Global Information Assurance Certification Paper

Abstract:

This practical starts out with a discussion of Honeypots as IDS tool. Two types of honeypots, Production and Research are discussed. Benefits and negatives are also discussed. Next comes three network detects. The first is from the Raw logs located at Incidents.org, and the other two are from my company's network. Last is the third part of the Practical, where I analyzed five contiguous day's worth of traffic and commented on trends, and particular events. I gave some information on the traffic that was seen the most, as well as some extended information on particular hosts. Last was my defensive recommendations, and how I went about the process of sifting through the logs.

GCIA Practical Assignment
Version 3.3

Thomas R Karetas
Jan. 25, 2003

Part #1 Describe the State of Intrusion Detection

Deception Systems as an IDS tool

Introduction

Deception systems, also referred to as honeypots, are an emerging technology in the world of Information Security. They have broad implications, and can benefit many disciplines of the InfoSec community. However, the purpose of this paper is to highlight the use of a honeypot as an IDS tool. A honeypot is "a security resource whose value lies in being probed, attacked, or compromised," as defined by Lance Spitzner, a prime member of the Honeynet project (http://www.honeynet.org) and author of the book Honeypots: Tracking Hackers. To elaborate, a honeypot's value lies in that it has no other purpose on a network segment. Therefore any traffic directed at it is suspect.

Types of Honeypots

Just as there are many forms of firewalls, there are a number of honeypots available. They can be commercial products, freeware, even built from the ground up by the user. Honeypot solutions are generally evaluated on the basis of their level of interaction. Regardless of the level of interaction, all honeypots provide an additional level of benefit to a network's defenses.

Level of Interaction	Advantages	Disadvantages
Low	Easy to Install Aid in Detection Generally run as a single program Low Risk of compromise	Limited amount of data collected Can't identify new data Simple emulated services
Medium	Provides more data, including the capture of worm payloads Provides more complex emulated services Usually resides in a virtual OS, like that created by the Unix jail or chroot command	More room for error in configuration and setup Compromised hosts can be used as attack platform Time-consuming
High	Allows the for the greatest level of data collection and observation No emulated services Can more easily discover new vulnerabilities	Provides an actual OS if compromised Must be monitored constantly

Figure 1.1 Interactive Levels of Honeypots
Based on Information from Tracking Hackers

Here is a rundown of some available honeypot products:

BackOfficer Friendly http://www.nfr.com/products/bof

A low-interaction honeypot. Free and simple to use.

Specter http://www.specter.com

Another low-interaction honeypot. Adds more emulated services. Can also emulate operating systems.

Honeyd http://www.citi.umich.edu/u/provos/honeyd/

An Opensource low-interaction honeypot. Can monitor multiple IP addresses and emulate hundreds of operating systems at both the application and IP stack level.

```
honeyd[2054]: Sending echo reply: 10.21.19.242 -> 240.81.64.14
honeyd[2054]: Connection request: (231.205.161.9:64843 - 10.21.19.240:80)
honeyd[2054]: Connection established: (231.205.161.9:64843 - 10.21.19.240:80) <-> /var/honeyd/scripts/web.sh
honeyd[2054]: Connection dropped with reset: (231.205.161.9:64843 - 10.21.19.240:80)
honeyd[2054]: Connection request: (12.237.70.38:4064 - 10.21.19.240:80)
honeyd[2054]: Connection established: (12.237.70.38:4064 - 10.21.19.240:80) <-> /var/honeyd/scripts/web.sh
honeyd[2054]: Connection dropped with reset: (12.237.70.38:4064 - 10.21.19.240:80)
honeyd[2054]: Connection request: (10.21.24.100:31537 - 10.21.19.240:80)
honeyd[2054]: Connection established: (10.21.24.100:31537 - 10.21.19.240:80) <-> /var/honeyd/scripts/web.sh
honeyd[2054]: Expiring (10.21.24.100:31537 - 10.21.19.240:80) (0x55800) in state 7
honeyd[2054]: Connection request: (10.21.24.101:36539 - 10.21.19.245:80)
honeyd[2054]: Connection established: (10.21.24.101:36539 - 10.21.19.245:80) <-> /var/honeyd/scripts/web.sh
```

Figure 1.2 An example of logs from Honeyd
Taken from the Honeyd website

Mantrap http://www.recourse.com

A medium- to high-interaction honeypot. A commercial product that uses actual services instead of emulation.

Homemade

Can be low- to high-interaction. The most versatile of the honeypots, as well as the hardest to setup.

Honeynet

A high-interaction honeypot, composed of multiple honeypots. It can also often include the associated firewall, and network sensors.

In addition to the level of interaction, honeypots are generally grouped into two broad categories, production and research. Production honeypots are usually either low to medium interaction. A discussion of production honeypots and how they can be used to monitor both internal and DMZ segments will come first. This will be followed by a discussion of how a proactive analyst can use a research honeypot to detect and analyze previously unidentified traffic. Research honeypots are often high-interaction, but are sometimes medium-interaction as well.

Production Systems

As stated previously a honeypot's value lies in that it does not normally generate traffic, either to or from the machine. A security professional can place a honeypot on a segment and know that any traffic to or from the machine is illegitimate, aside from the occasional mishap, such as a mistyped IP address. So then, a production honeypot is used to detect traffic on a segment, much like other IDS tools. However, due to its nature, it greatly reduces the number of false positives that occur. In addition, it also helps to alleviate the number of false negatives, which are often very hard to detect. There are two prime placements points for a honeypot.

The first of these is on an internal segment. A honeypot on an internal segment helps in many ways. It can help to monitor internal traffic, such as point out worms scanning local segments for new hosts. It can help to detect malicious users attempting to misuse company resources, as well as detect malicious from alternate access points, such as dial-ups and VPN connections. Lastly on an internal segment, it can help to point out holes in a firewall rule set, by helping to show how a non-specific computer on the network can be accessed.

On a DMZ or service segment, a honeypot holds much the same function. However a honeypot on a DMZ segment can also be setup so as to look like another production system, which allows the honeypot to be used to detect more directed attacks, that target specific services, such as attempted web page defacement for web servers, or attempted database access for a database server.

The prime disadvantage of a honeypot is that it can only detect traffic directed at it. One would still need the use of a NIDS device to monitor all traffic on a network segment. However, the combined use of a honeypot and a NIDS device to use for event correlation can be a great benefit to an intrusion analyst. The honeypot used in conjunction with a NIDS device, can help analyst more easily determine false positives, as well as point out the harder to detect, false negatives. This can also help to streamline a NIDS device's rule set.

An example of this could occur as follows. As the Internet has developed, many alternatives to HTML have arisen. One of these is Cold Fusion, a relatively simple and easy to use dynamic web page server. The Cold Fusion server utilizes an administration web page that allows the administrator to change settings for the server. A Snort signature has been developed in order to detect accesses to the page so as to monitor illegitimate accesses. However in a busy environment it can often be hard to differentiate between legitimate and illegitimate traffic. Looking at the log provided below, you can see 5 accesses to the Cold Fusion admin page. The accesses come from an off-site, but affiliated machine. So it would appear that it is all legitimate traffic. But suppose that 123.456.4.42 (Figure 1.3 – Detect #4) is actually a honeypot. Then the traffic becomes suspect. Instead of an authorized user, we may actually be dealing with

a curious employee probing around the network. The honeypot has alerted us to traffic we would otherwise consider legitimate.

Detect #1
[**] [1:908:5] WEB-COLDFUSION administrator access [**]
[Classification: Attempted Information Leak] [Priority: 2]
12/23-17:00:46.970867 111.222.333.444:1929 -> 123.456.100.101:80
TCP TTL:117 TOS:0x0 ID:48263 IpLen:20 DgmLen:379 DF
AP Seq: 0x571EA45 Ack: 0x2051FAFD Win: 0x40B0 TcpLen: 20
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0538]

GET /cfide/administrator/index.cfm HTTP/1.1..Accept: image/gif,
image/x-xbitmap, image/jpeg, image/pjpeg, application/vnd.ms-exc
el, application/msword, */*..Accept-Language: en-us..Accept-Enco
ding: gzip, deflate..User-Agent: Mozilla/4.0 (compatible; MSIE 6
.0; Windows NT 5.0; YComp 5.0.2.6)..Host: 112.223.334.445..Connect
ion: Keep-Alive....

Detect #2
[**] [1:908:5] WEB-COLDFUSION administrator access [**]
[Classification: Attempted Information Leak] [Priority: 2]
12/23-17:01:44.666654 111.222.333.444:1936 -> 123.456.100.102:80
TCP TTL:117 TOS:0x0 ID:15547 IpLen:20 DgmLen:379 DF
AP Seq: 0x84C31C6 Ack: 0x21338F65 Win: 0x40B0 TcpLen: 20
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0538]

GET /cfide/administrator/index.cfm HTTP/1.1..Accept: image/gif,
image/x-xbitmap, image/jpeg, image/pjpeg, application/vnd.ms-exc
el, application/msword, */*..Accept-Language: en-us..Accept-Enco
ding: gzip, deflate..User-Agent: Mozilla/4.0 (compatible; MSIE 6
.0; Windows NT 5.0; YComp 5.0.2.6)..Host: 112.223.334.445..Connect
ion: Keep-Alive....

Detect #3
[**] [1:908:5] WEB-COLDFUSION administrator access [**]
[Classification: Attempted Information Leak] [Priority: 2]
12/23-17:00:52.002852 111.222.333.444:1933 -> 123.456.100.127:80
TCP TTL:117 TOS:0x0 ID:14182 IpLen:20 DgmLen:379 DF
AP Seq: 0x14A65F7B Ack: 0x2065D172 Win: 0x40B0 TcpLen: 20
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0538]

GET /cfide/administrator/index.cfm HTTP/1.1..Accept: image/gif,
image/x-xbitmap, image/jpeg, image/pjpeg, application/vnd.ms-exc
el, application/msword, */*..Accept-Language: en-us..Accept-Enco
ding: gzip, deflate..User-Agent: Mozilla/4.0 (compatible; MSIE 6
.0; Windows NT 5.0; YComp 5.0.2.6)..Host: 112.223.334.445..Connect
ion: Keep-Alive....

Detect #4
[**] [1:908:5] WEB-COLDFUSION administrator access [**]
[Classification: Attempted Information Leak] [Priority: 2]
12/23-17:01:37.510442 111.222.333.444:1935 -> 123.456.4.42:80
TCP TTL:116 TOS:0x0 ID:17320 IpLen:20 DgmLen:379 DF
AP Seq: 0xEB04D061 Ack: 0x8D250B6C Win: 0x40B0 TcpLen: 20
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0538]

GET /cfide/administrator/index.cfm HTTP/1.1..Accept: image/gif,
image/x-xbitmap, image/jpeg, image/pjpeg, application/vnd.ms-exc
el, application/msword, */*..Accept-Language: en-us..Accept-Enco
ding: gzip, deflate..User-Agent: Mozilla/4.0 (compatible; MSIE 6
.0; Windows NT 5.0; YComp 5.0.2.6)..Host: 112.223.334.445..Connect
ion: Keep-Alive....

Detect #5
[**] [1:908:5] WEB-COLDFUSION administrator access [**]
[Classification: Attempted Information Leak] [Priority: 2]
12/23-17:01:53.957687 111.222.333.444:1938 -> 123.456.4.43:80
TCP TTL:116 TOS:0x0 ID:56691 IpLen:20 DgmLen:379 DF
AP Seq: 0x2A06D200 Ack: 0x92F966D4 Win: 0x40B0 TcpLen: 20
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0538]

GET /cfide/administrator/index.cfm HTTP/1.1..Accept: image/gif,
image/x-xbitmap, image/jpeg, image/pjpeg, application/vnd.ms-exc
el, application/msword, */*..Accept-Language: en-us..Accept-Enco
ding: gzip, deflate..User-Agent: Mozilla/4.0 (compatible; MSIE 6
.0; Windows NT 5.0; YComp 5.0.2.6)..Host: 112.223.334.445..Connect
ion: Keep-Alive....

Figure 1.3 Snort logs of a series of Cold Fusion admin accesses

Research Honeypots

A research honeypot has the same basic value as a production honeypot. However, with an enhanced level of interaction, a research honeypot is able to gain more data about an attack's methods. This can include capturing payloads of worms, and monitoring the text sent by an attack. This added level of information could allow an analyst to create new signatures and detection methods for previously unidentified attacks. In an industry where time is such a valuable resource, this benefit can help to keep an analyst on top of things, and better able to detect future attacks.

Research honeypots also have the same basic disadvantage of a production honeypot. However it has an additional disadvantage due to its higher level of interaction. This higher level of interaction creates a higher level of risk, that isn't as prevalent in a production honeypot, although it does still exist. This risk is that of being compromised and used as a platform for future attacks. This means to be a proper tool, a research honeypot must be vigilantly monitored. No honeypot is a "fire and forget" solution.

Conclusion

Deception systems, when implemented correctly can be a valuable tool to an intrusion analyst, in a multitude of ways. They are high maintenance tools, but the benefits of having a relatively false positive free IDS are a boon that cannot be provided in any other manner. Not to mention, there is almost no other type of system that helps to eliminate false negatives. But it must be remembered, that they are not a fix, nor are they a complete solution to intrusion detection alone. They work best when implemented in conjunction with other IDS tools.

It must be stressed that a honeypot is a time-consuming strategy. They must be constantly monitored in order to prevent their misuse. In the case of honeypots that use emulation of services, or operating systems, one must take care in order to prevent the honeypot from being discovered for what it is. A honeypot that is actively avoided does no good.

Also a few legal implications must be relayed. Since the average user of a honeypot does not represent a law enforcement agency, there is no real concern of entrapment. However, if a malicious user is able to take control of the honeypot and use it for nefarious deeds, you may be considered liable for the resultant damages. Once again, it must be stressed that the honeypot must be constantly monitored. Disclaimer: I am not a lawyer, nor am I versed in computer or security law. This is just information that is gleaned from discussions of honeypots across the web.

References

1. Cheswick, Bill. An Evening with Berferd in Which a Cracker is Lured, Endured, and Studied. 1991.
URL: http://www.tracking-hackers.com/papers/berferd.pdf.

2. Honeynet Project. Know Your Enemy: Motives, Worms at War. 2000.
URL: http://www.honeynet.org/papers/.

3. Honeypots: Monitoring and Forensics. 2002.
URL: http://honeypots.sourceforge.net/main.html.

4. Sink, Micheal. The Use of Honeypots and Packet Sniffers for Intrusion Detection. 2001.
URL: http://rr.sans.org/intrusion/honey_pack.php.

5. Spitzner, Lance. Honeypots: Tracking Hackers. 2003.
New York, New York: Addison-Wesley.

6. Spitzner, Lance. Know Your Enemy: I. 2000.
URL: http://www.honeynet.org/papers/.

Part #2 Network Detects

Trace #1

```
11/17-19:12:32.476507 202.108.254.204:14955 -> 111.222.149.62:8080
TCP TTL:46 TOS:0x0 ID:46493 IpLen:20 DgmLen:40
******S* Seq: 0x1B74C762  Ack: 0x1B74C762  Win: 0x400  TcpLen: 20

11/17-19:23:01.616507 202.108.254.204:39625 -> 111.222.149.62:3128
TCP TTL:46 TOS:0x0 ID:51319 IpLen:20 DgmLen:40
******S* Seq: 0x184581E9  Ack: 0x184581E9  Win: 0x400  TcpLen: 20

11/17-19:43:59.236507 202.108.254.204:53469 -> 111.222.149.62:1080
TCP TTL:46 TOS:0x0 ID:52921 IpLen:20 DgmLen:40
******S* Seq: 0x6DEB8587  Ack: 0x6DEB8587  Win: 0x400  TcpLen: 20

11/17-20:04:57.396507 202.108.254.204:47754 -> 111.222.215.53:8080
TCP TTL:46 TOS:0x0 ID:10654 IpLen:20 DgmLen:40
******S* Seq: 0x26B8ECC6  Ack: 0x26B8ECC6  Win: 0x400  TcpLen: 20

11/17-20:15:26.096507 202.108.254.204:22593 -> 111.222.215.53:3128
TCP TTL:46 TOS:0x0 ID:10731 IpLen:20 DgmLen:40
******S* Seq: 0x95CDA7B  Ack: 0x95CDA7B  Win: 0x400  TcpLen: 20

11/17-20:36:23.816507 202.108.254.204:2897 -> 111.222.215.53:1080
TCP TTL:46 TOS:0x0 ID:29679 IpLen:20 DgmLen:40
******S* Seq: 0x4749C18C  Ack: 0x4749C18C  Win: 0x400  TcpLen: 20

11/17-20:57:21.926507 202.108.254.204:2995 -> 111.222.252.40:8080
TCP TTL:46 TOS:0x0 ID:27732 IpLen:20 DgmLen:40
******S* Seq: 0x2DC8312A  Ack: 0x2DC8312A  Win: 0x400  TcpLen: 20

11/17-21:07:50.556507 202.108.254.204:691 -> 111.222.252.40:3128
TCP TTL:46 TOS:0x0 ID:21670 IpLen:20 DgmLen:40
******S* Seq: 0x5E0598A2  Ack: 0x5E0598A2  Win: 0x400  TcpLen: 20

11/17-21:28:48.676507 202.108.254.204:14924 -> 111.222.252.40:1080
TCP TTL:46 TOS:0x0 ID:25248 IpLen:20 DgmLen:40
******S* Seq: 0x277434C2  Ack: 0x277434C2  Win: 0x400  TcpLen: 20

11/17-21:49:46.296507 202.108.254.204:30295 -> 111.222.212.139:8080
TCP TTL:45 TOS:0x0 ID:37393 IpLen:20 DgmLen:40
******S* Seq: 0x1CB2533D  Ack: 0x1CB2533D  Win: 0x400  TcpLen: 20

11/17-22:00:15.076507 202.108.254.204:55675 -> 111.222.212.139:3128
TCP TTL:45 TOS:0x0 ID:30175 IpLen:20 DgmLen:40
******S* Seq: 0x4B7D6B6  Ack: 0x4B7D6B6  Win: 0x400  TcpLen: 20

11/17-22:21:13.116507 202.108.254.204:53269 -> 111.222.212.139:1080
TCP TTL:45 TOS:0x0 ID:52742 IpLen:20 DgmLen:40
******S* Seq: 0x6019CE4A  Ack: 0x6019CE4A  Win: 0x400  TcpLen: 20
```

1. Source of Trace:

This trace comes from the incidents.org raw logs sets. It is located at http://www.incidents.org/logs/Raw/2002.10.18. The network layout of this detect is unknown and not easily discernable from the detect. However it appears that the targeted machines are non-specific internal hosts.

2. Detect was generated by:

This detect was generated by snort with a fairly standard rule set. In order to examine and work with the log, it was viewed with ethereal and snort. The final output included here is from snort.

The log format is pretty standard. On the first line is the timestamp followed the source address and port, then followed by the destination address and port. The second line tells the embedded protocol, the time-to-live value,the Type of Service, the IP ID, the length of the IP header, and lastly, the length of the entire datagram. The last line gives the protocol specific data, in this case TCP. The first set of information on this line is the flags that are set, followed by the TCP sequence number, and the acknowledgement number, TCP Window Size, and lastly the TCP Header length.

Since this is a raw tcpdump log, all network traffic has been captured.

3. Probability the source address was spoofed:

It is unlikely that this source address is spoofed. Since this is a reconnaissance scan for a TCP service, it is necessary to receive a response in order to determine the presence of the service being scanned for. However, this is very likely some sort of tool, that is being used to perform this scan. With the low TTL value, the time of the scans being distributed, and the seq/ack number weirdness, it is without a doubt a scanning tool, although the exact tool cannot be determined.

4. Description of attack:

This attack is a scan across multiple hosts on the ports 1080(socks), 3128(squid-http), and 8080(HTTP Alternate). There are a few possibilities of the what the attacker could have been looking for Trojans commonly configured to run on these ports. These Trojans include Winhole(1080), SubSeven 2.2(1080), RingZero(3128,8080), Brown Orifice(8080), and RemoConChubo(8080). These Trojans here are taken from the ports list compile by Neohapsis, available at http://www.neohapsis.com/neolabs/neo-ports/neo-ports.html. However the more probable answer is that the attacker was simply scanning for open proxies in which to exploit.

5. Attack Mechanism

Since this is a scan, it simply works by sending a request connection and then taking note if a reply is received. Out of the two option, the first option being the attacker is looking for default configurations of various Trojans, the first is the least likely. This would require that (1) the host had been previously

compromised, and (2) that the default password was still in use on the compromised machine. While possible, this is an unlikely event.

The more likely thing going on, is that the attacker is looking for open proxies from which they can launch attacks against other machines from a state of relative anonymity.

6. Correlations:

Checking Dshield.org against a list of known scanners, the source address comes up with a history of 6882 known scans against different IP addresses. This information is available at http://www.dshield.org/ipinfo.php?ip=202.108.254.204.

7. Evidence of Active Targeting:

This traffic does not appear to be specifically targeting any machine. There is no pattern to the hosts being scanned, and the IP addresses seem to have been chose quite randomly. Combined with the record located at Dshield, I would confidently say that there was no active targeting.

8. Severity

Severity = (Criticality + Lethality) –
(System countermeasures + Network countermeasures)

Category	Rating	Reason
Criticality	**1**	These hosts just seem to be internal machines in the network and don't appear to serve any production purpose.
Lethality	**2**	This is a type of port scan, however if any of the scanned machines had been an open proxy, the machines could have been used as a base for further attacks.
System Countermeasures	**2**	Although detailed information on these hosts are unknown, the absence of further traffic from the scanned hosts would seem to indicate that these machine were not running the services being scanned for.
Network Countermeasures	**1**	No accurate measure of the network countermeasures can be achieved; therefore it is being assumed to be the lowest value, or a system with no network countermeasures.

The Severity is valued at **0.**

0 = (1 + 2) – (2 + 1)

This is achieved by using the values above in the prescribed formula.

9. Defensive Recommendation:

Also in accordance with the fine details of the network layout being unknown, there are a few simple recommendations.

Using the assumption that the hosts scanned were internal hosts, and that there are no network countermeasures present, I would recommend a firewall and rules that would block incoming traffic to the internal machines of the network, including the ports scanned for in this detect.

10. Multiple Choice Test Question:

Question: Which of the following is not a common Trojan running on either port 1080, 3128, or 8080?

A. RingZero
B. BrownOrifice
C. Squid-http
D. RemoConChubo

Answer: C, Squid-http is a legitimate web proxy service

***Question(s) from the Intrusions List:**

Only 1 question was asked about this detect.

1. Question: The Sequence and Acknowledgement values for each packet are the same. Is this a signature for some kind of tool?

   Answer: This is definitely the sign of a crafted packet. Such a thing could be done with libnet, however I don't believe that it is a specific tool that is creating this packet.

Trace #2

```
[**] [111:12:1] spp_stream4: NMAP FINGERPRINT (stateful) detection [**]
12/21-17:35:03.975877 80.197.33.62:63002 -> 111.222.333.444:80
TCP TTL:33 TOS:0x0 ID:45686 IpLen:20 DgmLen:60
***A**** Seq: 0x4C869DA8  Ack: 0x0  Win: 0xC00  TcpLen: 40
TCP Options (5) => WS: 10 NOP MSS: 265 TS: 1061109567 0 EOL

[**] [111:9:1] spp_stream4: STEALTH ACTIVITY (NULL scan) detection [**]
12/21-17:35:10.230534 80.197.33.62:63000 -> 111.222.333.444:80
TCP TTL:33 TOS:0x0 ID:39952 IpLen:20 DgmLen:60
******** Seq: 0x4C869DA8  Ack: 0x0  Win: 0xC00  TcpLen: 40
TCP Options (5) => WS: 10 NOP MSS: 265 TS: 1061109567 0 EOL

Dec 21 17:20:32.491: %SEC-6-IPACCESSLOGP: list com_dmz_incoming_access denied tcp
80.197.33.62(63003) -> 111.222.333.444(36321), 1 packet
*Repeats 2 more times*

Dec 21 17:21:04.455: %SEC-6-IPACCESSLOGP: list com_dmz_incoming_access denied udp
80.197.33.62(62992) -> 111.222.333.444(36321), 1 packet
```

1. Source of Trace:

This trace comes from a DMZ segment at the company I work for. It uses a Cisco router to control access to the segment as well as a Snort IDS sensor on the inside of the router to monitor traffic that makes it past the router.

2. Detect was generated by:

This detect was generated by a combination of Snort and a Cisco router. The first two events were picked up by Snort, and the last four events were blocked by the ACL rules of the router.

The Snort log is the same as previously explained, however an additional line of TCP Options is included. The Cisco log is also fairly simple. First is the timestamp, followed by the type of log message, then the access list which generated the message. After that comes that action taken against the packet and the protocol. Lastly is the source address and port, the destination address and port, and the number of packets acted on.

3. Probability the source address was spoofed:

It is unlikely that the source address was spoofed. This type of attack is informational in purpose and therefore needs to return back to the user. Often others of the same type accompany this type of attack from different addresses used as decoys. However since this event occurred alone it is unlikely that it is a decoy.

4. Description of attack:

This attack is an informational attack using the tool nmap(http://www.insecure.org/nmap/) in order to determine the operating system of the targeted computer.

5. Attack Mechanism

This attack works by sending six packets to the targeted machine. It sends two packets to a known open port and 4 packets to an assumed closed port. Then using a database of known operating systems, the reactions of the connections are compared to the database in order to determine the operating system of the targeted machine. The assumed closed port is chosen randomly by nmap from among the higher ephemeral ports.

The actual format of the nmap command used to perform this attack would have been similar to

nmap –v –sT –p 80 –O 111.222.333.444

6. Correlations:

Using ntool, a web front-end to nmap, located at http://www.false.net/ntool/n.p to scan the targeted machine, caused the same events to recur, albeit with a different random high port.

```
[**] [111:12:1] spp_stream4: NMAP FINGERPRINT (stateful) detection [**]
01/01-01:59:52.124741 209.207.210.180:36652 -> 111.222.333.444:80
TCP TTL:31 TOS:0x0 ID:21319 IpLen:20 DgmLen:60 DF
***A**** Seq: 0x9201259D  Ack: 0x0  Win: 0x400  TcpLen: 40
TCP Options (5) => WS: 10 NOP MSS: 265 TS: 1061109567 0 EOL

[**] [111:9:1] spp_stream4: STEALTH ACTIVITY (NULL scan) detection [**]
01/01-01:59:54.101398 209.207.210.180:36650 -> 111.222.333.444:80
TCP TTL:31 TOS:0x0 ID:21325 IpLen:20 DgmLen:60 DF
******** Seq: 0x9201259D  Ack: 0x0  Win: 0x400  TcpLen: 40
TCP Options (5) => WS: 10 NOP MSS: 265 TS: 1061109567 0 EOL

Jan  1 01:44:42.277: %SEC-6-IPACCESSLOGP: list com_dmz_incoming_access denied tcp
209.207.210.180(36653) -> 111.222.333.444(43100), 1 packet
*Repeats 2 more times*
Jan  1 01:45:02.729: %SEC-6-IPACCESSLOGP: list com_dmz_incoming_access denied udp
209.207.210.180(36642) -> 111.222.333.444(43100), 1 packet
```

7. Evidence of Active Targeting:

As this was the only host scanned and this host does serve web pages to the public, it is likely that the machine was actively targeted by the attacker in order to gather more information about it.

8. Severity

Severity = (Criticality + Lethality) –
(System countermeasures + Network countermeasures)

Category	Rating	Reason
Criticality	**3**	This is a production web server, although not as used as some of the other web servers.
Lethality	**1**	This attack was purely informational. No actual exploit was attempted against the machine.
System Countermeasures	**3**	This system is patched and up to date for currently known exploits.
Network Countermeasures	**3**	The router ACLs block all but the necessary traffic to the machine

The Severity is valued at **-2.**

-2 = (3 + 1) – (3 + 3)

This is achieved by using the values above in the prescribed formula.

9. Defensive Recommendation:

Technically the attack was successful. The attacker would have been able to gather the information on the operating system of the machine from the port 80 probe. However as stated all unneeded access is blocked and services are patched, so the attacker would have a hard time further exploiting this machine. The defenses are fine.

10. Multiple Choice Test Question:

Question: What port(s) does Nmap use in order to perform OS fingerprinting?

A. 80
B. 36321
C. All of the above
D. None of the above

Answer: D, None of the above. Nmap allows the user to specifiy, but it requires one open port, and one closed port to correctly identify an operating system.

Trace #3

```
Jan 21 07:28:51 firewall kernel: DROP:  IN=eth0 OUT=eth0 SRC=61.99.251.176
DST=111.222.232.214 LEN=60 TOS=0x00 PREC=0x00 TTL=44 ID=44359 DF PROTO=TCP SPT=1346
DPT=111 WINDOW=32120 RES=0x00 SYN URGP=0 OPT (020405B40402080A0116031B0000000001030300)
Jan 21 07:28:51 firewall kernel: DROP:  IN=eth0 OUT=eth0 SRC=61.99.251.176
DST=111.222.232.215 LEN=60 TOS=0x00 PREC=0x00 TTL=44 ID=44360 DF PROTO=TCP SPT=1347
DPT=111 WINDOW=32120 RES=0x00 SYN URGP=0 OPT (020405B40402080A0116031B0000000001030300)
Jan 21 07:28:51 firewall kernel: DROP:  IN=eth0 OUT=eth0 SRC=61.99.251.176
DST=111.222.232.216 LEN=60 TOS=0x00 PREC=0x00 TTL=44 ID=44361 DF PROTO=TCP SPT=1348
DPT=111 WINDOW=32120 RES=0x00 SYN URGP=0 OPT (020405B40402080A0116031B0000000001030300)
Jan 21 07:28:51 firewall kernel: DROP:  IN=eth0 OUT=eth0 SRC=61.99.251.176
DST=111.222.232.217 LEN=60 TOS=0x00 PREC=0x00 TTL=44 ID=44362 DF PROTO=TCP SPT=1349
DPT=111 WINDOW=32120 RES=0x00 SYN URGP=0 OPT (020405B40402080A0116031B0000000001030300)
Jan 21 07:28:51 firewall kernel: DROP:  IN=eth0 OUT=eth0 SRC=61.99.251.176
DST=111.222.232.218 LEN=60 TOS=0x00 PREC=0x00 TTL=44 ID=44363 DF PROTO=TCP SPT=1350
DPT=111 WINDOW=32120 RES=0x00 SYN URGP=0 OPT (020405B40402080A0116031B0000000001030300)
```

1. Source of Trace:

This trace comes from the network at my company. The log comes from a firewall that is protecting the internal network.

2. Detect was generated by:

This detect was generated by an iptables firewall. The iptables log format is fairly verbose.

This first portion is the syslog prefix, followed by the action taken. Next comes the IN and OUT interfaces, followed by the Source and Destination ports. After that is the length of the packet in bytes, the Type of Service “Type” field, and the Type of Service “Precedence” field. Next is the Time to Live value, and the packet id, followed by fragment information. After that is the IP options, and then the protocol. For the TCP protocol, this data is followed by the source and destination ports. Next comes the TCP Receive Window Size, followed by the TCP Flags, and lastly comes the TCP Options. This information was gleaned from http://logi.cc/linux/netfilter-log-format.php3.

3. Probability the source address was spoofed:

It is unlikely the source address is spoofed. The attacker would want a result back from this attack. However it is clearly the work of some tool. As the scan increments across IPs, other values such as the Packet ID and the source port do as well, each time incrementing by one.

4. Description of Attack:

This is a scan for RPC(Remote Procedure Call) services. It is simply sending out SYN packets, in the hopes of finding a host running the services.

5. Attack Mechanism:

This is most likely an automated rootkit. If an open port were found, it would attempt to exploit multiple vulnerabilities. One such vulnerability is the "IRIX ToolTalk RPC server Format String Vulnerability" updated on 01/21/2003 at http://online.securityfocus.com/advisory/4900. RPC services have a long standing history of being exploited.

6. Correlations:

While the exact nature of this attack cannot be determined, scans against RPC are by no means rare. Searching the vulnerabilities area at security focus turns up 65 known vulnerabilities for various RPC versions.

7. Evidence of Active Targeting:

This scan seems to be localized. However it does not seem to be targeting a specific machine. This host has no record on Dshield, and a cursory web search finds no mention of the IP.

8. Severity:

Severity = (Criticality + Lethality) –
(System countermeasures + Network countermeasures)

Category	Rating	Reason
Criticality	**1**	The machines targeted were inside hosts of no particular importance to business.
Lethality	**4**	If the attack had been successful, it most likely would have resulted in a compromised host.
System Countermeasures	**5**	The systems targeted do not run RPC services.
Network Countermeasures	**5**	The firewall blocks port 111 traffic from entering the network.

The Severity is valued at **-5.**

-5 = (1 + 4) – (5 + 5)

This is achieved by using the values above in the prescribed formula.

9. Defensive Recommendations:

The defenses are fine, the attack was blocked by the firewall.

10. Multiple Choice test Question:

Question: What port does the RPC portmap service run on?

A. 110
B. 111
C. 119
D. 120

Answer: B, RPC portmap runs on port 111, 110 is pop, 119 is nntp, and 120 doesn't have a standard service associated with it.

Part #3 Analyze This!

Executive Summary

First off, it needs to be mentioned, that searching through the logs of GIAC University is very time intensive, and requires a lot of creative thought. The sheer amount of data presented is an obstacle to all but the hardiest of people.

It is my opinion, that overall security at GIAC University is ample. Most of the traffic seen flowing through the gateways are benign in nature. Others are more neutral, things such as file-sharing, and IRC. And then there is the truly malicious traffic.

As with any University network, some amount of malicious traffic is going to be seen. In addition because of the nature of student networks, situations are going to occur. However it seems that GIAC University does a good job of keeping this to a minimum.

Overall, the traffic picked up in these logs is not malicious in nature, but there are a few incidents that require further looking in to.

File List

The set of files that I chose for this section were:

Alerts:
alert.021121.gz
alert.021122.gz
alert.021123.gz
alert.021124.gz
alert.021125.gz

Scans:
scans.021121.gz
scans.021122.gz
scans.021123gz
scans.021124gz
scans.021125gz

OOS:
OOS_Report_2002_11_21_6422.gz
OOS_Report_2002_11_22_23416.gz
OOS_Report_2002_11_23_7049.gz
OOS_Report_2002_11_24_31624.gz
OOS_Report_2002_11_25_28174.gz

Host Profile

The data was sorted through and mined, in order to try and pull out some machines that serve prominent services. Since this is just alert traffic, and not full traffic, we are only seeing a small amount of the actual services.

Port 21(FTP):
MY.NET.100.158
MY.NET.104.104
MY.NET.114.116

Port 23(Telnet):
MY.NET.168.150

Port 25(SMTP):
MY.NET.6.40

Port 80(HTTP):
MY.NET.70.231
MY.NET.99.174
MY.NET.162.87
MY.NET.179.77

Port 110(POP):
MY.NET.25.21

This data is just approximated based on the number of accesses to the specified port on the host.

Detects

Detects were ordered by the number of occurrences. I went in-depth on all events that had more than 10000 occurrences.

Number of Occurrences	Type of Event
62353	spp http decode: IIS Unicode attack detected
57259	SMB Name Wildcard
22435	TFTP – External UDP connection to internal tftp server
17778	Tiny Fragments – Possible Hostile Activity
13606	Watchlist 000220 IL – ISDNNET 990517
12332	spp http decode: CGI Null Byte attack detected
7506	High port 655535 udp – possible Red Worm traffic
6010	FTP DoS ftpd globbinh
5023	Queso fingerprint
3761	Incomplete Packet Fragments Discarded

1199	SUNRPC highport access
1151	Watchlist 000222 NET NCFC
1049	Null scan!
489	Port 55850 tcp – Possible myserver activity – ref. 010313-1
371	EXPLOIT x86 NOOP
263	connect to 515 from outside
180	Possible trojan server activity
172	TCP SRC and DST outside network
115	Highport 65535 tcp – possible Red Worm – traffic
108	Port 55850 udp – Possible myserver activity – ref. 010313-1
72	TFTP – Internal UDP connection to external tftp server
53	EXPLOIT x86 setuid 0
43	NMAP TCP ping!
28	EXPLOIT x86 setgid 0
22	External RPC call
16	EXPLOIT x86 stealth noop
16	EXPLOIT NTPDX buffer overflow
14	Attempted Sun RPC high port acess
8	connect to 515 from inside
6	RFB – Possible WinVNC – 010708-1
5	TFTP – Internal TCP connection to external tftp server
5	Back Orifice
4	TFTP – External TCP connection to internal
4	HelpDesk MY.NET.70.50 to external FTP
4	External FTP to HelpDesk MY.NET.70.50
4	External FTP to HelpDesk MY.NET.70.49
3	SYN-FIN scan!
2	Samba client access
2	Probable NMAP fingerprint attempt
2	NIMDA – Attempt to execute cmd from campus host
1	MY.NET.30.3 activity
1	Bugbear@MM virus in SMTP

Figure 3.1 Detects by Number

Detect #1 spp_http_decode: IIS Unicode attack detected

Number of Occurrences: 62353

Description of Vulnerability:

Through the use of Unicode character representations, a vulnerable IIS or Microsoft Personal Web Server can be used to retrieve unauthorized files from the machine hosting the server. This is done by using the Unicode representation for "../" or other directory traversal shortcuts.[1] This is the vulnerability that the Nimda, CodeRed, and CodeRed2 worms are based off of.

Logs of Detect:

```
11/21-00:00:41.164572  [**] spp_http_decode: IIS Unicode attack detected [**]
MY.NET.85.74:3554 -> 207.200.86.66:80
```

```
11/21-00:00:41.896038  [**] spp_http_decode: IIS Unicode attack detected [**]
MY.NET.85.74:3554 -> 207.200.86.66:80
11/21-00:00:47.622276  [**] spp_http_decode: IIS Unicode attack detected [**]
MY.NET.85.74:3561 -> 207.200.86.66:80
11/21-07:14:09.502601  [**] spp_http_decode: IIS Unicode attack detected [**]
MY.NET.183.59:1700 -> 64.12.180.19:80
11/21-07:14:09.502601  [**] spp_http_decode: IIS Unicode attack detected [**]
MY.NET.183.59:1700 -> 64.12.180.19:80
11/21-07:14:09.644622  [**] spp_http_decode: IIS Unicode attack detected [**]
MY.NET.183.59:1701 -> 64.12.42.116:80
11/21-07:14:09.644622  [**] spp_http_decode: IIS Unicode attack detected [**]
MY.NET.183.59:1701 -> 64.12.42.116:80
11/21-08:33:17.203745  [**] spp_http_decode: IIS Unicode attack detected [**]
MY.NET.153.163:1726 -> 211.32.117.31:80
11/21-08:33:17.203745  [**] spp_http_decode: IIS Unicode attack detected [**]
MY.NET.153.163:1726 -> 211.32.117.31:80
11/21-08:33:17.203745  [**] spp_http_decode: IIS Unicode attack detected [**]
MY.NET.153.163:1726 -> 211.32.117.31:80
```

Summary:

Over the five days of logs, there were 62353 IIS Unicode "attacks" detected. Over 50000 of these events originated from within MY.NET. Of those events the majority of them are to sites in the Asian region. Coming second are accesses to AOL, and third is Netscape. Doing some research, I have discovered where others have seen this traffic both to AOL[2] and Netscape[3]. This activity to international sites was also noted by Steven Drew in his practical[4], along with a reference to snort discussion lists on the topic[5]. So what we are seeing is a lot of false positives. Delving a little further and correlating the alert data against the scans, we do see that some of the internal hosts that have been alerted on Unicode attacks, have also been caught performing scans to port 80 on external machines. The machines involved are listed below:

MY.NET.53.30
MY.NET.53.41
MY.NET.53.42
MY.NET.53.44
MY.NET.53.52
MY.NET.53.55
MY.NET.53.60
MY.NET.53.160
MY.NET.87.123
MY.NET.88.169
MY.NET.99.203
MY.NET.152.157
MY.NET.153.110
MY.NET.153.111
MY.NET.153.126
MY.NET.153.145
MY.NET.153.153
MY.NET.189.45

Figure 3.2 Possible Infections resulting in Unicode "attacks"

While these hosts do not seem to be performing a large number of scans, or any sequential scans, it would be a good idea to audit these machines to verify either infection, or a lack thereof.

The other approximately 12000 hosts all originate from external sources. The external traffic seems to be devoid of any sort of sequential scanning that would be indicative of worm activity. This would indicate either directed attacks or normal traffic.

Defensive Recommendations:

I agree with John Beckers[5], that the Unicode and cginull options should be disabled in the http preprocessor. Instead the snort signatures should be kept up to date, and be allowed to catch the individual attacks that take advantage of the Unicode attack. Otherwise what has happened will continue to happen, the large number of legitimate uses of Unicode will overwhelm the malicious uses, making it a huge task to find the real problems.

Detect #2 SMB Name Wildcard

Number of Occurrences: 57249

Description of Vulnerability:

This event indicates a standard netbios name table retrieval query[6]. This is used to find out information about Windows shares when only an IP address is known. Malicious users can use this information to find out workstation names, domain, and users who are logged in. This is the method that the Opaserv worm attacks systems.

Log of Detect:

```
11/21-00:25:03.546648  [**] SMB Name Wildcard [**] 61.78.186.31:1046 ->
MY.NET.133.218:137
11/21-00:25:03.699402  [**] SMB Name Wildcard [**] 61.78.186.31:1046 ->
MY.NET.133.219:137
11/21-00:25:03.857493  [**] SMB Name Wildcard [**] 61.78.186.31:1046 ->
MY.NET.133.220:137
11/23-05:07:29.470678  [**] SMB Name Wildcard [**] 68.160.53.166:17612 ->
MY.NET.133.84:137
11/23-05:07:29.766341  [**] SMB Name Wildcard [**] 68.160.53.166:17618 ->
MY.NET.133.86:137
11/23-05:07:30.216847  [**] SMB Name Wildcard [**] 68.160.53.166:17624 ->
MY.NET.133.89:137
11/25-23:29:11.384512  [**] SMB Name Wildcard [**] 200.206.134.80:1027 ->
MY.NET.134.236:137
11/25-23:29:11.535130  [**] SMB Name Wildcard [**] 200.206.134.80:1027 ->
MY.NET.134.237:137
11/25-23:29:13.993068  [**] SMB Name Wildcard [**] 200.206.134.80:1027 ->
MY.NET.134.240:137
```

Summary:

This sort of traffic has increased dramatically since September 2002, due to the rise of Opaserv and similar worm infections. All of the events from these logs originate from external sources, except for two; 192.168.5.2 and 192.168.0.7. However it should be noted that an interesting side effect of the

network.vbs worm is to show scans from two IP addresses from the same host, one a legitimate address and one a private (RFC1918) address. Most likely this is the case here, but it would not hurt to check these two IP addresses if they are assigned anywhere on your network.

Defensive Recommendations:

I would suggest blocking of access to ports 137 and 139 to external traffic. The Microsoft file-sharing protocol is meant to be used internally only, and there is no need for external sites to have access to these ports.

Detect #3 TFTP – External UDP connection to internal tftp server

Number of Occurrences: 22435

Description of Vulnerability:

TFTP is a very simple protocol used to transfer files[7]. It is not made with security in mind. It uses no method of authentication, and access to it must be controlled by other means. An initial request to UDP port 69 on the host machine is what is used as the qualifier for this snort rule. Due to the insecure nature of the protocol, to allow outside access to a tftp server, would possibly allow unknown users to read and write files to the host machine.

Log of Detect:

```
11/21-00:09:15.408135  [**] TFTP - External UDP connection to internal tftp server [**]
MY.NET.111.235:69 -> 192.168.0.253:2004
11/21-00:09:15.408257  [**] TFTP - External UDP connection to internal tftp server [**]
MY.NET.111.232:69 -> 192.168.0.253:2004
11/21-00:09:15.408269  [**] TFTP - External UDP connection to internal tftp server [**]
MY.NET.111.231:69 -> 192.168.0.253:2004
11/25-18:26:00.194315  [**] TFTP - External UDP connection to internal tftp server [**]
63.250.205.23:8192 -> MY.NET.112.223:69
11/25-16:24:04.382737  [**] TFTP - External UDP connection to internal tftp server [**]
66.199.142.165:8260 -> MY.NET.122.120:69
11/23-18:17:22.693800  [**] TFTP - External UDP connection to internal tftp server [**]
63.250.205.26:8918 -> MY.NET.53.46:69
11/21-18:58:47.237242  [**] TFTP - External UDP connection to internal tftp server [**]
63.250.205.104:21276 -> MY.NET.87.71:69
11/21-18:58:47.611926  [**] TFTP - External UDP connection to internal tftp server [**]
63.250.205.104:21276 -> MY.NET.87.71:69
```

Summary:

Of the 22435 events, all but 5 of the events were to the host 192.168.0.253, clearly an internal host. This host was accessed by only 4 hosts, all of which reside in the MY.NET segment. So clearly this rule was picking up bad traffic, as this server is only being accessed by internal machines. The remaining five accesses come from the IPs listed below:

63.250.205.23 66.199.142.165 63.250.205.26 63.250.205.104

Figure 3.3 External TFTP accesses

However, these accesses were not to the same machine, and access four separate hosts in the MY.NET segment. Doing some research, comes up with 3 of these hosts belong to Yahoo, and the remaining host to Peer 1 Internet Bandwidth and Server Co-Location Facilities. Directing a web browser to these addresses provide some insight to this traffic, as each hosts attempts to start a video streams. This is most likely a use of some Messaging service such as Yahoo messenger that includes built-in video features.

Defensive Recommendations:

There is no real threat represented by the data presented in these logs. As long as the TFTP server resides on an internal address like 192.168.0.253 external accesses to it should not be too much concern. I would recommend some tweaking of the rule to prevent it from alerting on accesses from MY.NET however, in order to reduce the amount of false positives.

Detect #4 Tiny Fragments – Possible Hostile Activity

Number of Occurrences: 17778

Description of Vulnerability:

With many IP implementations it is possible to impose an unusually small fragment size on outgoing packets. If the fragment size is made small enough to force some of a TCP packet's TCP header fields into the second fragment, filter rules that specify patterns for those fields will not match[8].

Log of Detect:

```
11/21-08:50:28.976825  [**] Tiny Fragments - Possible Hostile Activity [**] 68.55.87.49 -
> MY.NET.168.231
11/21-08:50:32.224001  [**] Tiny Fragments - Possible Hostile Activity [**] 68.55.87.49 -
> MY.NET.168.231
11/21-08:50:34.512391  [**] Tiny Fragments - Possible Hostile Activity [**] 68.55.87.49 -
> MY.NET.168.231
11/23-00:22:42.202413  [**] Tiny Fragments - Possible Hostile Activity [**] 68.36.243.142
-> MY.NET.140.47
11/23-00:22:42.226113  [**] Tiny Fragments - Possible Hostile Activity [**] 68.36.243.142
-> MY.NET.140.47
11/23-14:50:47.069392  [**] Tiny Fragments - Possible Hostile Activity [**]
12.218.242.149 -> MY.NET.70.176
11/23-14:50:47.087513  [**] Tiny Fragments - Possible Hostile Activity [**]
12.218.242.149 -> MY.NET.70.176
11/21-15:25:53.165433  [**] Tiny Fragments - Possible Hostile Activity [**] 67.68.200.43
-> MY.NET.88.220
11/22-02:55:49.996860  [**] Tiny Fragments - Possible Hostile Activity [**] 80.116.226.41
-> MY.NET.70.176
11/22-05:19:53.876459  [**] Tiny Fragments - Possible Hostile Activity [**] 68.42.122.189
-> MY.NET.70.176
```

Summary:

This data represents possible attacks against the University network. Other possibilities include VPN or DSL traffic, which sometimes have different MTU sizes than that standard 1500 for Ethernet, which in turn can cause problems. However it is hard to tell exactly what is going on without more information. Most of the offending hosts, seem to come from the Comcast Network. I've highlighted two sets of transactions which should be looked into further, including packet data. The contacted internal hosts should also be audited for possible compromise.

68.36.243.142 -> MY.NET.140.47
68.55.87.49 -> MY.NET.168.231

Figure 3.4 Tiny Fragment Transactions

In particular the transaction between 68.36.243.142 and MY.NET.140.47 should be scrutinized as multiple alert types came from this host. I've included more information on this host below.

Defensive Recommendations:

There is not much that can be done about tiny fragments. Dropping fragments less than a certain size, could drop legitimate traffic. The best action to take, is to maintain a vigilant eye, and continue to check all the hosts that are the target of the Tiny Fragments event.

Another good thing to check, is the fragment size that generates this event. It should be validated to make sure that it isn't too small.

Detect #5 Watchlist 000220 IL – ISDNNET 990517
Number of Occurrences: 13606

Description of Vulnerability:

This is not an actual vulnerability. It is a watch list that has been created by the University to monitor a certain group of IPs, those owned by ISDN Net Ltd., an ISP located in Israel. Attempting to judge from the category, it seems it was put in place around May of 1999.

Log of Detect:

```
11/21-05:20:51.106870  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.76.25:4577 ->
MY.NET.150.220:1214
11/21-05:22:01.330141  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.76.25:4630 ->
MY.NET.150.133:1214
11/21-06:00:41.074973  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.112.72:80 ->
MY.NET.188.19:2881
11/21-06:00:41.218196  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.112.72:80 ->
MY.NET.188.19:2881
11/22-09:47:39.482106  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.35.118:1214 -
> MY.NET.91.252:1898
```

```
11/22-09:47:39.482193  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.35.118:1214 -
> MY.NET.91.252:1898
11/22-09:47:39.482206  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.35.118:1214 -
> MY.NET.91.252:1898
11/23-03:45:15.741960  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.86.36:2363 ->
MY.NET.113.4:1214
11/23-03:45:28.038350  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.86.36:2363 ->
MY.NET.113.4:1214
11/23-03:45:33.129898  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.86.36:2363 ->
MY.NET.113.4:1214
```

Summary:

When examining this event, it is nice to understand why the watch list was put into place. Doing some research into ISDN Net Ltd. I found some discussion on the incidents list, located at Insecure.org[9]. It seems around July 2000 a large number of distributed scans was seen coming from this network. That brings two questions to the front. Were the scans being seen as early as May 1999, and are these scans still occurring. The first question cannot be answered alone from the data presented here, and unfortunately research on the Internet indicated nothing about it. The second question can be answered using the logs provided, and the answer would appear to be no. Most of the traffic seems to be originating from port 80, or port 1214. Port 80 is of course web traffic, and probing many of these hosts turn up web servers. One hosts even redirected to download.com, offering the opportunity to download iMesh, a peer-to-peer client. The other port 1214[10], is the designated port of the popular peer-to-peer client KaZaa. So it would seem no specific malicious traffic is originating from this zone anymore.

Defensive Recommendations:

I would recommend that this filter be reevaluated. It seems to be unnecessary and causes needless data in the logs that need to be sorted through. On a side note, peer-to-peer services have, over the past years, become a popular method for transferring data, both legal and illegal, as well as a popular target vector for viruses and other malicious code. It can also become a hindrance in terms of bandwidth being absorbed into this type of traffic. I would recommend to the University that they evaluate their stand on peer-to-peer networks, if they have not already done so.

Detect #6 spp_http_decode: CGI Null Byte attack detected

Number of Occurrences: 12332

Description of Vulnerability:

This vulnerability takes advantage of the difference in how Perl evaluates strings, versus how C evaluates strings. Perl recognizes a Null as a non-terminating character, whereas when C interprets it, it will stop when it comes to a Null. This is used maliciously by inserting the Null character “%00” into a CGI script that does file access. When a Perl script evaluates, if says that “/etc/passwd\0” is not the same as “/etc/passwd.” But when it is passed into the C system calls that do the file access, the C reads it as “/etc/passwd” because it stops at the first Null[11]. Malicious users can use these programs to retrieve valuable data about a resource.

Log of Detect:

```
11/21-13:03:20.729738  [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.115.184:1721 -> 64.14.122.229:80
11/21-13:03:20.729738  [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.115.184:1721 -> 64.14.122.229:80
11/21-13:03:20.729738  [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.115.184:1721 -> 64.14.122.229:80
11/25-22:21:07.165115  [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.162.18:51767 -> 209.73.180.8:80
11/25-22:31:53.865842  [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.162.18:51778 -> 209.73.180.8:80
11/25-22:36:54.862633  [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.162.18:51785 -> 209.73.180.8:80
11/23-10:34:31.180160  [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.53.36:2693 -> 209.10.239.135:80
11/23-10:34:31.269908  [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.53.36:2694 -> 209.10.239.135:80
11/23-10:34:31.269908  [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.53.36:2694 -> 209.10.239.135:80
```

Summary:

Like the IIS Unicode attacks, this traffic is mostly internal to external. Of the 12332 occurrences, 60 of them were from external source, and all of those were to the same host. The rest originated from MY.NET. Also like the IIS Unicode attack, you will see a lot of false positives from sites that use urlencoded binary data, or if you are scanning port 443 and picking up SSLencrypted traffic[12]. It is hard to tell more about these events without the data from the packets.

Defensive Recommendations:

Once again, as with the IIS Unicode attacks, it is better to use the signatures that pick up the specific attacks. This would reduce the noise, and make it easier to find threats to the security of the network.

Top Talkers

Top Talkers have been chosen from external addresses only.

Top 10 Talkers from Alerts

Source Address	No. of Occurences
68.36.243.142	18267
194.106.96.8	4129
212.179.35.118	2349
212.179.35.128	1822
212.179.104.142	1739
219.63.120.93	1498
212.179.21.161	1427
200.158.16.118	1328
80.11.88.6	1195
217.225.223.230	1108

Figure 3.5 Top 10 Talkers(alerts)

A. IP Address: 68.36.243.142

Log of Activity:

```
11/23-00:22:22.098885  [**] Tiny Fragments - Possible Hostile Activity [**]
68.36.243.142 -> MY.NET.140.47
11/23-00:22:22.866494  [**] Tiny Fragments - Possible Hostile Activity [**]
68.36.243.142 -> MY.NET.140.47
11/23-00:22:22.930378  [**] Tiny Fragments - Possible Hostile Activity [**]
68.36.243.142 -> MY.NET.140.47
```

Summary of Activity:

This host is the source of multiple events, include Tiny Fragments, Null scan!, and SYN-FIN scan!. From the logs provided, it cannot be determined the exact nature of the traffic. This host should be looked into further, and if possible packet data of the traffic also examined.

B. IP Address: 194.106.96.8

Log of Activity:

```
11/22-10:30:29.231681  [**] Queso fingerprint [**] 194.106.96.8:43216 ->
MY.NET.70.231:80
11/22-10:30:53.201199  [**] Queso fingerprint [**] 194.106.96.8:43592 ->
MY.NET.70.231:80
11/22-10:30:57.567908  [**] Queso fingerprint [**] 194.106.96.8:43652 ->
MY.NET.70.231:80
```

Summary of Activity:

This host is listed as performing a Queso fingerprint against the network, however as the destination traffic is limited into scope, it is most likely just a false positive. However I have include more data about this

host below. It would be recommended to check the packet data for this host as well.

C. IP Address: 212.179.35.118

Log of Activity:

```
11/22-09:47:36.166654  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.35.118:1214 -> MY.NET.91.252:1898
11/22-09:47:39.477815  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.35.118:1214 -> MY.NET.91.252:1898
11/22-09:47:39.482106  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.35.118:1214 -> MY.NET.91.252:1898
```

Summary of Activity:

This host was alerted on because it is in a range of IPs on Watchlist 000220. Most of the traffic seen on this watchlist is peer to peer network traffic.

D. IP Address: 212.179.35.128

Log of Activity:

```
11/22-15:53:01.751874  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.35.128:80 -> MY.NET.177.34:2265
11/22-15:53:01.757263  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.35.128:80 -> MY.NET.177.34:2265
11/22-15:53:01.757430  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.35.128:80 -> MY.NET.177.34:2265
```

Summary of Activity:

This host was alerted on because it is in a range of IPs on Watchlist 000220. Most of the traffic seen on this watchlist is peer to peer network traffic.

E. IP Address: 212.179.104.142

Log of Activity:

```
11/25-00:32:08.905760  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.104.142:2370 -> MY.NET.53.41:2234
11/25-00:32:09.125795  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.104.142:2370 -> MY.NET.53.41:2234
11/25-00:32:09.131579  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.104.142:2370 -> MY.NET.53.41:2234
```

Summary of Activity:

This host was alerted on because it is in a range of IPs on Watchlist 000220. Most of the traffic seen on this watchlist is peer to peer network traffic.

F. IP Address: 219.63.120.93

Log of Activity:

```
11/21-17:55:07.060623  [**] High port 65535 udp - possible Red Worm - traffic [**]
219.63.120.93:65535 -> MY.NET.70.176:6257
11/21-18:36:35.633356  [**] High port 65535 udp - possible Red Worm - traffic [**]
219.63.120.93:65535 -> MY.NET.70.176:6257
11/21-18:36:36.672588  [**] High port 65535 udp - possible Red Worm - traffic [**]
219.63.120.93:65535 -> MY.NET.70.176:6257
```

Summary of Activity:

This host was alerted on because of the originating hosts. However this host isn't scanning, and isn't targeting port 80. This is just a noisy false positive. It is most likely peer to peer traffic.

G. IP Address: 212.179.21.161

Log of Activity:

```
11/23-14:00:51.423797  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.21.161:1682 -> MY.NET.84.151:1214
11/23-14:00:54.393558  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.21.161:1682 -> MY.NET.84.151:1214
11/23-14:00:54.994447  [**] Watchlist 000220 IL-ISDNNET-990517 [**]
212.179.21.161:1682 -> MY.NET.84.151:1214
```

Summary of Activity:

This host was alerted on because it is in a range of IPs on Watchlist 000220. Most of the traffic seen on this watchlist is peer to peer network traffic.

H. IP Address: 200.158.16.118

Log of Activity:

```
11/21-22:30:01.049977  [**] FTP DoS ftpd globbing [**] 200.158.16.118:56348 ->
MY.NET.104.104:21
11/21-22:30:02.884789  [**] FTP DoS ftpd globbing [**] 200.158.16.118:56348 ->
MY.NET.104.104:21
11/21-04:18:20.697871  [**] Port 55850 tcp - Possible myserver activity - ref.
010313-1 [**] 200.158.16.118:55850 -> MY.NET.22.9:21
```

Summary of Activity:

This event is most likely a false positive. With the number of accesses that occurred to this machine, it is most likely just normal traffic that was some how caught by the Snort signature. If it were an actual attempt, there would have been less events.

I. IP Address: 80.11.88.6

Log of Activity:

```
11/24-09:54:13.556899  [**] FTP DoS ftpd globbing [**] 80.11.88.6:2509 ->
MY.NET.100.158:21
11/24-09:55:24.766190  [**] FTP DoS ftpd globbing [**] 80.11.88.6:2509 ->
MY.NET.100.158:21
11/24-09:55:27.774136  [**] FTP DoS ftpd globbing [**] 80.11.88.6:2509 ->
MY.NET.100.158:21
```

Summary of Activity:

This event is most likely a false positive. With the number of accesses that occurred to this machine, it is most likely just normal traffic that was some how caught by the Snort signature. If it were an actual attempt, there would have been less events.

J. IP Address: 217.225.223.230

Log of Activity:

```
11/23-17:58:05.369610  [**] FTP DoS ftpd globbing [**] 217.225.223.230:2750 ->
MY.NET.100.158:21
11/23-17:58:10.289927  [**] FTP DoS ftpd globbing [**] 217.225.223.230:2750 ->
MY.NET.100.158:21
11/23-17:58:21.311726  [**] FTP DoS ftpd globbing [**] 217.225.223.230:2750 ->
MY.NET.100.158:21
```

Summary of Activity:

This event is most likely a false positive. With the number of accesses that occurred to this machine, it is most likely just normal traffic that was some how caught by the Snort signature. If it were an actual attempt, there would have been less events.

Top 10 Talkers from Scans

Source Address	No. of Occurences
211.237.39.111	17933
213.82.34.146	16745
217.84.36.27	11629
212.78.131.38	10643
205.188.228.65	9513
195.188.210.50	9132
80.13.105.187	9057

202.219.102.54	7536
65.67.179.7	7286
203.69.244.44	7098

Figure 3.6 Top 10 Talkers(scans)

A. IP Address: 211.237.39.111

Log of Activity:

```
Nov 21 01:48:59 213.82.34.146:50816 -> MY.NET.21.11:21 SYN ******S*
Nov 21 01:48:59 213.82.34.146:50815 -> MY.NET.21.10:21 SYN ******S*
Nov 21 01:48:59 213.82.34.146:50830 -> MY.NET.21.12:21 SYN ******S*
```

Summary of Activity:

This host is scanning for port 21(ftp). This is not a targeted action. This host has a record on Dshield.org. This activity represents a low threat.

B. IP Address: 213.82.34.146

Log of Activity:

```
Nov 21 01:48:59 213.82.34.146:50810 -> MY.NET.21.5:21 SYN ******S*
Nov 21 01:48:59 213.82.34.146:50811 -> MY.NET.21.6:21 SYN ******S*
Nov 21 01:48:59 213.82.34.146:50812 -> MY.NET.21.7:21 SYN ******S*
```

Summary of Activity:

This host is scanning for port 21(ftp). This is not a targeted action. This host has a record on Dshield.org. This activity represents a low threat.

C. IP Address: 217.84.36.27

Log of Activity:

```
Nov 24 06:09:18 217.84.36.27:62595 -> MY.NET.10.111:135 SYN ******S*
Nov 24 06:19:06 217.84.36.27:63595 -> MY.NET.53.39:445 SYN ******S*
Nov 24 06:20:59 217.84.36.27:64075 -> MY.NET.10.84:80 SYN ******S*
```

Summary of Activity:

This host is scanning for ports 135, 445, and occasional accesses to port 80. This has has no record on Dshield.org.

D. IP Address: 212.78.131.38

Log of Activity:

```
Nov 24 01:18:01 212.78.131.38:4898 -> MY.NET.10.9:80 SYN ******S*
Nov 24 01:18:01 212.78.131.38:4899 -> MY.NET.10.10:80 SYN ******S*
Nov 24 01:18:01 212.78.131.38:4900 -> MY.NET.10.11:80 SYN ******S*
```

Summary of Activity:

This host is scanning for port 80(http). This is not a targeted act. This could be worm traffic, or just an automated scanner. This host has no record on Dshield.org. This activity represents a low threat.

E. IP Address: 205.188.228.65

Log of Activity:

```
Nov 21 11:36:30 205.188.228.65:28436 -> MY.NET.106.178:6970 UDP
Nov 21 11:36:30 205.188.228.65:13264 -> MY.NET.91.51:6970 UDP
Nov 21 11:36:30 205.188.228.65:10184 -> MY.NET.157.101:6970 UDP
```

Summary of Activity:

This host has a record on Dshield.org, however it is in error. This host is part of spinner.com(a net radio service). Port 6970 is one of the ports that the RealAudio server uses to relay it's content[13]. This activity represents no threat.

F. IP Address: 195.188.210.50

Log of Activity:

```
Nov 23 11:33:12 195.188.210.50:4722 -> MY.NET.10.25:80 SYN ******S*
Nov 23 11:33:12 195.188.210.50:4630 -> MY.NET.10.2:80 SYN ******S*
Nov 23 11:33:12 195.188.210.50:4634 -> MY.NET.10.3:80 SYN ******S*
```

Summary of Activity:

This host is scanning for port 80(http). This is not a targeted attack. This host has a record on Dshield.org for similar scanning. This activity represents a low threat.

G. IP Address: 80.13.105.187

Log of Activity:

```
Nov 24 13:25:16 80.13.105.187:3369 -> MY.NET.108.45:445 SYN ******S*
Nov 24 13:25:17 80.13.105.187:3373 -> MY.NET.108.30:445 SYN ******S*
Nov 24 13:25:19 80.13.105.187:3376 -> MY.NET.108.30:445 SYN ******S*
```

Summary of Activity:

This host is scanning for port 135, and 445. This host has no record on Dshield.org.

H. IP Address: 202.219.102.54

Log of Activity:

```
Nov 23 17:01:38 202.219.102.54:4924 -> MY.NET.10.74:80 SYN ******S*
Nov 23 17:01:38 202.219.102.54:4925 -> MY.NET.10.75:80 SYN ******S*
Nov 23 17:01:38 202.219.102.54:4842 -> MY.NET.10.41:80 SYN ******S*
```

Summary of Activity:

This host is scanning for port 80(http). This is not a targeted act. This host has a record on Dshield.org. This activity represents a low threat.

I. IP Address: 65.67.179.7

Log of Activity:

```
Nov 21 07:22:48 65.67.179.7:4599 -> MY.NET.10.36:80 SYN ******S*
Nov 21 07:22:48 65.67.179.7:4600 -> MY.NET.10.37:80 SYN ******S*
Nov 21 07:22:48 65.67.179.7:4601 -> MY.NET.10.38:80 SYN ******S*
```

Summary of Activity:

This host is scanning for port 80(http). This is not a targeted act. This host has a record on Dshield.org for similar scanning. This activity represents a low threat.

J. IP Address: 203.69.244.44

Log of Activity:

```
Nov 22 06:54:42 203.69.244.44:39778 -> MY.NET.10.69:80 SYN ******S*
Nov 22 06:54:42 203.69.244.44:39779 -> MY.NET.10.70:80 SYN ******S*
Nov 22 06:54:42 203.69.244.44:39780 -> MY.NET.10.71:80 SYN ******S*
```

Summary of Activity:

This host is scanning for port 80(http). This is not a targeted act. This host has a record on Dshield.org for port 21 scanning. This activity represents a low threat.

Highlighted Hosts

A. IP Address: 212.179.76.25

Registration Information:

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

inetnum:      212.179.76.16 - 212.179.76.31
netname:      KIBBUTZ-MASADA
mnt-by:       INET-MGR
descr:        KIBBUTZ-MASADA-LAN
country:      IL
admin-c:      MR916-RIPE
tech-c:       ZV140-RIPE
status:       ASSIGNED PA
notify:       hostmaster@bezeqint.net
changed:      hostmaster@bezeqint.net 20020826
source:       RIPE

route:        212.179.64.0/18
descr:        ISDN Net Ltd.
origin:       AS8551
notify:       hostmaster@bezeqint.net
mnt-by:       AS8551-MNT
changed:      hostmaster@bezeqint.net 20020618
source:       RIPE

person:       Miri Roaky
address:      bezeq-international
address:      40 hashacham
address:      petach tikva 49170 Israel
phone:        +972 1 800800110
fax-no:       +972 3 9203033
e-mail:       hostmaster@bezeqint.net
nic-hdl:      MR916-RIPE
changed:      hostmaster@bezeqint.net 20021027
source:       RIPE

person:       Zehavit Vigder
address:      bezeq-international
address:      40 hashacham
address:      petach tikva 49170 Israel
phone:        +972 1 800800110
fax-no:       +972 3 9203033
e-mail:       hostmaster@bezeqint.net
```

```
nic-hdl:      ZV140-RIPE
changed:      hostmaster@bezeqint.net 20021027
source:       RIPE
```

Reason Included:

This is one of the hosts that is included under Watchlist 220. The record presented is the segment that is covered by the watchlist.

B. IP Address: 207.233.194.203

Registration Information:

```
OrgName:    Whiting Turner Contracting Company
OrgID:      WTC-11

NetRange:   207.233.194.0 - 207.233.194.255
CIDR:       207.233.194.0/24
NetName:    DIGINET173
NetHandle:  NET-207-233-194-0-1
Parent:     NET-207-233-128-0-1
NetType:    Reassigned
Comment:
RegDate:    1999-11-17
Updated:    1999-11-17

TechHandle: PCI-ORG-ARIN
TechName:   Network Operations Center
TechPhone:  +1-703-736-9800
TechEmail:  noc@diginetusa.net

# ARIN Whois database, last updated 2003-01-20 20:00
# Enter ? for additional hints on searching ARIN's
Whois database.
```

Reason Included:

This host was picked up for possible Trojan activity(SubSeven). It was the number one originating host in that category, and as such I felt it deserved more attention.

C. IP Address: 68.36.243.142

Registration Information:

```
OrgName:    Comcast Cable Communications, Inc.
OrgID:      CMCS
```

```
NetRange:   68.36.0.0 - 68.39.255.255
CIDR:       68.36.0.0/14
NetName:    JUMPSTART-NJ-NORTH-1
NetHandle:  NET-68-36-0-0-1
Parent:     NET-68-32-0-0-1
NetType:    Reassigned
NameServer: NS01.JDC01.PA.COMCAST.NET
NameServer: NS02.JDC01.PA.COMCAST.NET
Comment:
RegDate:    2002-02-22
Updated:    2002-07-16

TechHandle: IC161-ARIN
TechName:   Comcast Cable Communications, Inc.
TechPhone:  +1-856-317-7300
TechEmail:  cips-ip-registration@cable.comcast.com

OrgAbuseHandle: NAPO-ARIN
OrgAbuseName:   Network Abuse and Policy Observance
OrgAbusePhone:  +1-856-317-7272
OrgAbuseEmail:  abuse@comcast.net

OrgTechHandle: IC161-ARIN
OrgTechName:   Comcast Cable Communications, Inc.
OrgTechPhone:  +1-856-317-7300
OrgTechEmail:  cips-ip-registration@cable.comcast.com

# ARIN Whois database, last updated 2003-01-20 20:00
# Enter ? for additional hints on searching ARIN's
Whois database.
```

Reason Included:

This host was picked up for multiple alerts, including Tiny Fragments, Null scan!, and SYN-FIN scan!. It should be checked out further in the logs, including looking for packet data.

D. IP Address: 211.237.39.111

Registration Information:

```
# ENGLISH

KRNIC is not ISP but National Internet Registry
similar with APNIC.
```

```
Please see the following end-user contacts for IP
address information.

IP Address          : 211.237.39.64-211.237.39.127
Network Name        : HANINTERNET-LLINE-ITONETPC
Connect ISP Name    : HANINTERNET
Connect Date        : 20010217
Registration Date   : 20010221

[ Organization Information ]
Orgnization ID      : ORG202425
Org Name            : ITONETPC
State               : SEOUL
Address             : 701-2 YUCKSAM KANGNAM
Zip Code            : 135-080

[ Admin Contact Information]
Name                : TAEHOON KIM
Org Name            : ITONETPC
State               : SEOUL
Address             : 701-2 YUCKSAM KANGNAM
Zip Code            : 135-080
Phone               : +82-2-3462-7309
E-Mail              : IP@HANINTERNET.CO.KR

[ Technical Contact Information ]
Name                : TAEHOON KIM
Org Name            : ITONETPC
State               : SEOUL
Address             : 701-2 YUCKSAM KANGNAM
Zip Code            : 135-080
Phone               : +82-2-3462-7309
E-Mail              : IP@HANINTERNET.CO.KR

-------------------------------------------------
-------------------------

If the above contacts are not rechable, please see the
following ISP contacts
for relevant information or network abuse complaints.

[ ISP IP Admin Contact Information ]
Name                : Sunhwa Jung
Phone               : +82-2-860-8160
Fax                 : +82-2-852-8535
E-Mail              : iservice@haninternet.co.kr
```

```
[ ISP IP Tech Contact Information ]
Name                : Raeeun Yeo
Phone               : +82-2-860-8144
Fax                 : +82-2-852-8535
E-Mail              : ip@haninternet.co.kr

[ ISP Network Abuse Contact Information ]
Name                : Sangwon So
Phone               : +82-2-860-8002
Fax                 : +82-2-852-8535
E-Mail              : support@haninternet.co.kr
```

Reason Included:

This is the number one top talker for scans. I decided it was worthwhile to put in here.

E. IP Address: 194.106.96.8

Registration Information:

```
inetnum:      194.106.96.0 - 194.106.96.255
netname:      MLO-BACKBONE
descr:        MicroLink Online Backbone
descr:        Tallinn
descr:        Estonia
country:      EE
admin-c:      AR38-RIPE
tech-c:       AK67-RIPE
rev-srv:      ns.online.ee
rev-srv:      ns2.online.ee
status:       ASSIGNED PA
mnt-by:       AS5546-MNT
changed:      hostmaster@ripe.net 19960124
changed:      andre@online.ee 19960129
changed:      andre@online.ee 19961031
changed:      andre@online.ee 20011218
source:       RIPE

route:        194.106.96.0/19
descr:        MicroLink Online
descr:        Parnu mnt. 158
descr:        11317 Tallinn
descr:        Estonia
origin:       AS5546
mnt-by:       AS5546-MNT
changed:      andre@ml.ee 19960214
```

```
changed:      andre@ml.ee 19991210
source:       RIPE

person:       Avo Raup
address:      MicroLink Online
address:      Parnu mnt. 158
address:      11317 Tallinn
address:      Estonia
phone:        +372 6501 707
fax-no:       +372 6501 708
e-mail:       afka@online.ee
nic-hdl:      AR38-RIPE
changed:      afka@online.ee 19951108
changed:      andre@ml.ee 19991210
source:       RIPE

person:       Andres Kroonmaa
address:      MicroLink Online
address:      Parnu mnt. 158
address:      11317 Tallinn
address:      Estonia
phone:        +372 6501 731
fax-no:       +372 6501 708
e-mail:       andre@online.ee
nic-hdl:      AK67-RIPE
changed:      afka@online.ee 19951108
changed:      andre@ml.ee 19991210
source:       RIPE
```

Reason Included:

This host was picked up for Queso fingerprinting. It would be worthwhile to further check data about this host. If more confirming information is find, it would be a good idea to send an abuse letter.

Data Analysis With Link Graph

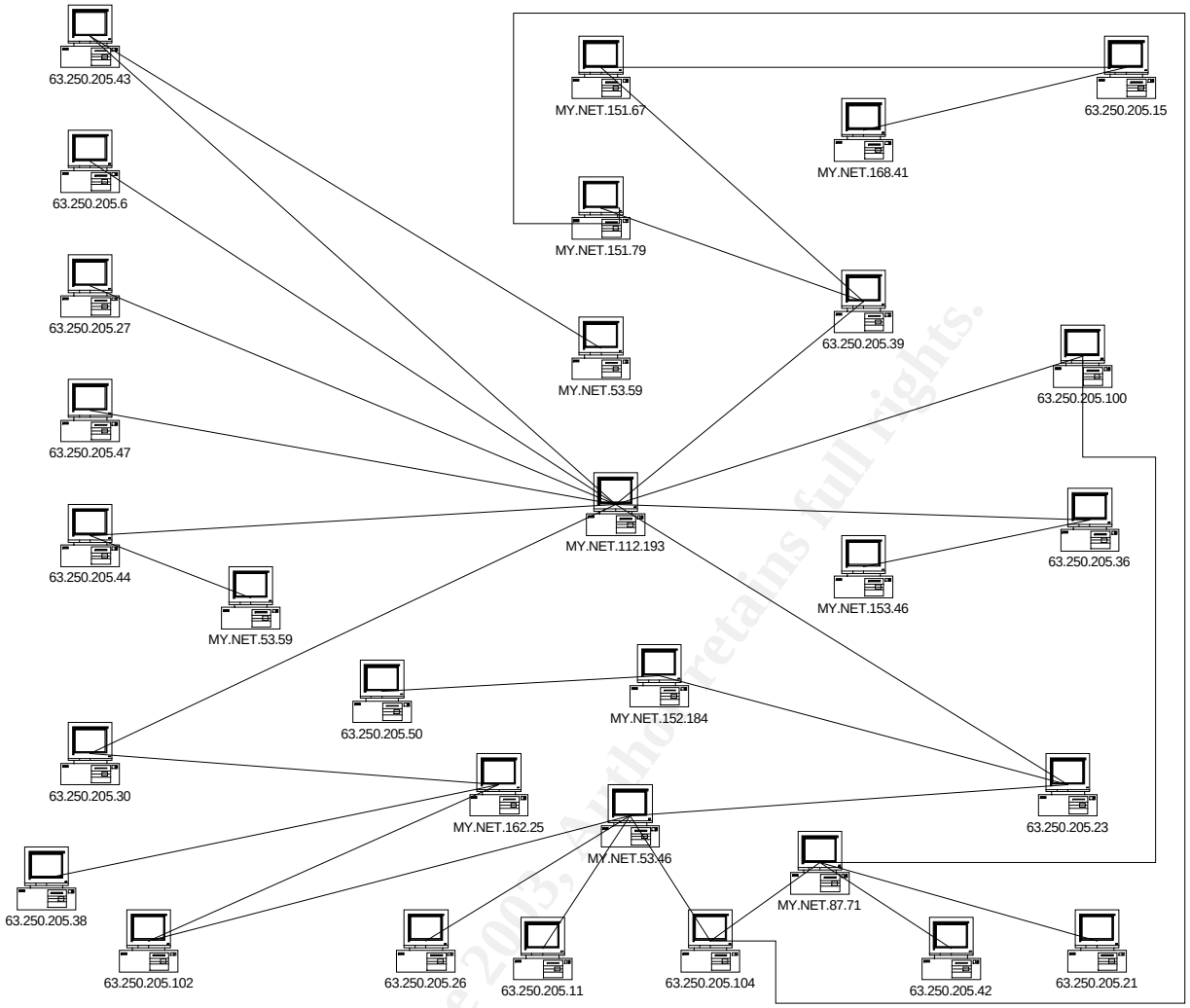

Figure 3.7 Link Graph of Traffic from 63.250.205.x

Just by looking through the logs, it's hard to see the relationship of these hosts. What we are seeing, is a lot of traffic from the 63.250.205.x network. Now these addresses are in a netblock, that is owned by Yahoo Broadcasting, according to Arin. While the big name on the netblock, would give us a pause, this traffic is nonetheless suspicious. It seems to be widespread, and across many ports to be able to actually pin down to one type of traffic. I would recommend auditing some of the targeted machines, especially MY.NET.112.193. If any evidence of tampering is found, then it would be advisable to begin a dialogue with the admins at Yahoo Broadcasting in order to better pin down the traffic, and to check for possible compromise.

Internal Activity

Port 55850 tcp – Possible myserver activity ref. 010313-1

Number of Occurrences: 489

Description of Vulnerability:

Myserver is a DDoS agent that binds on UDP port 55850[9].

Log of Detect:

```
11/21-06:25:00.538468  [**] Port 55850 tcp - Possible myserver activity - ref. 010313-1
[**] 218.145.25.43:55850 -> MY.NET.151.73:80
11/21-06:25:00.539062  [**] Port 55850 tcp - Possible myserver activity - ref. 010313-1
[**] MY.NET.151.73:80 -> 218.145.25.43:55850
11/23-00:27:05.362899  [**] Port 55850 tcp - Possible myserver activity - ref. 010313-1
[**] MY.NET.104.204:1214 -> 24.29.115.2:55850
11/23-00:27:08.314318  [**] Port 55850 tcp - Possible myserver activity - ref. 010313-1
[**] MY.NET.104.204:1214 -> 24.29.115.2:55850
```

Summary:

The events of this type seem to be false positives. It is not UDP traffic, which is the protocol that the myserver agent is bound to. It is comprised of legitimate web browser traffic, as well as some traffic from peer to peer networks.

Defensive Recommendations:

I would recommend that UDP port 55850 be blocked on incoming connections. This would cut down on the risk of machines on MY.NET being used as Myserver agents, and would not interfere with normal traffic.

IRC Evil – running XDCC

Number of Occurrences: 463

Description of Vulnerability:

XDCC is a way of transferring files, usually via IRC. Files are usually transmitted via programs called “bots.” These bots are often setup on compromised hosts.

Log of Detect:

```
11/21-00:14:19.930824  [**] IRC evil - running XDCC [**] MY.NET.104.64:1029 ->
193.163.220.3:6667
11/21-00:20:32.942645  [**] IRC evil - running XDCC [**] MY.NET.104.64:1029 ->
193.163.220.3:6667
11/23-18:10:57.948812  [**] IRC evil - running XDCC [**] MY.NET.91.249:1603 ->
193.163.220.3:6667
11/23-19:00:01.565906  [**] IRC evil - running XDCC [**] MY.NET.91.249:1603 ->
193.163.220.3:6667
```

Summary:

Most of the connections were to servers on the Efnet IRC network. These events are most likely the result of users requesting XDCC transfers. However, these host could possibly be receiving XDCC commands. Here is a list of possibly compromised hosts which should be checked out:

MY.NET.70.91
MY.NET.91.249
MY.NET.104.64
MY.NET.108.42

Figure 3.8 Possibly Compromised Hosts

Defensive Recommendations:

Depending on the campuses policy, one of two actions can be taken. Either all access to IRC can be cut off, or selective access can be denied. While IRC can be used for illegal activities, it is also used for normal activities, and to deny the use of such services, would be something that the administration would need to put great thought in. However if access was selectively denied, then it could help to alert the administration to users who are mis-using bandwidth.

Possible Trojan server activity

Number of Occurrences: 180

Description of Vulnerability:

This rule picks up on accesses to port 27374, the default port for SubSeven. SubSeven is a popular Trojan that allows control of a client computer. IRC is a common infection vector for SubSeven.

Log of Detect:

```
11/23-08:48:57.917425  [**] Possible trojan server activity [**] 211.132.104.25:27374 ->
MY.NET.113.4:1214
11/23-08:48:57.924719  [**] Possible trojan server activity [**] 211.132.104.25:27374 ->
MY.NET.113.4:1214
11/24-11:59:52.059292  [**] Possible trojan server activity [**] 213.33.113.174:27374 ->
MY.NET.104.104:80
11/24-11:59:52.061597  [**] Possible trojan server activity [**] 213.33.113.174:27374 ->
MY.NET.104.104:80
```

Summary:

A lot of normal traffic can be seen including webserver and peer to peer traffic. However there is also some traffic that could be SubSeven traffic. In particular one source host stands out, which is the IP 207.233.194.203. I have included more information on this IP above. A number of machine have been possible compromised and would do well to have a looking over. A list of these follows:

MY.NET.190.30	MY.NET.84.147
MY.NET.190.32	MY.NET.113.4
MY.NET.190.36	MY.NET.91.104
MY.NET.190.38	MY.NET.104.104
MY.NET.190.17	MY.NET.137.7
MY.NET.190.19	MY.NET.152.169
MY.NET.190.26	MY.NET.83.146
MY.NET.190.34	MY.NET.179.77
MY.NET.190.52	

Figure 3.9 Possibly SubSeven Infected Hosts

Defensive Recommendations:

Just as with Myserver, I would recommend that this port be blocked to incoming taffic. Blocking this port will alleviate the risk of machines being controlled by versions of SubSeven configured to use port 27374, and will not interfere with legitimate traffic.

Back Orifice

Number of Occurrences: 5

Description of Vulnerability:

Back Orifice is another Trojan, that runs on port 31337. It gives the controller complete access to an infected machine.

Log of Detect:

```
11/22-16:40:00.208019 [**] Back Orifice [**] 63.250.205.30:44048 -> MY.NET.162.25:31337
11/22-16:40:01.696312 [**] Back Orifice [**] 63.250.205.30:44048 -> MY.NET.162.25:31337
11/22-17:21:02.705652 [**] Back Orifice [**] 64.152.216.82:57167 -> MY.NET.87.55:31337
11/24-12:06:50.525665 [**] Back Orifice [**] 195.92.252.254:39174 -> MY.NET.115.251:31337
```

Summary:

There were not very many events of this type. The host 63.250.205.30 is listed as owned by Yahoo! Broadcast Services, and while it could be a compromised host, is more likely a response to packet from the one host. A list of hosts that were attempted to be accessed follows below:

MY.NET.87.55 MY.NET.115.251 MY.NET.162.25

Figure 3.10 Hosts possibly infected with Back Orifice

Defensive Recommendations:

Blocking access to this port at the firewall should take care of this problem.

NIMDA – Attempt to execute cmd from campus host

Number of Occurrences: 2

Description of Vulnerability:

Nimda is a worm that takes advantage of the IIS Unicode vulnerability. It can wreak havoc across a network, and is a bandwidth hog, causing slowdowns and lag on infected networks.

Log of Detect:

```
11/21-13:05:24.030212  [**] NIMDA - Attempt to execute cmd from campus host [**]
MY.NET.27.210:1153 -> 65.54.250.120:80
11/22-07:31:47.071043  [**] NIMDA - Attempt to execute cmd from campus host [**]
MY.NET.163.146:1066 -> 65.54.250.120:80
```

Summary:

There were only two events of this type, and both events were connections to the Microsoft Download Center. It seems that MY.NET is free of the Nimda worm. These events are false positives.

Defensive Recommendations:

The University is doing pretty well in this area. However it is always a good idea to make sure that all IIS and Personal Web Servers are up to date, and therefore not vulnerable to the Unicode vulnerability.

Bugbear@MM virus in smtp

Number of Occurrences: 1

Description of Vulnerability:

Log of Detect:

```
11/21-00:38:22.628748  [**] Bugbear@MM virus in SMTP [**] 203.176.60.253:60239 ->
MY.NET.6.40:25
```

Summary:

There was only 1 event of this type, and was from an incoming mail.

Defensive Recommendations:

If not already in place, an AV scanner on the Universities mail scanners would be a good addition to the network.

Defensive Recommendations

Since the exact nature of the network is unknown, I will give some basic recommendations, based on the general layout of University networks.

Firstly, there needs to be some consensus on what to do about certain types of network traffic such as IRC and peer to peer networks. Although they are not direct security risks, the campus administrators may want to limit the activities of such usage. Once a formal policy is issued, action can be taken to either block or regulate the traffic. Another issue to be decided on, is the issue of student run services. Should students be allowed to run HTTP or FTP servers from their dorms? While disallowing would go against the common open policy of

most campuses, it keeps the number of potential security risks down. In either situation, the residential network should be behind a firewall, to protect it from the outside, and to protect the outside from it. In addition I would place a NIDS behind this firewall to monitor the residential traffic for both potential security risks, and would-be miscreants. For this I would recommend an iptables firewall, It is a powerful easy to learn tool, as well as cost effective. As well, in the hands of a creative administrator, it could easily provide solutions to the IRC and peer to peer problems in such a way as to compromise between complete shutdown and an open door. The use of a simple cron job can switch between firewall rulesets at certain times of the day allowing the type of traffic that is allowed to be dynamic, depending on the campuses policy and desires. I would also recommend that the NIDS be a Snort box for much the same reasons.

Secondly, it would need to be addressed as to whether the Department/School Networks should be put into one large administrative LAN, or if they should each be individual. The basic strategy would be the same for both, a firewall and a NIDS. However individual segments require more work, but allow greater freedom and make for less obfuscating rulesets. Either way, neither solution includes special services such as HTTP or FTP servers. These servers would need to be placed in a DMZ segment, because of the special access required for these servers.

Thirdly, there should be a firewall on the outside of the entire network. This is the firewall that will control access for all the traffic entering the network, and the DMZ segment will be off this firewall as well. This is the firewall at which you would block any special ports, such as 31337(Back Orifice) or 27374(SubSeven). This segment would also have a NIDS box on the inside of the firewall.

This type of setup will give finer control over the individual segments of the network, as well as good coverage IDS-wise and let the security administrator know what type of traffic is getting where. In addition to the hardware, there also needs to be an audit of all current machines, with focus on machines that serve content, to make sure that all programs are currently up to date. This should be monitored and kept track of, and the security administrator should keep an eye on security alerts for those that affect his network.

In general I recommend iptables as a firewall solution. It is cost effective, and easily learnable. It doesn't require special tools to access(beyond ssh), and rulesets are easily portable, as they are contained in a flat file format.As an IDS I recommend Snort. The signatures are constantly being updated and reviewed, as well as new features being added. There are also numerous programs(such as ACID, and SnortSnarf) that allow for the captured Snort data to be reviewed and manipulated.

Analysis Process

Going through the large amount of snort logs was a daunting task. After downloading the previously mentioned files, I took a few cursory glances at them, while trying to wrap my mind around the best way to go about the whole task. Looking through some of the previous GCIA practicals. I came across a Perl script, designed to separate the alerts data by messages and sources. I took this script and used it as the basis for all of my work. I'm not sure what the original author named it, but I dubbed it parse.pl, and set it to work.

```
#!/usr/bin/perl

#09/18-10:03:26.121357  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.35.118:80 -
> MY.NET.153.150:1870
# The above is for reference on how the alerts are formatted

my $count = 0;
while (<>) {
        if (/.*\[\*\*\] (.*) \[\*\*\] (.*) -> (.*)/) {

                $count++;
                my $msg = $1; my $src = $2; my $dst = $3;
                $msg =~ s/[^A-Za-z0-9 -_]//g;
                $msg =~ s/ /_/g;
                open(OUT, ">>msg/$msg");
                print OUT;
                close OUT;
                $src =~ s/:.*//g;
                open(OUT, ">>src/$src");
                print OUT;
                close OUT;
                print "Processed $count lines\n";
        } else {
                print "Skipped $_";
                $skipped .= $_;
        }
}
print "Skipped the following lines:\n$skipped\n";
```

Figure 3.11 parse.pl

Using the script as a base, I came up with several variations, for multiple purposes. parse2.pl was used to separate out scan data, while parse3.pl created a list of destination hosts, categorized by source. And lastly parse4.pl was use to list destination hosts, categorized by port.

```
#!/usr/bin/perl

#Nov 21 00:00:20 130.85.137.7:53 -> 204.183.84.240:45131 UDP
# The above is for reference on how the alerts are formatted

my $count = 0;
while (<>) {
	if (/.* [0-9][0-9]:[0-9][0-9]:[0-9][0-9] (.*) -> (.*:[0-9]*) ([A-Z]*) /) {

		$count++;
		my $msg = $3; my $src = $1; my $dst = $2;
		#$msg =~ s/[^A-Za-z0-9 -_]//g;
		$msg =~ s/ /_/g;
		open(OUT, ">>msg/$msg");
		print OUT;
		close OUT;
		$src =~ s/:.*//g;
		open(OUT, ">>src/$src");
		print OUT;
		close OUT;
		print "Processed $count lines\n";
	} else {
		print "Skipped $_";
		$skipped .= $_;
	}
}
print "Skipped the following lines:\n$skipped\n";
```

Figure 3.12 parse2.pl


```
#!/usr/bin/perl

#09/18-10:03:26.121357  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.35.118:80 -
> MY.NET.153.150:1870
# The above is for reference on how the alerts are formatted

my $count = 0;
while (<>) {
	if (/.*\[\*\*\] (.*) \[\*\*\] (.*) -> (.*)/) {

		$count++;
		my $msg = $1; my $src = $2; my $dst = $3;
		$src =~ s/:.*//g;
		$dst =~ s/:.*//g;
		open(OUT, ">>src/$src");
		print OUT $dst;
		print OUT "\n";
		close OUT;
		print "Processed $count lines\n";
	} else {
		print "Skipped $_";
		$skipped .= $_;
	}
}
print "Skipped the following lines:\n$skipped\n";
```

Figure 3.13 parse3.pl

```
#!/usr/bin/perl

#09/18-10:03:26.121357  [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.35.118:80 -
> MY.NET.153.150:1870
# The above is for reference on how the alerts are formatted

my $count = 0;
while (<>) {
        if (/.*\[\*\*\] (.*) \[\*\*\] (.*) -> (.*):([0-9]*)/) {

                $count++;
                my $msg = $1; my $src = $2; my $dst = $3; my $port = $4;
                $src =~ s/:.*//g;
                $dst =~ s/:.*//g;
                open(OUT, ">>services/$port");
                print OUT $dst;
                print OUT "\n";
                close OUT;
                print "Processed $count lines\n";
        } else {
                print "Skipped $_";
                $skipped .= $_;
        }
}
print "Skipped the following lines:\n$skipped\n";
```

Figure 3.14 parse4.pl

Once everything was sorted out, I used various shell tools, like grep, wc, and sort, to gather counts of events, and order them by occurrence. I made lists of the events that occurred the most, as well as lists of hosts that most often occurred as sources. After those list, I also made a list of events that have particular bearing on possible compromises, such as Back Orifice, and Possible Trojan Activity.

After my lists were compiled, I started researching on the exact nature of each vulnerability. The tools I used at this stage were Google, various whois databases(ARIN, RIPE, APNIC), and dig. I also took notice of information was presented that matched what I was looking at. Using the extrapolated logs, and the data that I found with research I was able to account for most events.

Over the course of doing the research and looking at the logs, I took notice of certain hosts, that seemed suspicious. I jotted these down, and used them for the hosts that I provided registration information for.

The hardest part to overcome was the sheer amount of data, but I think that I did a decent job of sorting through it, and bringing to light the important issues.

References

1. Microsoft IIS and PWD Extended Unicode Directory Traversal Vulnerability. http://online.securityfocus.com/bid/1806/discussion

2. Flooded with IIS Unicode and CGI Null Byte false positives.(Herschel Gelman). http://archives.neohapsis.com/archives/snort/2001-02/0093.html

3. Re. Overwhelmed……(Micheal R Jinks). http://lists.jammed.com/incidents/2001/06/0161.html

4. GCIA Practical.(Steven Drew). http://www.giac.org/practical/Steven_Drew_GCIA.doc

5. IIS Unicode attack detected. (John Beckers). http://www.geocrawler.com/mail/msg.php3?msg-id=6340557&list=4890

6. IDS177 "NETBIOS-NAME-QUERY".(Max Vision). http://www.whitehats.com/info/IDS177

7. The TFTP Protocol. (K. Sollins). http://www.ietf.org/rfc/rfc1350.txt

8. RFC1858. (G. Zienba). http://www.scit.wlv.ac.uk/rfc/rfc18xx/RFC1858.html

9. Security Incidents: Re: large scale distributed scan from Israel. http://list.insecure.org/lists/incidents/2000/Aug/0014.html

10. Weird scan on port 1214. (Greg Woods). http://lists.jammed.com/incidents/2001/06/6252.html

11. Perl CGI problems (rain.forest.puppy). http://www.phrack-don't-give-a-shit-about-dmca.org/phrack/55/P55-07

12. RE: [Snort-users] CGI Null Byte Attack. (Joe Stewart). http://archives.neohapsis.com/archives/snort/2000-11/0244.html

13. RE: Connection fro Unknown. (Mike Worman). http://archives.neohapsis.com/archives/incidents/2000-10/0136.html

14. RE: Port 6970 (Bill Custi). http://www.shmoo.com/mail/firewalls/jun99/msg00788.html