps. Any packet that deviates from the RFC would be dropped; which makes fingerprinting a lot harder.

SITE EXEC - Possible wu-ftpd exploit - GIAC000623

wu-ftpd version 2.6.0 has a very well known exploit that might enable someone to execute arbitrary commands as root. This is very serious and your FTP server could become completely compromised. For more information concerning this alert, we refer to http://www.securityfocus.com/bid/1387 .

This is what the Snort IDS saw:

```
03/06-16:44:02.658052 128.61.136.233:4705   -> MY.NET.219.22:21
```

128.61.136.233 resolves to tann6233.mse.gatech.edu. This can be correlated to the SYN-FIN scan that took place between 03/06-16:07:53 and 03/06-16:29:23. Apparently, (s)he found a FTP server and then tried an exploit.

We strongly recommend verifying whether this host (MY.NET.219.22) was compromised or not. If the server wasn't patched against this kind of attack, it is most probably compromised.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Recommendations:

- You should verify whether the host MY.NET.219.22 is compromised or not. If it is running a wu-ftpd daemon, version 2.6.0 or lower, we recommend shutting it down and taking appropriate actions (see previously mentioned URL).
- You should always get the latest patch for every server (daemon) that you are running. This will seriously reduce the risk of getting hacked.

Russia Dynamo - SANS Flash 28-jul-00

We didn't find any information about this alert, nor at SANS itself, nor at securityfocus, whitehats or CVE. Anyway, this is the alert seen by the Snort IDS:

02/03-20:46:15.618252 MY.NET.203.50:6346 -> 194.87.6.79:1791

Port 6346 is associated with gnutella-svc. Most likely this is a return packet and MY.NET.203.50 is running Gnutella. We didn't find a dump of this packet, but there were other dumps from traffic going to MY.NET.203.50 at port 6346.

We recommend checking the machine out. Better be safe than sorry.

Recommendations:

- Because of seeing an alert of which we couldn't find any information, we recommend checking it out anyway. Most likely this is just Gnutella traffic from MY.NET.203.50, but better safe than sorry.
- Once again, you might consider blocking all Napster, Gnutella and related traffic on your firewall(s). This traffic shouldn't be allowed because of their inherent security risk (file sharing).

Analysis of the scan filess

First we will give some general statistical information concerning the scan files.

Table 3.34 shows the distribution of the scans by source IP address. This normally indicates that they are used for scanning (as the scanner). Only the top 30 is shown. For your convenience, the hostname corresponding to the IP address is shown for each entry (*nslookup* information). If the hostname couldn't be retrieved, the name of the network block to which the host belongs is shown (*whois* information).

Table 3.34: Distribution of scans by source IP address (top 30)

source IP address	# alerts	percent	host or network information
MY.NET.218.90	34517	2.90	
MY.NET.150.220	24185	2.03	
MY.NET.221.26	21060	1.77	
MY.NET.204.66	20040	1.68	
MY.NET.229.154	19785	1.66	
MY.NET.70.38	15606	1.31	
MY.NET.150.133	15130	1.27	
MY.NET.202.50	14282	1.20	
MY.NET.227.254	12721	1.07	
169.226.202.234	12129	1.02	sumac.asrc.cestm.albany.edu
MY.NET.150.225	11512	0.97	
MY.NET.206.42	11246	0.94	

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

MY.NET.212.206	10897	0.91	
MY.NET.228.54	10770	0.90	
MY.NET.228.214	10005	0.84	
206.112.192.106	9992	0.84	shell.one.net
MY.NET.220.142	9793	0.82	
MY.NET.210.250	9679	0.81	
130.234.184.112	9446	0.79	termos.keltti.jyu.fi
MY.NET.100.230	9313	0.78	
MY.NET.223.34	9303	0.78	
MY.NET.150.143	8893	0.75	
24.141.226.62	8642	0.73	d141-226-62.home.cgocable.net
MY.NET.217.74	8523	0.72	
MY.NET.210.190	8211	0.69	
MY.NET.228.122	8036	0.67	
MY.NET.205.174	8003	0.67	
MY.NET.204.150	7922	0.66	
MY.NET.219.130	7894	0.66	
MY.NET.179.78	7827	0.66	

It is remarkable how many of your machines are used as source for a scanning incident.

Table 3.35 shows the distribution of the scans by destination IP address. Only the top 30 is shown. This probably indicates that they were the victim of a port scan.

Table 3.35: Distribution of scans by destination IP address (top 30)

source IP address	# alerts	percent	host or network information
129.2.246.94	21060	1.77	NotRegistered-129-2-246-94.student.umd.edu
MY.NET.160.109	9995	0.84	
MY.NET.60.8	8814	0.74	
216.155.34.54	4079	0.34	forsaken.magpage.com
169.197.49.83	3879	0.33	ppp338.mc01.dsl.azstarnet.com
MY.NET.218.86	3032	0.25	
24.157.10.197	2341	0.20	cr483660-a.hnsn1.on.wave.home.com
63.71.84.102	2180	0.18	mercury.kci.com
24.156.151.85	2172	0.18	<< NETBLK-ROGERS-6-BLOCK -- Rogers@Home >>
24.21.239.107	2112	0.18	c310358-a.btnrug1.la.home.com
216.19.133.116	2041	0.17	<< NETBLK-EC08-1 -- Exodus Communications Inc. Herdon >>
172.132.71.130	2012	0.17	AC844782.ipt.aol.com
63.121.232.185	1953	0.16	newburgh-b-185.sigecom.net
24.91.199.203	1833	0.15	h00207812c97f.ne.mediaone.net
63.71.84.104	1815	0.15	voyager.kci.com
63.21.61.147	1729	0.15	<< NETBLK-NETBLK-UUNET97DU -- UUNET Technologies, Inc. >>
63.71.84.103	1685	0.14	venus.kci.com
65.1.199.105	1642	0.14	cc79015-a.chstfld1.va.home.com
172.141.108.212	1636	0.14	AC8D6CD4.ipt.aol.com
194.251.249.182	1588	0.13	parallel.subspace.edome.net
172.169.147.76	1580	0.13	ACA9934C.ipt.aol.com
24.3.45.174	1538	0.13	cc265407-a.hwrd1.md.home.com
142.103.36.176	1533	0.13	srtb0211-176.resnet.ubc.ca
209.163.147.37	1518	0.13	147-37.waldenweb.com

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

4.42.37.229	1512	0.13	lsanca1-ar15-037-229.lsanca1.dsl.gtei.net
66.24.125.138	1489	0.12	roc-66-24-125-138.rochester.rr.com
24.19.99.230	1451	0.12	c204103-a.roalok1.mi.home.com
208.231.55.57	1446	0.12	devfvapserver.iren.org
63.14.172.15	1425	0.12	1Cust15.tnt1.phoenix.az.da.uu.net
66.30.167.225	1401	0.12	h00104c12be82.ne.mediaone.net

Some machines are yours, but most of them are external, so most likely some of your machines are performing port scans.

Table 3.36 shows the distribution of the scans by source port. Only the top 15 is shown. Some tools use specific source ports, so you might guess which tool it is, just by looking at the source port (this is guessing; look at the packet to be sure!).

Table 3.36: Distribution of scans by source port (top 15)

source port	# alerts	percent
27888	139432	11.70
28800	132429	11.11
13139	55614	4.67
0	36587	3.07
1036	34149	2.87
6112	33811	2.84
21	29716	2.49
28001	23492	1.97
9001	20936	1.76
1025	13283	1.11
53	9690	0.81
27010	9581	0.80
1676	8688	0.73
2000	8599	0.72
2002	8498	0.71

Table 3.37 shows the distribution of the scans by destination port. Only the top 30 is shown. This is interesting because it will tell you for which services they are searching. For most of the well known ports (< 1024) there are well known vulnerabilities. Some of the ports shown in table 3.37 are in use by trojans. If something is listening on those ports, that could indicate that someone already compromised the machine, which would make life easier for you if you had malicious intent.

Possible trojans are listed in *italic*.

Table 3.37: Distribution of scans by destination port (top 10)

destination port	# alerts	percent	service	service description
28800	135330	11.36		
7778	62138	5.21		
13139	51951	4.36		
0	36229	3.04	(reserved)	
53	35392	2.97	domain	Domain
21	34227	2.87	ftp	File Transfer [Control] *Back Construction, Blade Runner, Cattivik FTP Server, CC Invader,*

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

			Dark FTP, Doly Trojan, Fore, Invisible FTP, Juggernaut 42, Larva, MotIv FTP, Net Administrator, Ramen, Senna Spy FTP server, The Flu, Traitor 21, WebEx, WinCrash
6112	32592	2.74	
27018	19957	1.67	
32768	19659	1.65	
27020	17370	1.46	

Table 3.38 shows the distribution of the scans by destination port for whom the ports are known to be possibly used by trojans. Only the top 20 is shown.

Table 3.38: Distribution of scans by destination port (top 20)

destination port	# alerts	percent	trojan
21	34227	2.87	*Back Construction, Blade Runner, Cattivik FTP Server, CC Invader, Dark FTP, Doly Trojan, Fore, Invisible FTP, Juggernaut 42, Larva, MotIv FTP, Net Administrator, Ramen, Senna Spy FTP server, The Flu, Traitor 21, WebEx, WinCrash*
7000	4896	0.41	*Exploit Translation Server, Kazimas, Remote Grab, SubSeven, SubSeven 2.1 Gold*
1080	4363	0.37	*WinHole*
27374	4021	0.34	*Bad Blood, Ramen, Seeker, SubSeven, SubSeven 2.1 Gold, Subseven 2.1.4 DefCon 8, SubSeven Muie, Ttfloader*
1	3322	0.28	*Sockets des Troie (UDP)*
1025	3243	0.27	*Remote Storm*
7777	2938	0.25	*God Message, Tini*
1053	2730	0.23	*The Thief*
25	2506	0.21	*Ajan, Antigen, Barok, Email Password Sender - EPS, EPS II, Gip, Gris, Happy99, Hpteam mail, Hybris, I love you, Kuang2, Magic Horse, MBT (Mail Bombing Trojan), Moscow Email trojan, Naebi, NewApt worm, ProMail trojan, Shtirlitz, Stealth, Tapiras, Terminator, WinPC, WinSpy*
1054	2328	0.20	*AckCmd*
23	2134	0.18	*Fire HacKer, Tiny Telnet Server - TTS, Truva Atl*
6970	1439	0.12	*GateCrasher*
1099	1395	0.12	*Blood Fest Evolution, Remote Administration Tool - RAT*
54321	1394	0.12	*Back Orifice 2000, School Bus*
1095	1354	0.11	*Remote Administration Tool - RAT*
30000	1289	0.11	*Infector*
1090	1100	0.09	*Xtreme*
2155	1055	0.09	*Illusion Mailer*
1082	1033	0.09	*WinHole*
2000	973	0.08	*Der Späher / Der Spaeher, Insane Network, Last 2000, Remote Explorer 2000, Senna Spy Trojan Generator*

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Because of the tremendous amount of scan information, we will only analyze some scans which look really interesting.

The following is a tremendous scan.

```
Feb  4 11:37:52 MY.NET.218.90:1036  -> 172.155.98.112:2039 UDP
Feb  4 11:37:52 MY.NET.218.90:1036  -> 208.248.195.240:1027 UDP
Feb  4 11:37:52 MY.NET.218.90:1036  -> 172.162.57.165:1335 UDP
Feb  4 11:37:52 MY.NET.218.90:1036  -> 172.164.98.34:1621 UDP
Feb  4 11:37:48  MY.NET.218.90:1036  -> 24.88.100.40:1618 UDP
Feb  4 11:37:49 MY.NET.218.90:1036  -> 161.184.182.53:1028 UDP
Feb  4 11:37:50 MY.NET.218.90:1036  -> 209.126.78.230:1036 UDP
Feb  4 11:37:55 MY.NET.218.90:1036  -> 208.248.195.240:1027 UDP
Feb  4 11:37:55 MY.NET.21 8.90:1036 -> 172.155.98.112:2039 UDP
Feb  4 11:37:55 MY.NET.218.90:1036  -> 172.164.98.34:1621 UDP
Feb  4 11:37:55 MY.NET.218.90:1036  -> 172.162.57.165:1335 UDP
Feb  4 11:37:58 MY.NET.218.90:1036  -> 172.164.98.34:1621 UDP
Feb  4 11:37:59 MY.NET.218.90:1036  -> 208.248.195.240:1027 UDP
Feb  4 11:37:58 MY.NET.218.90:1036  -> 172.155.98.112:2039 UDP
Feb  4 11:37:57 MY.NET.218.90:1036  -> 172.162.57.165:1335 UDP
Feb  4 11:38:01 MY.NET.218.90:1036  -> 172.164.98.34:1621 UDP
Feb  4 11:38:01 MY.NET.218.90:1036  -> 208.2 48.195.240:1027 UDP
Feb  4 11:38:00 MY.NET.218.90:1036  -> 172.162.57.165:1335 UDP
Feb  4 11:38:01 MY.NET.218.90:1036  -> 172.155.98.112:2039 UDP
Feb  4 11:38:04 MY.NET.218.90:1036  -> 172.162.57.165:1335 UDP
...                   (34021 entries omitted)
Feb  4 19:31:19 MY. NET.218.90:1036 -> 165.121.82.187:1025 UDP
Feb  4 19:31:20 MY.NET.218.90:1036  -> 209.122.69.88:1170 UDP
Feb  4 19:31:20 MY.NET.218.90:1036  -> 66.24.86.89:2227 UDP
Feb  4 19:31:20 MY.NET.218.90:1036  -> 63.204.177.73:1148 UDP
Feb  4 19:31:22 MY.NET.218.90:10  36 -> 208.191.76.2:3366 UDP
Feb  4 19:31:22 MY.NET.218.90:1036  -> 24.6.245.220:2798 UDP
Feb  4 19:31:25 MY.NET.218.90:1036  -> 208.191.76.2:3366 UDP
Feb  4 19:31:26 MY.NET.218.90:1036  -> 24.6.245.220:2798 UDP
Feb  4 19:31:28 MY.NET.218.90:1036  -> 208.191.76 .2:3366 UDP
Feb  4 19:31:28 MY.NET.218.90:1036  -> 24.6.245.220:2798 UDP
Feb  4 19:31:29 MY.NET.218.90:1036  -> 64.7.13.73:52980 UDP
Feb  4 19:31:31 MY.NET.218.90:1036  -> 208.191.76.2:3366 UDP
Feb  4 19:31:29 MY.NET.218.90:1036  -> 24.218.212.178:1029 UDP
Feb  4 19:31:31 MY.NET.218.90:1036  -> 24.6.245.220:2798 UDP
Feb  4 19:31:31 MY.NET.218.90:1036  -> 63.225.87.194:2383 UDP
Feb  4 19:31:32 MY.NET.218.90:1036  -> 24.179.170.205:2503 UDP
Feb  4 19:31:34 MY.NET.218.90:1036  -> 208.191.76.2:3366 UDP
Feb  4 19:31:34  MY.NET.218.90:1036  -> 24.6.245.220:2798 UDP
Feb  4 19:31:34 MY.NET.218.90:1036  -> 143.195.110.35:3183 UDP
Feb  4 19:31:37 MY.NET.218.90:1036  -> 24.140.31.205:1193 UDP
```

In 8 hours more than 34 thousand hosts were scanned. This is a tremendous host scan. Wha t is more important is that the source is inside your network. We don't know which tool is used (not enough information).

The second entry in table 3.34 (MY.NET.150.220) uses the same port for source and destination (28800). The third (MY.NET.221.26) is d oing a SYN scan.

169.226.202.234 did a host scan against your network on port 21. Probably idlescan or synscan was used (reflexive ports, SYN and FIN flag set).

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Another interesting scan took place on February 23 between 22h57 and 23h17. 2385 hosts were scanned for the presence of a trojan listening on port 27374. Most likely searching for some SubSeven variant, which is a very dangerous trojan…

Looking at table 3.35 we see that 129.2.246.94 is targeted more than 20 thousand times. Looking at the entries in the logs, we see the following:

```
Feb 10 03:19:09 MY.NET.221.26:1103   -> 129.2.246.94:12 SYN **S*****
Feb 10 03:19:09 MY.NET.221.26:1106   -> 129.2.246.94:15 SYN **S*****
Feb 10 03:19:12 MY.NET.221.26:1127   -> 129.2.246.94:36 SYN **S*****
Feb 10 03:19:12 MY. NET.221.26:1141  -> 129.2.246.94:50 SYN **S*****
Feb 10 03:19:09 MY.NET.221.26:1144   -> 129.2.246.94:53 SYN **S*****
Feb 10 03:19:10 MY.NET.221.26:1147   -> 129.2.246.94:56 SYN **S*****
Feb 10 03:19:10 MY.NET.221.26:1150   -> 129.2.246.94:59 SYN **S*****
Feb 10  03:19:10 MY.NET.221.26:1151  -> 129.2.246.94:60 SYN **S*****
Feb 10 03:19:10 MY.NET.221.26:1158   -> 129.2.246.94:67 SYN **S*****
Feb 10 03:19:10 MY.NET.221.26:1159   -> 129.2.246.94:68 SYN **S*****
...                        (21040 entries omitted)
Feb 10 14:04:49 MY.NET.221.26:4  542 -> 129.2.246.94:12029 SYN **S*****
Feb 10 14:04:50 MY.NET.221.26:4557   -> 129.2.246.94:12044 SYN **S*****
Feb 10 14:04:50 MY.NET.221.26:4563   -> 129.2.246.94:12050 SYN **S*****
Feb 10 14:04:50 MY.NET.221.26:4564   -> 129.2.246.94:12051 SYN **S*****
Feb 10  14:04:50 MY.NET.221.26:4520  -> 129.2.246.94:12007 SYN **S*****
Feb 10 14:04:50 MY.NET.221.26:4544   -> 129.2.246.94:12031 SYN **S*****
Feb 10 14:04:50 MY.NET.221.26:4583   -> 129.2.246.94:12070 SYN **S*****
Feb 10 14:04:51 MY.NET.221.26:4597   -> 129.2.246.94:12 084 SYN **S*****
Feb 10 14:04:50 MY.NET.221.26:4576   -> 129.2.246.94:12063 SYN **S*****
Feb 10 14:04:51 MY.NET.221.26:4581   -> 129.2.246.94:12068 SYN **S*****
```

MY.NET.221.26 is performing a SYN port scan against 129.2.246.94 (NotRegistered -129-2-246-94.stude nt.umd.edu).

Recommendations:

- We strongly advise you to check whether MY.NET.218.90 is compromised or not. The same holds for MY.NET.150.220 and MY.NET.221.26. It might be best to check all your machines of table 3.34 whether they are compromised or someo ne of your colleagues/employees was doing some scanning "for fun".
- You might investigate why MY.NET.221.26 did a port scan against 129.2.246.94 (NotRegistered -129-2-246-94.student.umd.edu); if someone is watching, it might backfire on you (if it happens of ten, you might get a bad reputation, which you wouldn't want).

Analysis of the packet dumps

Because of the tremendous amount of packet dumps, we will only analyze those which look really interesting.

```
01/20-00:23:09.169938 MY.NET.217.150:2340  -> 193.6.61.53:2119
TCP TTL:126 TOS:0x0 ID:5248  DF
21SFRPAU Seq: 0x2A9CF07   Ack: 0xA0E7E742   Win: 0x5010
09 24 08 47 02 A9 CF 07 A0 E7 E7 42 00 FF 50 10  .$.G.......B..P.
FD 8B D3 63 00 00 05 64 0B F9 61 7A CF 15 08 F8  ...c...d..az....
42 A8                                             B.

01/20-00:28:00.593242 MY.NET.217.150:2340  -> 193.6.61.53:2119
TCP TTL:126 TOS:0x0 ID:10627  DF
21SFR**U Seq: 0x31D73F3   Ack: 0xA0E7E742   Win: 0x5010
09 24 08 47 03 1D 73 F3 A0 E7 E7 42 00 E7 50 10  .$.G..s....B..P.
FD 8B E0 29  00 00 64 C3 8D AA D2 85 18 06 B8 DD  ...)..d.........
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
69 21                                                  i!

01/20-01:44:30.974511 MY.NET.217.150:2340  -> 193.6.61.53:2323
TCP TTL:126 TOS:0x0 ID:45252  DF
**SFR*AU Seq: 0x11011D   Ack: 0xA804FB46   Win: 0x5018
TCP Options => EOL EOL Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 Opt 57
Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 Opt
57 Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 Opt 57
Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 Opt 57 O  pt 57 Opt
57 Opt 57 Opt 57 Opt 57

01/20-01:55:09.047106 MY.NET.217.150:2340  -> 193.6.61.53:2323
TCP TTL:126 TOS:0x0 ID:46330  DF
**SFR*AU Seq: 0xD00239   Ack: 0x1E94FB46   Win: 0x5010
TCP Options => EOL EOL
```

This looks like **you might have some broken hard ware**. Setting the reserved bits might be legitimate, but setting all flags never is, neither is the SFRU combination. Some weird TCP options also. More than two thousand strange packets were seen coming from **MY.NET.217.150** , so it's worth checking out.

```
01/23-11:22:30.428449 129.104.19.94:109  -> MY.NET.1.1:109
TCP TTL:22 TOS:0x0 ID:39426
**SF**** Seq: 0x4626BCBD   Ack: 0x41947B80   Win: 0x404
00 00 00 00 00 00                                  ......

01/23-11:22:30.444284 129.104.19.94:109  -> MY.NET.1.2:109
TCP TTL:22 TOS:0x0 ID:39426
**SF**** Seq: 0x4626BCBD   Ack: 0x41947B80   Win: 0x404
00 00 00 00 00 00                                  ......

01/23-11:22:30.475321 129.104.19.94:109  -> MY.NET.1.3:109
TCP TTL:22 TOS:0x0 ID:39426
**SF**** Seq: 0x4626BCBD   Ack:  0x41947B80   Win: 0x404
00 00 00 00 00 00                                  ......

...             (11040 packets omitted)

01/23-11:44:05.740670 129.104.19.94:109  -> MY.NET.254.249:109
TCP TTL:22 TOS:0x0 ID:39426
**SF**** Seq: 0x532EA0D6   Ack: 0x78F131B3   Win: 0x4  04
00 00 00 00 00 00                                  ......

01/23-11:44:05.740737 129.104.19.94:109  -> MY.NET.254.250:109
TCP TTL:22 TOS:0x0 ID:39426
**SF**** Seq: 0x532EA0D6   Ack: 0x78F131B3   Win: 0x404
00 00 00 00 00 00                                  .  .....

01/23-11:44:05.866015 129.104.19.94:109  -> MY.NET.254.255:109
TCP TTL:22 TOS:0x0 ID:39426
**SF**** Seq: 0x532EA0D6   Ack: 0x78F131B3   Win: 0x404
00 00 00 00 00 00
```

This is obviously a scan. We showed this one because of the specific characteristics for idlescan or synscan: reflexive ports (this time 109, which is the port for pop2 (Post Office Protocol version 2)), SYN and FIN flags set, window size of 1028 and a IP ID of 39426. Other scans of this type were seen for ports 53 (domain) and 21 (ftp).

```
02/10-04:01:05.557978 194.222.79.210:30973  -> MY.NET.100.165:20
TCP TTL:114 TOS:0x0 ID:11869  DF
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
21*FRPAU Seq: 0x78FD0014   Ack: 0x78FD0014   Win: 0x14
TCP Options => EOL EOL EOL EOL EOL EOL

02/11-09:03:21.051452 194.222.196.210:30973  ->
MY.NET.100.165: 33164
TCP TTL:239 TOS:0x0 ID:29653  DF
21*FRPAU Seq: 0x78FD818C   Ack: 0x78FD818C   Win: 0x818C
TCP Options => EOL EOL EOL EOL EOL EOL SackOK

02/11-11:53:53.773946 194.222.70.195:30969   -> MY.NET.181.144:49612
TCP TTL:112 TOS:0x0 ID:51261  DF
21*F*PAU Seq:  0x78F9C1CC   Ack: 0x78F9C1CC   Win: 0xC1CC
78 F9 C1 CC 78 F9                                 x...x.

02/12-04:51:16.077132 194.222.96.40:30973  -> MY.NET.60.14:32788
TCP TTL:114 TOS:0x0 ID:9  DF
21*FRPAU Seq: 0x78FD8014   Ack: 0x78FD8014   Win: 0x8014
TCP Options => EOL EOL EOL EOL EOL EOL SackOK

02/12-04:57:50.433457 194.217.242.35:30975   -> MY.NET.253.24:20
TCP TTL:241 TOS:0x0 ID:44050  DF
21SFRPAU Seq: 0x78FF0014   Ack: 0x78FF0014   Win: 0x14
TCP Options => EOL EOL EOL EOL EOL EOL SackOK
```

These are all pa ckets coming from the DEMON -NET (194.222.0.0/16), a UK Provider has showed malfunctioning hardware in the past (see http://www.sans.org/y2k/020300.htm for more information).

```
01/31-00:54:42.967699 MY.NET. 227.26:6699 -> 24.91.117.197:2681
TCP TTL:126 TOS:0x0 ID:54136  DF
2*SFR*AU Seq: 0x382   Ack: 0x8E9DA5E4   Win: 0x5018
8E 9D A5 E4 2A 77 50 18 21 C7 B8 8F 00 00 C6 A2  ....*wP.!.......
74 DA F8 F6 82 CC DD 14 9C 31                    t........1

01/31-09:51:59.406019 213.89.132.17:6699  -> MY.NET.207.226:1194
TCP TTL:107 TOS:0x0 ID:11709  DF
21**R*AU Seq: 0x122   Ack: 0xFA52002A   Win: 0x8010
TCP Options => EOL EOL NOP NOP Sack: 42@60696 EOL EOL EOL EOL EOL
EOL EOL EOL EOL EOL EOL EOL EOL EOL EOL EOL EOL   EOL EOL EOL
EOL

01/31-15:54:14.018236 24.68.113.81:6699  -> MY.NET.225.250:1494
TCP TTL:52 TOS:0x0 ID:52675
*1SFR*** Seq: 0xE60089   Ack: 0x37C2009F   Win: 0x5018
TCP Options => EOL EOL

01/31-20:49:52.913913 MY.NET.226.38:0  -> 129.252.203.142:6699
TCP TTL:126 TOS:0x0 ID:49115  DF
*1SF**AU Seq: 0x8FB0246   Ack: 0xE2D0135A   Win: 0x5010
TCP Options => Opt 32 (32): 2020 2000 0402 0101 080A 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 EOL EOL EOL EOL EOL EOL
EOL EOL
```

Some strange flags and options. We sh owed these packets because they are all using port 6699, which is known to be used by Napster for client -to-client connections. Maybe napster was poorly written. We didn't see the complete packet content, so we are not sure whether this actually is Napster traffic (the first packet could be transferring a little piece of an MP3 file), but these packets are weird.

```
01/31-01:44:04.410214 206.65.191.129:46877   -> MY.NET.213.198:580
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
TCP TTL:50 TOS:0x0 ID:0  DF
21S***** Seq: 0x784E8CC4   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 24307409 0 EOL EOL EOL EOL

01/31-01:44:05.449244 206.65.191.129:47018  -> MY.NET.213.198:1723
TCP TTL:50 TOS:0x0 ID:0  DF
21S***** Seq: 0x78D0F9F2   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 24307513 0 EOL   EOL EOL EOL

01/31-01:44:06.799587 206.65.191.129:47169  -> MY.NET.213.198:114
TCP TTL:50 TOS:0x0 ID:0  DF
21S***** Seq: 0x7878125E   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 24307648 0 EOL EOL EOL EOL

01/31-01:44:06.799649 206.65.191.12 9:47170 -> MY.NET.213.198:1084
TCP TTL:50 TOS:0x0 ID:0  DF
21S***** Seq: 0x7860FB54   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 24307648 0 EOL EOL EOL EOL

...             (217 packets omitted)

01/31-01:52:57.273046 206.65.191.129:46247  -> MY.NET.2 13.198:416
TCP TTL:50 TOS:0x0 ID:0  DF
21S***** Seq: 0x9A81CEC5   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 24360697 0 EOL EOL EOL EOL
```

Looks like a port scan, but (s)he is a little creative by not scanning consecutive ports, but by—at least at first sight—going for random ports. The timings however make it definitely a scan (sometimes less than a millisecond between 2 packets).

Also IP ID equaling 0 is suspicious, certainly using it every time, although I've seen the same behavior from my RedHat linux at home (using kernel 2.4.4).

```
02/08-20:51:43.842140 MY.NET.203.102:51943  -> 209.225.8.42:22
TCP TTL:46 TOS:0x0 ID:40105
**SF*P*U Seq: 0x69BEC4F3   Ack: 0x0   Win: 0x400
TCP Options => WS: 5 Opt 156 SackOK Opt 9 (8): 0A3F 3F3F 3F00 EOL
EOL EOL EOL EOL

02/08-20:52:19.974601 MY.NET.203.102:39831  -> 209.225.8.42:22
TCP TTL:44 TOS:0x0 ID:55633
**SF*P*U Seq: 0x89635B4C   Ack: 0x0   Win: 0xC00
TCP Options => WS: 5 Opt 156 SackOK Opt 9 (8): 0A3F 3F3F 3F00 EOL
EOL EOL EOL EOL

02/08-20:52:39.907916 M Y.NET.203.102:36160  -> 209.225.8.42:22
TCP TTL:45 TOS:0x0 ID:21867
**SF*P*U Seq: 0x67001C43   Ack: 0x0   Win: 0x1000
TCP Options => WS: 5 Opt 156 SackOK Opt 9 (8): 0A3F 3F3F 3F00 EOL
EOL EOL EOL EOL

02/08-20:57:07.230824 MY.NET.203.102:43165  -> 209.225.8. 42:22
TCP TTL:44 TOS:0x0 ID:50226
**SF*P*U Seq: 0xEC156B52   Ack: 0x0   Win: 0xC00
TCP Options => WS: 5 Opt 156 SackOK Opt 9 (8): 0A3F 3F3F 3F00 EOL
EOL EOL EOL EOL
```

Some remarkable traffic originating from **MY.NET.203.102** : port 22 is used by SSH, so this might be SSH traffic. SSH can use the PSH flag, but we never saw it use the URG flag, but it might be possible. SYN and FIN together however is never possible. **You might have**

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

broken hardware or software. An other remark: the TTL field is not always the same , which indicates different routing for the different packets.

```
02/09-01:41:18.048351 MY.NET.98.130:1433  -> 192.168.1.1:53
TCP TTL:62 TOS:0x0 ID:0  DF
21S***** Seq: 0x91830383   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 18714695 0 EOL EOL   EOL EOL

02/09-01:54:37.142521 MY.NET.98.130:1434  -> 192.168.1.1:53
TCP TTL:62 TOS:0x0 ID:0  DF
21S***** Seq: 0xC5CE9693   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 18794594 0 EOL EOL EOL EOL

02/09-02:08:11.246724 MY.NET.98.130:1435  -> 192.168.1.1:53
TCP TTL:62 TOS:0x0 ID:0  DF
21S***** Seq: 0xF98E86A0   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 18876894 0 EOL EOL EOL EOL

02/09-02:21:39.367786 MY.NET.98.130:1436  -> 192.168.1.1:53
TCP TTL:62 TOS:0x0 ID:0  DF
21S***** Seq:  0x2BAFA398   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 18957694 0 EOL EOL EOL EOL
```

We are not sure whether "21S" is a valid flag -set for tcp-domain, but normally, when you see a SYN packet to initiate a connection, no other flags should be in use. Besides that, why are we seeing so much domain TCP traffic from **MY.NET.98.130** (we saw this about 70 times in 8 hours). This is only used for zone transfers. **You might have a DNS server going crazy or even broken hardware.** Worth investigating…

```
03/08-02:52:40.906296 MY.NET.205.206:6688  -> 129.170.107.104:2709
TCP TTL:126 TOS:0x0 ID:38565  DF
12*****F Seq: 0x92F5440   Ack: 0x7AD   Win: 0x5010
TCP Options => EOL
18 03 00 00 6C 6F 31 00 00 00 00 00 00 00 00 00  ....lo1.........

03/08-02:52:41.952842 MY.NET.205.206:6688  -> 129.170.107.104:2709
TCP TTL:126 TOS:0x0 ID:57254  DF
12*****F Seq: 0x934   Ack: 0x69F407AD   Win: 0x5018
TCP Options => EOL
18 03 00 00 6C 6F 31 00 00 00 00 00 00 00 00 00  ....lo1.........

03/08-08:43:49.846370 MY.NET.205.206:6688  -> 24.226.144.29:1044
TCP TTL:126 TOS:0x0 ID:46414  DF
1***PRSF Seq: 0xAA211AC   Ack: 0x546   Win: 0x5010
TCP Options => EOL
18 03 00 00 6C 6F 31 00 00 00 00 00 00 00 00 00  ....lo1.........

03/08-08:44:58.464011 MY.NET.205.206:6688  -> 24.226.144.29:1044
TCP TTL:126 TOS:0x0 ID:9304  DF
1***PRSF Seq: 0xAB247AC   Ack: 0x546   Win: 0x5010
TCP Options => EOL
18 03 00 00 6C 6F 31 00 00 00 00 00 00 00 00 00  ....lo1.........

03/08-10:53:34.405899 MY.NET.205.206:6688  -> 62.20.218.9:1190
TCP TTL:126 TOS:0x0 ID:17 612  DF
***A**SF Seq: 0xAF4   Ack: 0x6373005E   Win: 0x5010
TCP Options => EOL
18 03 00 00 6C 6F 31 00 00 00 00 00 00 00 00 00  ....lo1.........
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Not sure what **MY.NET.205.206** is trying to accomplish, but it **might be compromised** . It might also be broken har dware, but it sends always packets out on port 6688 and it flags settings are really weird. Maybe it's an unknown trojan…

Things used to make this third assignment

Because of the overwhelming amount of data to process, we thought it might be interesting to know which tools, etc… were used for this assignment.

Programs:

- perl scripting
- shell scripting
- grep
- *snort_sort.pl* by *Andrew R. Baker* as basis for some of my own scripts

Books:

- "TCP/IP Illustrated, Volume 1" by W. Richard Stevens (Addison -Wesley)
- Programming Perl, 2 nd Ed. by Larry Wall, Tom Christiansen & Randal L. Schwartz (O'Reilly)
- Perl Cookbook by Tom Christiansen & Nathan Torkington (O'Reilly)

Online Information:

- RFC database
- several Port databases (including the original from IANA)
- Vulnerability sites: cve.mitre.org, whitehats.com, securityfocus.com, simovits.com and others…
- Other sites: snort.org, sans.org, google.com and others…

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.