Global Information Assurance Certification Paper

SANS GIAC Intrusion Detection In Depth Certification (GCIA) Version 3.0

Jeff Zahr

SANS Boston Conference
September 5 - 12, 2001

Submitted November 15, 2001

Part I The State of Intrusion Detection

The state of intrusion detection technology can quite simply be summed up in one word, inefficient. The reason? The current Intrusion Detection technology in existence is mostly signature based. Throughout this paper I will use one specific intrusion example, the code known as Nimda, to demonstrate this point.

So what is a signature? An "intrusion" signature is very similar to the more commonly understood and accepted "virus" signature. To explain intrusion signatures, let's look at "virus" signatures.

Any current anti-virus software on the market runs in the background of the operating system. It looks at every file and compares it to a known list of file types. The type of file determines its' capability to execute code within that file. If the file is capable of running any form of code (be it an MS Excel or Word macro, an actual compiled executable file or a batch or script file) it searches the file for code and then compares the code contents to a known list of specific code fragments for that file type that have been deemed malicious. If a match to that code is found an alert is generated, and the file is quarantined as to avoid possible infection. This code fragment is what is known as a "signature".

Similarly, intrusion detection systems compare packets they receive against their "type" as defined in the IP header. Depending on the type of packet, it is compared to a known set of malicious criteria. The criteria are a little more complex than that of virus protection, as slight changes at the binary level, can compromise system integrity.

To explain further, a file only has content. An IP packet has the same content in the data payload, but also has all the header information that describes the type of packet, the destination and source of the packet, specific protocol options (such as Push and Ack flags in the TCP header) and some bits that are reserved for future use, among other things. Therefore, the criteria list or "signature" for intrusion detection not only contains pieces of code in the payload of the packet similar to the code fragments in "virus" signatures, but also certain bit changes within the packet (such as certain TCP flags Syn and Fin together) that have been used in the past as exploits.

The problem? The detection for either is only as good as the most recent signatures. New viruses and exploits are created constantly, and the signatures are only updated after those exploits are released into the wild, discovered, and either submitted to intrusion detection software manufacturers such as Snort IDS or Cisco software's Secure IDS or taken on one's self, analyzed, and a particular "signature" is found for that attack and the appropriate signature is created and tested.

How long does that process take? As a general rule for viruses, a seemingly short period of time. For the most part, viruses come in the form of an executable program, often times attached to another program or email. If there already exists a virus "signature" the infected file is normally flagged as infected and "quarantined". But what if it isn't?

With traditional viruses, the full virus code is packaged and distributed in some way such as email or floppy disk. The infection occurs when that code is executed. Whether or not it is executed, it is in fact whole before it begins. That file can than be sent to a lab and thoroughly analyzed. If its' a batch file, script or macro, it can be opened directly in an editing mode instead of executed to look at it's contents to determine the particular "signature" that can identify this virus. If it's an executable, it can be run and the code can be "stepped through" with special software tools, showing the analyst the exact intent of the virus producer.

If there is no virus "signature", it is very possible that we become infected with that virus and begin to display signs of infection. Those signs can also be potentially investigated and a determination as to what changes on the system were made could also produce a signature without the actual virus code, but it obviously takes much more effort. What I more commonly done, is that the originally infected file is deconstructed as mentioned above to locate that "signature". With Internet intrusion attacks the case is different.

Let's think for a moment about our intrusion detection systems. A hacker attempts a TCP connection, gets the signal from our web server to start sending packets, and the first set comes in. Because we are patched, the first set of instructions is rejected. Attack is stopped short. Our patches worked, but what did we learn about the attack?

If we had an intrusion detection system on the network and this was a new exploit (before a signature is released), nothing, because if it didn't match a specific signature, nothing was logged. Because the server is patched, we didn't get infected and therefore have no concept of what the attacker was attempting to do. And, the complete attack may never be known, as if a box is exploited there is no way to tell what pieces are there from the exploit directly, indirectly or were there before the exploit. But there are even more issues with the current state of intrusion detection.

If we had a packet sniffer on the network in addition to our IDS, we might very well see the first full packets that attempted to compromise this host. But the full code that runs the remainder of the attack is sitting on the attackers machine, and unless we are fully compromised, we will never know the full intention of the extent of the code in that attack, making the creation of a "signature" for that attack much more complicated.

The web site dedicated to the Snort IDS, www.snort.org, claims it updates its' signature files every thirty minutes as seen specifically at the site: http://www.snort.org/downloads.html#1.14. However, this simply means that if all the above criteria are met, that within a thirty minute window, the file is put on the web site. But how long does it take to meet all the criteria above for intrusion detection systems?

As a new exploit is found it takes some time to discover it in the wild for all the reasons mentioned above. As a for instance, the Nimda code was discovered not because a receiver found a suspicious email attachment and submitted it for review, but because the code set off flags in the total number of port 80 scans as it began to propagate so much

that its' traffic began to affect normal Internet patterns at roughly 13:00 GMT on September 18th, 2001 as stated by http://www.incidents.org/react/nimda.pdf.

Most anti-virus software companies did not release a signature for the file until late in the evening of September 18th, 2001. According to F-Secure's (a virus software based in Finland) web site at http://www.europe.f-secure.com/v-descs/nimda.shtml: "F-Secure anti-virus detects the worm with updates released on September 18th, 2001 19:20 EET. Disinfection was added in the updates from September 19th, 2001 17:12 EET."

Accounting for time zone differences, this gave Nimda a 5-hour jump on virus definitions, or 106,000 hosts as extrapolated from the graph "Hourly Distinct Source IP's Probing Port 80 Addresses" at http://www.incidents.org/react/ nimda.pdf. In this country, Network Associates' Macafee Anti-Virus shield product released it's virus definition signature on September 20, 2001, a full 2 days after the discovery (http://vil.nai.com/vil /virusSummary.asp?virus_k=99209).

What's worse is thinking about what would have happened if Nimda didn't spread so fast? A big reason that a signature for Nimda was created so quickly was because of its' rapid propagation. If it had been a little more "stealthy" and not have changed Internet patterns so dramatically so quickly, maybe the end infections would have been much higher?

As stated by Herve Debar of the IBM Zurich Research Library in relation to signature based or as he called them "knowledge based" intrusion detections systems at http://www.sans.org/newlook/resources/IDFAQ/knowledge_based.htm, "Drawbacks include the difficulty of gathering the required information on the known attacks and keeping it up to date with new vulnerabilities and environments. Maintenance of the knowledge base of the intrusion detection system requires careful analysis of each vulnerability and is therefore a time-consuming task. Knowledge-based approaches also have to face the generalization issue. Knowledge about attacks is very focused, dependent on the operating system, version, platform, and application. The resulting intrusion detection tool is therefore closely tied to a given environment. Also, detection of insider attacks involving an abuse of privileges is deemed more difficult because no vulnerability is actually exploited by the attacker."

It is clear that the time lag in realizing an attack is taking place to the production of a signature for it can be devastating. But once it is found, the creation of a signature is equally inefficient.

Nimda has four distinct methods of propagation as mentioned in the following article http://www.cert.org/advisories/CA-2001-26.html:

"This new worm appears to spread by multiple mechanisms:

- from client to client via email
- from client to client via open network shares
- from web server to client via browsing of compromised web sites

- from client to web server via active scanning for and exploitation of various Microsoft IIS 4.0 / 5.0 directory traversal vulnerabilities (VU#111677 and CA-2001-12)
- from client to web server via scanning for the back doors left behind by the "Code Red II" (IN-2001-09), and "sadmind/IIS" (CA-2001-11) worms

The worm modifies web documents (e.g., .htm, .html, and .asp files) and certain executable files found on the systems it infects, and creates numerous copies of itself under various file names."

With so many different propagation methods running through tcp port 80 for web traffic, tcp port 25 for smtp traffic, and the associated code that is attached to each of these possible ports, the detecting of this specific intrusion cannot be narrowed into a single "signature".

In short, this means that the intrusion "signature" cannot be a simple port match, like the IDS Snort rule:

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

as found at www.snort.com/downloads.html#1.14 to detect a possible Squid Proxy scan. It can not be a simple one line content string in the payload like in the IDS Snort rule:

> "alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB-MISC iPlanet GETPROPERTIES attempt"; content:"GETPROPERTIES"; offset:0; depth:13; classtype:attempted-admin; sid:1050; rev:1;)

as found at www.snort.com/downloads.html#1.14 to detect the iPlanet attack.

Instead, there are many rules that have to be created to identify Nimda in the Snort IDS as found at www.snort.com/downloads.html#1.14.

> "alert tcp $EXTERNAL_NET any -> $HOME_NET 139 (msg:"NETBIOS nimda .eml"; content:"|00|E|00|M|00|L"; flags:A+; classtype:bad-unknown; reference:url,www.datafellows.com/v-descs/nimda.shtml; sid:1293; rev:2;)"

> "alert tcp $EXTERNAL_NET any -> $HOME_NET 139 (msg:"NETBIOS nimda .nws"; content:"|00|N|00|W|00|S"; flags:A+; classtype:bad-unknown; reference:url,www.datafellows.com/v-descs/nimda.shtml; sid:1294; rev:2;)"

> "alert tcp $EXTERNAL_NET any -> $HOME_NET 139 (msg:"NETBIOS nimda RICHED20.DLL"; content:"R|00|I|00|C|00|H|00|E|00|D|00|2|00|0"; flags:A+; classtype:bad-unknown; reference:url,www.datafellows.com/v-descs/nimda.shtml; sid:1295; rev:2;)"

Instead of copying all the rules that help to detect Nimda, I have demonstrated only the alerts that were added that specifically mention Nimda. As mentioned above, Nimda takes advantages of holes left by CodeRed II and sadmind/IIS, and each of those has their own "signatures" that would be flagged if Nimda attempted to use those back doors. Therefore these three "signatures" plus the signatures of "CodeRedII" and sadmind/IIS all are necessary in order to fully identify the Nimda exploit and all its' propagation methods.

To further demonstrate this, the Cisco IDS called Cisco Secure protects against Nimda with the following rules:

"The following Cisco IDS Host Sensor rules prevent the Nimda worm from succeeding:

- IIS Directory Traversal (four rules)
- IIS Directory Traversal and Code Execution (four rules)
- IIS Double Hex Encoding Directory Traversal (four rules)"

As stated in the document http://www.cisco.com/warp/public/63/nimda-ids.pdf

Although the rules are extensive for Nimda, they were easy to create and could have been much more difficult. Nimda, unlike other attacks in the past, attempted every possible combination of exploits in its' arsenal on every host. What if it was written more carefully, and only launched the exploits that were specific to that host. If this were the case, the rules to detect Nimda would have taken a far greater amount of time to create.

In addition to the sheer number of rules that need to be applied and managed to detect the possible intrusion, there is the sheer number of logs to look through as the intrusions are automated and the attacks become rampant on the Internet. "Currently, due to the successful propagation of this worm throughout the Internet, you may experience a high volume of alerts from your intrusion detection systems, as well as possible network bandwidth degradation. The IDS alert volume can be on the order of up to 6000 alerts an hour. " - stated by Counterpane Internet Security at http://www.counterpane.com/alert-nimda.html

As an intrusion analyst, how can you keep up with so many alerts? The human equation in intrusion detection adds to the complexity of finding the true attacks hidden in a sea of false positives, or of true attacks on protected assets. The "noise" created by these Internet worms and viruses, degrades the true attack signals all the time, especially during rampant attack periods. The fact that to detect things like Nimda requires so many specific "signatures" and the that the attack itself normally triggers all these signatures on every attempt bogs down the analyst in a meaningless sea of alert reports.

Clearly this is inefficient.

So what else could be done instead of matching specific signatures? Herve Debar refers to a different type of approach called "behavior-based" files. Here is what he said about the advantages of this type of IDS in the following web site http://www.sans.org/newlook/resources/IDFAQ/behavior_based.htm:

"Advantages of behavior-based approaches are that they can detect attempts to exploit new and unforeseen vulnerabilities. They can even contribute to the (partially) automatic discovery of these new attacks. They are less dependent on operating system-specific mechanisms. They also help detect 'abuse of privileges' types of attacks that do not actually involve exploiting any security vulnerability. In short, this is the paranoid approach: Everything which has not been seen previously is dangerous."

This method has its' disadvantages too, as Herve Debar from the same article, (http://www.sans.org/newlook/resources/IDFAQ /behavior_based.htm) states:

"The high false alarm rate is generally cited as the main drawback of behavior-based techniques because the entire scope of the behavior of an information system may not be covered during the learning phase. Also, behavior can change over time, introducing the need for periodic online retraining of the behavior profile, resulting either in unavailability of the intrusion detection system or in additional false alarms. The information system can undergo attacks at the same time the intrusion detection system is learning the behavior. As a result, the behavior profile contains intrusive behavior, which is not detected as anomalous."

Clearly, this battle of signature based or behavior based intrusion detection technology will be one of the defining factors of how effective and efficient intrusion detection is in the battle to protect corporations over time. I look forward to seeing a time when an IDS is capable of detecting attacks before they happen based on behaviors of other attacks, but is also capable of only showing true attacks and not false positives. A day I'm sure all analysts dream for.

References:

www.snort.org
http://www.snort.org/downloads.html#1.14
http://www.incidents.org/react/nimda.pdf
http://www.europe.f-secure.com/v-descs/nimda.shtml
http://vil.nai.com/vil/virusSummary.asp?virus_k=99209
http://www.sans.org/newlook/resources/IDFAQ/knowledge_based.htm
http://www.cert.org/advisories/CA-2001-26.html
http://www.cisco.com/warp/public/63/nimda-ids.pdf
http://www.counterpane.com/alert-nimda.html
http://www.sans.org/newlook/resources/IDFAQ/behavior_based.htm

Part II Network Detects

Detect 1

[**] IDS177/netbios-name-query [**]
02/26-03:01:24.326077 63.106.48.202:137 -> Target IP:137
UDP TTL:118 TOS:0x0 ID:29798 IpLen:20 DgmLen:78
Len: 58
length = 50
000 : 5E 34 00 10 00 01 00 00 00 00 00 00 20 43 4B 41 ^4..... CKA
010 : 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAA
020 : 41 41 41 41 41 41 41 41 41 41 41 41 41 00 00 21 AAAAAAAAAAAAA..!
030 : 00 01   ..

[**] IDS177/netbios-name-query [**]
02/26-03:01:25.827974 63.106.48.202:137 -> Target IP:137
UDP TTL:118 TOS:0x0 ID:30054 IpLen:20 DgmLen:78
Len: 58
length = 50
000 : 5E 36 00 10 00 01 00 00 00 00 00 00 20 43 4B 41 ^6..... CKA
010 : 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAA
020 : 41 41 41 41 41 41 41 41 41 41 41 41 41 00 00 21 AAAAAAAAAAAAA..!
030 : 00 01   ..

[**] IDS177/netbios-name-query [**]
02/26-03:01:27.328233 63.106.48.202:137 -> Target IP:137
UDP TTL:118 TOS:0x0 ID:30310 IpLen:20 DgmLen:78
Len: 58
length = 50
000 : 5E 38 00 10 00 01 00 00 00 00 00 00 20 43 4B 41 ^8..... CKA
010 : 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 AAAAAAAAAAAAAAAA
020 : 41 41 41 41 41 41 41 41 41 41 41 41 41 00 00 21 AAAAAAAAAAAAA..!
030 : 00 01   ..

1.1 Source of Trace.

It was taken from the SANS.org web site, specifically the address: www.sans.org/y2k/022701-1600.htm.

1.2. Detect was generated by:

This scan was logged by the Snort IDS. I was not the individual that took it, and am unsure of the setup of the specific machine that captured it, but am able to tell which specific IDS was used by the format.

1.3. Probability the source address was spoofed:

The chance of the source IP address being spoofed is very low. The scans all come from the same source address, so there is no attempt at decoying the scan with a variety of spoofed IP's. The packets also come in rapid succession from that host to multiple destination hosts all at the same port. These things together with the fact that this is a recon mission looking for responses not a worm type UDP exploit, make a strong case that the source IP address is real.

1.4. Description of attack:

The Snort IDS flagged this attack as a reconnaissance mission. It is a simple recon mission, which is under investigation for a CVE number CAN-1999-0621. Since it isn't a specific exploit just a recon mission, there are many attacks that could be used if the recon returns information.

1.5. Attack mechanism:

The attack works by collecting information about a machine by sending a NETBIOS name query packet. If the firewall is open, this request can produce a list of machines, domains, users etc. from a windows host to be used later in any number of attacks (such as the leaves worm). It is clearly a recon mission.

http://www.whitehats.com/IDS/177 had this to say about this type of scan:

"This event indicates a standard netbios name table retrieval query. Windows machines often exchange these queries as a part of the filesharing protocol to determine NetBIOS names when only IP addresses are known. An attacker could use this same query to extract useful information such as workstation name, domain, and users who are currently logged in."

1.6. Correlations:

This attack is a common method of reconnaissance. Although I have no other information about this specific network or other traces from it, it is assumed that if this were a successful recon mission, there would be attacks launched at a later date from this machine or others that are owned by the same attacker.

Some URL's that point to the same type of attack are:

http://archives.neohapsis.com/archives/snort/2000-01/0222.html
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids177&view=event

1.7. Evidence of active targeting:

Since the log shows only this one machine doing the querying and only one host being queried with different forms of the NETBIOS name queries, it would appear as active targeting. However, there is no time lapse between queries and yet the packet IDs are jumping up quite dramatically, making it a clear case that this is probably not active targeting, but a machine poking many machines for NETBIOS names.

1.8. Severity:

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

	Description	**Rating**
Criticality	Because this type of request could gather information about the whole Windows network, it is not about one machine.	4
Lethality	This is just a request for information, and doesn't produce an immediate threat	3
System Countermeasures	If the port is open, and it is a Windows host, it will respond to this request. No patch stops the response to this query, as it is part of the windows operating system functionality.	2
Network Countermeasures	Since this is launched at a specific machine we can guess that it the port on the firewall to this machine is open	1

Severity = 4 (+) 3 – 2 (+) 1
Severity = 4

1.9. Defensive recommendation:

Port 137 has no reason for access from the Internet. The firewall should have a rule blocking port 137 traffic as well as the 139 and 445 ports that are all attributed to Windows networking functionality.

1.10. Multiple choice test question:

What is the most probable reason for the change in the first two characters of the payload in these packets?
a) A crafted packet designed to find an overflow.
b) Different types of NetBios name queries.
c) A limit in the IDS that this isn't a true NetBios name query, but instead another type of attack.
d) This is actually a port 137 Denial of Service attack.
Answer: b

Detect 2

```
Apr  2 08:39:52 195.223.184.81:2701 -> a.b.c.170:515 SYN ******S*
Apr  2 08:39:55 195.223.184.81:3413 -> a.b.e.106:515 SYN ******S*
Apr  2 08:39:56 195.223.184.81:3714 -> a.b.f.152:515 SYN ******S*
Apr  2 08:39:56 195.223.184.81:3716 -> a.b.f.154:515 SYN ******S*
Apr  2 08:39:56 195.223.184.81:3752 -> a.b.f.190:515 SYN ******S*
Apr  2 08:39:56 195.223.184.81:3754 -> a.b.f.192:515 SYN ******S*
Apr  2 08:39:56 195.223.184.81:3814 -> a.b.f.252:515 SYN ******S*

Apr  2 08:39:50 hostka portsentry[430]: attackalert: Connect from host:
  195.223.184.81/195.223.184.81 to TCP port: 515
Apr  2 08:44:47 hostka portsentry[430]: attackalert: Connect from host:
  195.223.184.81/195.223.184.81 to TCP port: 515
Apr  2 08:44:47 hostka portsentry[430]: attackalert: Connect from host:
  195.223.184.81/195.223.184.81 to TCP port: 515
Apr  2 08:44:48 hostka portsentry[430]: attackalert: Connect from host:
  195.223.184.81/195.223.184.81 to TCP port: 515
Apr  2 08:44:49 hostka portsentry[430]: attackalert: Connect from host:
  195.223.184.81/195.223.184.81 to TCP port: 515
Apr  2 08:44:50 hostka portsentry[430]: attackalert: Connect from host:
  195.223.184.81/195.223.184.81 to TCP port: 515
Apr  2 08:44:51 hostka portsentry[430]: attackalert: Connect from host:
  195.223.184.81/195.223.184.81 to TCP port: 515
```

Apr 2 08:44:52 hostka portsentry[430]: attackalert: Connect from host:
195.223.184.81/195.223.184.81 to TCP port: 515
Apr 2 08:44:53 hostka portsentry[430]: attackalert: Connect from host:
195.223.184.81/195.223.184.81 to TCP port: 515
Apr 2 08:44:53 hostka portsentry[430]: attackalert: Connect from host:
195.223.184.81/195.223.184.81 to TCP port: 515
Apr 2 08:44:54 hostka portsentry[430]: attackalert: Connect from host:
195.223.184.81/195.223.184.81 to TCP port: 515

Apr 2 08:44:47 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:1675 -> a.b.c.225:515
Apr 2 08:44:48 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:1706 -> a.b.c.225:515
Apr 2 08:44:48 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:1745 -> a.b.c.225:515
Apr 2 08:44:49 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:1790 -> a.b.c.225:515
Apr 2 08:44:50 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:1837 -> a.b.c.225:515
Apr 2 08:44:51 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:2194 -> a.b.c.225:515
Apr 2 08:44:52 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:2276 -> a.b.c.225:515
Apr 2 08:44:53 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:2336 -> a.b.c.225:515
Apr 2 08:44:54 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:2399 -> a.b.c.225:515
Apr 2 08:44:55 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:2748 -> a.b.c.225:515
Apr 2 08:44:55 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:2809 -> a.b.c.225:515

2.1. Source of Trace.

It was taken from the SANS.org web site, specifically the address: http://www.sans.org/y2k/040901-1500.htm.

2.2 Detect was generated by:

Three different machines logged this attack all running at the same time capturing packets. The first machine is running either tcpdump or windump. The second is running the Psionic Software's PortSentry IDS. The third is running the Snort IDS. I was not the individual that setup the machines, and am unsure of the specific setup of any of the machines that captured the data, but am able to tell the specific IDSs that were used by the format.

2.3. Probability the source address was spoofed:

As an attempt for a buffer overflow attack the chance of a spoofed IP address is low. The initial probe above captured a TCP port 515 syn scan to determine if the port is open for the known Unix LPR or Windows print buffer overflow attack. The attack was then

launched from that same IP address. As this attack requires a TCP three-way handshake to complete, the source IP could not have been spoofed.

2.4. Description of attack:

The tcpdump file first identifies a port scan for the TCP port 515 open. PortSentry flagged this as a connection attempt. PortSentry captured the search for port 515 hosts because the machine it is running on has port 515 blocked, and so therefore flagged this as a connection attempt. It is assumed (even though it is not in the trace) that the port scan for port 515 found an open machine on a.b.c.225. An attack was then launched, and Snort flagged that attack. The Snort IDS flagged this attack as a buffer overflow because the three-way tcp handshake completed, and the attack launched matched a signature for known vulnerabilities of TCP port 515. There are several CVE numbers that reference port 515 exploits. Without the full payload or more information about the type of machine this is that is being attacked, it is not possible to put down a specific CVE number.

2.5. Attack mechanism:

The attack sends a packet to that listening port 515 with a padded packet designed to overflow the buffer. Following that padding are commands to gain root access to the machine since after the buffer is filled, code will overwrite the variable buffer space and be able to be executed the next time the variable is called.

2.6. Correlations:

Without the full payload, it is difficult to tell exactly what exploit is being launched. However, there are many exploits related to port 515.

Some URL's that point to similar types of attacks are:

http://www.whitehats.com/info/IDS457
http://www.whitehats.com/info/IDS456
http://www.cert.org/advisories/CA-2000-22.html
http://www.securityfocus.com/bid/3252
http://www.securityfocus.com/bid/2894

2.7. Evidence of active targeting:

It appears as though this is a random probe looking for port 515, as the first tcpdump file shows many different random addresses to the same port in a short period of time. Once found, an automated tool launches the attack as noted in the PortSentry and Snort logs below that. It is assumed that it is some form of automated attack since the time between port scans and attack launches, is short. Therefore, this does not look like active targeting.

2.8. Severity:

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

	Description	Rating
Criticality	Since this attack was launched at the same IP address, it isn't random. Without more details of the environment, this is a difficult measure. We'll assume for the rating that this is a critical server.	4
Lethality	Since this is a buffer overflow with root access it is definitely very lethal	5
System Countermeasures	Since there is no evidence of any packet going back to the attacker, and there are multiple attempts at the same overflow from different source ports, it is assumed that this machine is well patched	4
Network Countermeasures	Since there are several IDS in place, it is assumed that there is a proper firewall, this is an assumption in this network as we don't have the details about the network	4

Severity = 4 (+) 5 – 4 (+) 4
Severity = 1

2.9. Defensive recommendation:

Since TCP port 515 is both a Unix LPD port and a Windows 2000 printer port, the firewall should be blocking this port form the outside world. In this scenario, since this is a TCP Syn packet sent to multiple hosts, and there are actual attacks sent to port 515 it is assumed that the firewall is not blocking this port to this specific host (a.b.c.225). It is recommended to block that port.

2.10. Multiple choice test question:

```
Apr  2 08:44:47 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:1675 -> a.b.c.225:515
Apr  2 08:44:48 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:1706 -> a.b.c.225:515
Apr  2 08:44:48 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:1745 -> a.b.c.225:515
Apr  2 08:44:49 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:1790 -> a.b.c.225:515
```

Apr 2 08:44:50 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:1837 -> a.b.c.225:515
Apr 2 08:44:51 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:2194 -> a.b.c.225:515
Apr 2 08:44:52 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:2276 -> a.b.c.225:515
Apr 2 08:44:53 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:2336 -> a.b.c.225:515
Apr 2 08:44:54 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:2399 -> a.b.c.225:515
Apr 2 08:44:55 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:2748 -> a.b.c.225:515
Apr 2 08:44:55 hostka snort: EXPLOIT x86 NOOP: 195.223.184.81:2809 -> a.b.c.225:515

What in the above network trace makes a case that it is not the x85 Exploit that Snort flagged but instead a denial of service attempt?

a) The fact that it is coming from different port numbers from the attacking machine making it look like the attack isn't working, so it is probably something else.
b) The short intervals of time between each launch.
c) The fact that the port numbers are jumping upward but quickly in such a short period of time, making it look like this machine is DOSing many machines simultaneously.
d) All of the above.

Answer: d

Detect 3

Jan 8 12:28:20 hosty portsentry[594]: attackalert: Connect from host:
61.141.205.214/61.141.205.214 to TCP port: 3128
Jan 8 12:29:21 hostj portsentry[481]: attackalert: Connect from host:
61.141.205.214/61.141.205.214 to TCP port: 3128
Jan 8 12:30:22 hostm portsentry[455]: attackalert: Connect from host:
61.141.205.214/61.141.205.214 to TCP port: 3128

3.1. Source of Trace.

It was taken from the SANS.org web site, specifically the address:
http://www.sans.org/y2k/011701.htm

3.2 Detect was generated by:

The detect was generated from Psionic Software's PortSentry IDS. Since it was not my detect, I don't know any more about the specifics of the machine.

3.3. Probability the source address was spoofed:

The probability that this attackers IP address is spoofed is slim. There are not a lot of packets from different hosts showing that it isn't an attempt to hide the true IP with a

variety of different decoy IP addresses also scanning the network. Because the alerts come in rapid succession from the same host to different hosts on the internal network to one specific destination port, this gives good reason to believe that it is a true IP address. Normally the true IP address of the machine doing the scan must be able to receive the response to utilize the results of the scan. Since this is a scan for a proxy server, or more specifically a squid proxy, the point is to find the addresses that respond to use them to later hide the source IP address. The source IP must be real to get the response. While it is true that the source IP address might be spoofed as with the nmap feature known as zombie host (using the sI switch), the facts above show this as unlikely.

3.4. Description of attack:

This is another type of scan for a specific port, this one is looking for a squid proxy. Since this is just a scan, there is no specific CVE number associated with the scan. There are no known exploits at this time on Squid Proxies so there are no CVE numbers to quote on the possible attacks either. More likely the search is not to exploit a machine, but to use it to protect the attackers identity.

3.5. Attack mechanism:

The attack really would come after this scanning found a host that would respond. What typically happens is if a machine were to respond then that machine would be used in the future to hide the source IP address of the attacker to launch any number of attacks. Chances are, this machine by itself is probably safe from being destroyed or defaced, but has a STRONG possibility of being used to launch DOS or DDOS or other types of attacks.

3.6. Correlations:

Since this is just a scan, an actual exploit is not documented yet for a squid proxy. However, there are many uses for a squid proxy and some URL's that point to those are:

http://archives.neohapsis.com/archives/incidents/2000-03/0140.html
http://www.securityfocus.com/infocus/1508

3.7. Evidence of active targeting:

Since the machines in the scan are all part of the same network and the source is the same and the times are short, it is my opinion that it is "trawling" not active targeting on this network. Since the IP's are somewhat in order, it looks as though this is an automated tool just probing the network. The missing machines probably either don't have this port enabled and didn't respond, or are blocked at the firewall.

3.8. Severity:

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

	Description	Rating
Criticality	Since no machine has responded, it is assumed that there are no squid proxy machines, and if there were, it would only be a proxy server not a critical DNS server.	2
Lethality	Since there are no known squid proxy exploits, it is assumed that this is a scan to find a way to hide the source IP address, not root the machine.	1
System Countermeasures	Since no machine has responded it is assumed that no machine has a squid proxy enabled.	5
Network Countermeasures	Since no machine has responded it is assumed that the firewall is blocking this port.	5

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

3.9. Defensive recommendation:

It looks as though the firewall is not blocking the scans to these machines, as PortSentry is a host based product that sits inside the firewall. The first recommendation is to block the Squid Proxy ports, unless for some reason this proxy is a required. If it is a requirement, perhaps limiting the firewall to permit only the IP addresses that need connectivity.

3.10. Multiple choice test question:

What is the significance of someone finding a squid proxy and using it on your network?

a) There are many known exploits on squid proxies.
b) Squid proxies control your internal machines access to the Internet, and can therefore take your internal machines off line.
c) The squid proxy could be used by an attacker to hide their IP address as the source IP from attacks they launch through you.
d) All of the above.

Answer: c

Detect 4

C:\Downloads\Windump>windump -v -r c:\dumpfile2.txt src host 209.53.29.123 or dst host 209.53.29.123

08:28:43.992526 209.53.29.123.22 > x.x.27.83.22: SF 372925843:372925843(0) win 1028 (ttl 28, id 39426)

08:28:44.033035 209.53.29.123.22 > x.x.27.85.22: SF 372925843:372925843(0) win 1028 (ttl 27, id 39426)

08:28:44.093111 209.53.29.123.22 > x.x.27.88.22: SF 372925843:372925843(0) win 1028 (ttl 27, id 39426)

08:28:44.113027 209.53.29.123.22 > x.x.27.89.22: SF 372925843:372925843(0) win 1028 (ttl 28, id 39426)

08:28:44.132617 209.53.29.123.22 > x.x.27.90.22: SF 372925843:372925843(0) win 1028 (ttl 28, id 39426)

08:28:44.153018 209.53.29.123.22 > x.x.27.91.22: SF 372925843:372925843(0) win 1028 (ttl 28, id 39426)

08:28:44.155339 x.x.27.91.22 > 209.53.29.123.22: S 1146407057:1146407057(0) ack 372925844 win 5840 <mss 1460> (DF) (ttl 63, id 0)

08:28:44.172727 209.53.29.123.22 > x.x.27.92.22: SF 372925843:372925843(0) win 1028 (ttl 28, id 39426)

08:28:44.192516 209.53.29.123.22 > x.x.27.93.22: SF 372925843:372925843(0) win 1028 (ttl 28, id 39426)

08:28:44.212110 209.53.29.123.22 > x.x.27.94.22: SF 372925843:372925843(0) win 1028 (ttl 28, id 39426)

08:28:44.232330 209.53.29.123.22 > x.x.27.95.22: SF 372925843:372925843(0) win 1028 (ttl 28, id 39426)

08:28:44.253515 209.53.29.123.22 > x.x.27.91.22: R 372925844:372925844(0) win 0 (ttl 241, id 2548)

4.1. Source of Trace.

This trace was captured on my corporate network's lab (a place where we have full control to test things, not build fake scans).

4.2. Detect was generated by:

For the previous network scan, I used four laptop machines. To see the strength of the firewall, I put two machines between the inside of the router and the outside of the

firewall on a hub and two machines between the inside of the firewall and the first corporate switch on another hub. This gave me the ability to see what was attempting to come in through the T-1 line, and what was succeeding. Overall my firewall blocked almost none of what Snort considered to be potential attacks. However, as it turned out, almost none of them were actual attacks.

Two of the machines were running Snort version 1.8.1, PHP and Acid using configuration files downloaded on 10/1/01, one outside, one inside the firewall. This gave me a nice GUI to view my alerts.

The other two machines were running Windump and logging all packets to a text file on the local drive. This gave me the actual packets that I demonstrated above.

4.3. Probability the source address was spoofed:

There is a slim chance that the source IP was spoofed as a TCP 3 way handshake was launched immediately following the scan (see below). Since a 3 way handshake requires communication with the true host, then the source IP address was probably not spoofed.

4.4. Description of attack:

The attack is a scan to port 22 only. It is a reconnaissance mission poking for machines to respond in some way to a Syn or Fin packet. It is assumed that this port scan is looking for machines that are vulnerable to a specific attack as the ones listed below.

4.5. Attack mechanism:

This attack is an obvious attempt to find hosts that are responding in some way to port 22. The attacker is somewhat randomly poking into networks by sending a TCP packet with both the Syn and Fin flags set. Since these flags do not occur normally together, an attacker can learn quite a bit from a response from the hosts that do respond, and whether they send back a reset or and acknowledgement of the Syn or Fin.

The packet also contains a Fin flag set to attempt to avoid some older intrusion detection systems and firewalls.

In addition, because a router's ACL only matches port numbers, any packet, Syn, Fin or Christmas tree will be allowed through if that port is open on the router. A statefull inspection firewall should not allow a Fin packet into the network until a connection has been established. However, because the Syn flag is also set, some firewalls see this as an initiation packet, and allow it through the network. Furthermore, some older IDS units will see this as a Syn packet and also believe it to be an initiation of a communication and not set off an alert.

In short, the attack mechanism is to learn about machines by sending in a packet and investigating the response.

4.6. Correlations:

There are again many different types of exploits that could be used if in fact a machine is listening at TCP port 22.

http://razor.bindview.com/publish/advisories/adv_ssh1crc.html
http://www.securityfocus.com/archive/1/11018
http://www.securityfocus.com/archive/1/7718
http://www.securityfocus.com/archive/1/8398

4.7. Evidence of active targeting:

In doing multiple searches on the data collected, this particular IP address only shows up these 10 times scanning this network. Since our IP block is much larger than this set of 10 IP addresses, and there is another IP block that is directed to this same router, it is my estimation that the attacker was looking for something in our range, not randomly searching the net.

More specifically, when the attacker got a response, the scanning stopped.

This network address was not seen anywhere in the 7 day period monitored, except for the port 22 scans and this trace that happened immediately following the scan:

08:28:44.594641 209.53.29.123.1740 > x.x.27.91.22: S 3079849825:3079849825(0) win 32120 <mss 1460,sackOK,timestamp 542107866 0,nop,wscale 0> (DF) (ttl 50, id 2551)

08:28:44.595378 x.x.27.91.22 > 209.53.29.123.1740: S 1134898389:1134898389(0) ack 3079849826 win 5792 <mss 1460,sackOK,timestamp 135432545 542107866,nop,wscale 0> (DF) (ttl 63, id 0)

08:28:44.693959 209.53.29.123.1740 > x.x.27.91.22: . ack 1 win 32120 <nop,nop,timestamp 542107876 135432545> (DF) (ttl 50, id 2552)

08:28:44.707555 x.x.27.91.22 > 209.53.29.123.1740: P 1:24(23) ack 1 win 5792 <nop,nop,timestamp 135432557 542107876> (DF) (ttl 63, id 42429)

08:28:44.806454 209.53.29.123.1740 > x.x.27.91.22: . ack 24 win 32120 <nop,nop,timestamp 542107887 135432557> (DF) (ttl 50, id 2553)

08:28:44.837053 209.53.29.123.1740 > x.x.27.91.22: F 1:1(0) ack 24 win 32120 <nop,nop,timestamp 542107890 135432557> (DF) (ttl 50, id 2555)

08:28:44.837776 x.x.27.91.22 > 209.53.29.123.1740: F 24:24(0) ack 2 win 5792 <nop,nop,timestamp 135432570 542107890> (DF) (ttl 63, id 42430)

08:28:44.936550 209.53.29.123.1740 > x.x.27.91.22: . ack 25 win 32120 <nop,nop,timestamp 542107900 135432570> (DF) (ttl 50, id 2558)

Since traffic was initiated immediately following the port 22 scan, it is obvious that whomever the attacker was, they got what they were looking for, a connection. However, from the monitoring above, there was only a small amount of information passed from the targeted machine and no further communication. This is probably a reconnaissance mission, although an attempt was made to communicate to that host. This information was probably an attempt to gain version information, and it wasn't the version being sought after, since

a. No other information was sent from the attacking host.
b. It did not trigger an event in the Snort log's.
c. It did not send out any other information back to the attacking host other than an acknowledgement
d. It did not then begin showing signs of compromise
e. There was no log of this SSH connection on the machine in question

In conclusion, as seen above, the attacker immediately used the information, making a strong case that the attack was not random, but targeted and carefully watched. However, it is a crafted packet attack in that the packet ID is the same even though the IP address changes. While it is a targeted attack specific to our network today, it looks as though it could be an automated tool that is doing the work, leading me to believe that the attacker will use this tool again elsewhere.

4.8. Severity:

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

	Description	**Rating**
Criticality	The machine is only a storage facility for non-critical data.	3
Lethality	Not high as it appears to be simply a port scan, no damage from the scan alone.	2
System Countermeasures	The machine is well patched and all other ports are closed.	3
Network Countermeasures	While there is a firewall in the network, it doesn't protect against this scan.	1

Severity = 3 (+) 2 – 3 (+) 1
Severity = 1

4.9. Defensive recommendation:

Consider purchasing a more up to date firewall that does not allow Sin and Fin together. In this case, since the packet had both flags set, it was able to avoid detection by the firewall and a connection was established.

4.10. Multiple choice test question:

C:\Downloads\Windump>windump -v -r c:\dumpfile2.txt src host 209.53.29.123 or dst host 209.53.29.123

08:28:43.992526 209.53.29.123.22 > x.x.27.83.22: SF 372925843:372925843(0) win 1028 (ttl 28, id 39426)

08:28:44.033035 209.53.29.123.22 > x.x.27.85.22: SF 372925843:372925843(0) win 1028 (ttl 27, id 39426)

08:28:44.093111 209.53.29.123.22 > x.x.27.88.22: SF 372925843:372925843(0) win 1028 (ttl 27, id 39426)

08:28:44.113027 209.53.29.123.22 > x.x.27.89.22: SF 372925843:372925843(0) win 1028 (ttl 28, id 39426)

08:28:44.132617 209.53.29.123.22 > x.x.27.90.22: SF 372925843:372925843(0) win 1028 (ttl 28, id 39426)

08:28:44.153018 209.53.29.123.22 > x.x.27.91.22: SF 372925843:372925843(0) win 1028 (ttl 28, id 39426)

What evidence of packet crafting is noticed in the network trace above?

a) The time to lives are all the same.
b) The window sizes are all the same.
c) The packet ID's are all the same.
d) The source ports are all the same.
Answer: c

Detect 5

[**] spp_http_decode: IIS Unicode attack detected [**]

04/12-05:44:29.537613 213.121.247.193:61522 -> x.x.x.23:80

TCP TTL:41 TOS:0x0 ID:2938 IpLen:20 DgmLen:289 DF

AP Seq: 0xEF818D34 Ack: 0x844F3E92 Win: 0x7D78 TcpLen: 32

TCP Options (3) => NOP NOP TS: 15433327 0

47 45 54 20 2F 6D 73 61 64 63 2F 2E 2E 25 63 30 GET /msadc/..%c0

25 61 66 2E 2E 2F 2E 2E 25 63 30 25 61 66 2E 2E %af../..%c0%af..

2F 2E 2E 25 63 30 25 61 66 2E 2E 2F 77 69 6E 6E /..%c0%af../winn

```
74 2F 73 79 73 74 65 6D 33 32 2F 63 6D 64 2E 65  t/system32/cmd.e
78 65 3F 2F 63 2B 64 69 72 2B 63 3A 5C 20 48 54  xe?/c+dir+c:\ HT
54 50 2F 31 2E 30 0D 0A 56 69 61 3A 20 31 2E 30  TP/1.0..Via: 1.0
20 50 72 6F 78 79 3A 33 31 32 38 20 28 53 71 75   Proxy:3128 (Squ
69 64 2F 32 2E 33 2E 53 54 41 42 4C 45 31 29 0D  id/2.3.STABLE1).
0A 58 2D 46 6F 72 77 61 72 64 65 64 2D 46 6F 72  .X-Forwarded-For
3A 20 36 32 2E 34 31 2E 33 38 2E 31 30 0D 0A 48  : 62.41.38.10..H
6F 73 74 3A 20 31 34 30 2E 31 37 38 2E 33 33 2E  ost: x.x.x.
32 33 0D 0A 43 61 63 68 65 2D 43 6F 6E 74 72 6F  23..Cache-Contro
6C 3A 20 6D 61 78 2D 61 67 65 3D 32 35 39 32 30  l: max-age=25920
30 0D 0A 43 6F 6E 6E 65 63 74 69 6F 6E 3A 20 6B  0..Connection: k
65 65 70 2D 61 6C 69 76 65 0D 0A 0D 0A           eep-alive....
```

=+=
+

```
[**] spp_http_decode: IIS Unicode attack detected [**]
04/12-05:44:29.589223 213.121.247.193:61528 -> x.x.x.23:80
TCP TTL:39 TOS:0x0 ID:2943 IpLen:20 DgmLen:292 DF
***AP*** Seq: 0xEFCCA502  Ack: 0x8450CA83  Win: 0x7D78  TcpLen: 32
TCP Options (3) => NOP NOP TS: 15433329 0
47 45 54 20 2F 5F 76 74 69 5F 62 69 6E 2F 2E 2E  GET /_vti_bin/..
25 63 30 25 61 66 2E 2E 2F 2E 2E 25 63 30 25 61  %c0%af../..%c0%a
66 2E 2E 2F 2E 2E 25 63 30 25 61 66 2E 2E 2F 77  f../..%c0%af../w
69 6E 6E 74 2F 73 79 73 74 65 6D 33 32 2F 63 6D  innt/system32/cm
64 2E 65 78 65 3F 2F 63 2B 64 69 72 2B 63 3A 5C  d.exe?/c+dir+c:\
20 48 54 54 50 2F 31 2E 30 0D 0A 56 69 61 3A 20   HTTP/1.0..Via:
31 2E 30 20 50 72 6F 78 79 3A 33 31 32 38 20 28  1.0 Proxy:3128 (
53 71 75 69 64 2F 32 2E 33 2E 53 54 41 42 4C 45  Squid/2.3.STABLE
31 29 0D 0A 58 2D 46 6F 72 77 61 72 64 65 64 2D  1)..X-Forwarded-
46 6F 72 3A 20 36 32 2E 34 31 2E 33 38 2E 31 30  For: 62.41.38.10
0D 0A 48 6F 73 74 3A 20 31 34 30 2E 31 37 38 2E  ..Host: x.x.
33 33 2E 32 33 0D 0A 43 61 63 68 65 2D 43 6F 6E  x.23..Cache-Con
74 72 6F 6C 3A 20 6D 61 78 2D 61 67 65 3D 32 35  trol: max-age=25
39 32 30 30 0D 0A 43 6F 6E 6E 65 63 74 69 6F 6E  9200..Connection
3A 20 6B 65 65 70 2D 61 6C 69 76 65 0D 0A 0D 0A  : keep-alive....
```

=+=
+

```
[**] spp_http_decode: IIS Unicode attack detected [**]
04/12-05:44:30.335189 213.121.247.193:61550 -> x.x.x.23:80
TCP TTL:41 TOS:0x0 ID:3033 IpLen:20 DgmLen:296 DF
***AP*** Seq: 0xEFD1578B  Ack: 0x84617566  Win: 0x7D78  TcpLen: 32
TCP Options (3) => NOP NOP TS: 15433377 0
47 45 54 20 2F 69 69 73 61 64 6D 70 77 64 2F 2E  GET /iisadmpwd/.
2E 25 63 30 25 61 66 2E 2E 2F 2E 2E 25 63 30 25  .%c0%af../..%c0%
61 66 2E 2E 2F 2E 2E 25 63 30 25 61 66 2E 2E 2F  af../..%c0%af../
77 69 6E 6E 74 33 35 31 2F 73 79 73 74 65 6D 33  winnt351/system3
32 2F 63 6D 64 2E 65 78 65 3F 2F 63 2B 64 69 72  2/cmd.exe?/c+dir
2B 63 3A 5C 20 48 54 54 50 2F 31 2E 30 0D 0A 56  +c:\ HTTP/1.0..V
69 61 3A 20 31 2E 30 20 50 72 6F 78 79 3A 33 31  ia: 1.0 Proxy:31
32 38 20 28 53 71 75 69 64 2F 32 2E 33 2E 53 54  28 (Squid/2.3.ST
41 42 4C 45 31 29 0D 0A 58 2D 46 6F 72 77 61 72  ABLE1)..X-Forwar
64 65 64 2D 46 6F 72 3A 20 36 32 2E 34 31 2E 33  ded-For: 62.41.3
38 2E 31 30 0D 0A 48 6F 73 74 3A 20 31 34 30 2E  8.10..Host: x.
31 37 38 2E 33 33 2E 32 33 0D 0A 43 61 63 68 65  x.x.23..Cache
2D 43 6F 6E 74 72 6F 6C 3A 20 6D 61 78 2D 61 67  -Control: max-ag
65 3D 32 35 39 32 30 30 0D 0A 43 6F 6E 6E 65 63  e=259200..Connec
74 69 6F 6E 3A 20 6B 65 65 70 2D 61 6C 69 76 65  tion: keep-alive
0D 0A 0D 0A                                      ....
```

=+=
+

5.1. Source of Trace.

This trace was listed on Sans.org page, more specifically the web address: http://www.sans.org/y2k/041901.htm.

5.2. Detect was generated by:

This scan was logged by the Snort IDS. I was not the individual that took it, and am unsure of the setup of the specific machine that captured it, but am able to tell which specific IDS was used by the format.

5.3. Probability the source address was spoofed:

The probability of the source code being spoofed is slim, as this attack is attempting to get root access through the command line interpreter, cmd.exe as you can see in the trace. Since this is the prize the attacker is looking for, the address can't be spoofed or it will never see the results of its' attempt. However, it is noted that it is a hidden address behind at least one squid proxy.

5.4. Description of attack:

This attack looks like CVE number: CVE-2000-0884, where it is an attack using a known vulnerability of placing Unicode characters in the URL giving the attacker the ability to get access to directories out of the web root.

5.5. Attack mechanism:

This attack is attempting to gain root access to the box through the command shell interpreter known as cmd.exe n the Microsoft Windows world. It is using a known vulnerability in IIS 4.0 and 5.0 attempting to get cmd.exe through the default paths for Windows. The vulnerability it is using is mal formed URL's with Unicode encoded characters.

5.6. Correlations:

Other instances of this same attack are referenced below:

http://www.securityfocus.com/cgi-bin/vulns-item.pl?section=info&id=1806
http://support.vigilante.com/support/documents/nx_sans.htm

5.7. Evidence of active targeting:

Since I do not have any traces around this one, it is tough to see any relationship with other machines in this network. However, some things can be determined simply by examining the traces.

First off, the time between each of these attacks is very small, yet the packet ID's and the source ports are jumping up by a few numbers, saying that the source IP is probably very busy. However, looking further shows that this IP address is a squid proxy, so this is not a valid test.

Secondly, since the attacks are going after different versions of IIS servers (as noted by the changes in directories from winnt to winnt351, we can also assume that the machine referenced here was not previously OS fingerprinted, again assuming the attacker is not taking time for this machine alone.

Lastly, there is no other machines noted in this trace for this network, showing a clear sign that the attacker is only looking for this machine on this network.

Putting these things together lead me to believe that this attacker is a attempting this attack on this network only (right now) to it's known port 80 hosts. It is my opinion with this limited information that this is active targeting. However, without more information, this is a tough call.

5.8. Severity:

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

	Description	**Rating**
Criticality	The machine is assumed to be a critical web server, as it is open from the firewall to port 80.	4
Lethality	It's a root access attempt, so very high.	5
System Countermeasures	This machine didn't respond, so we will assume that it is well patched, but after all it is IIS, so the rating can't be over 3.	3
Network Countermeasures	The submitter stated that this attack was going through the firewall, so we assume there is one there. Without knowing how strong the ruleset is, we'll give it an average rating.	3

Severity = 4 (+) 5 – 3 (+) 3
Severity = 3

5.9. Defensive recommendation:

If this is in fact the web server, then port 80 must remain open on the firewall. If it isn't, blocking that port would be a good start. The machine did not respond to this request, so it looks as though it is well patched. Therefore, no other defense recommendations are neccessary.

5.10. Multiple choice test question:

```
[**] spp_http_decode: IIS Unicode attack detected [**]
04/12-05:44:29.537613 213.121.247.193:61522 -> x.x.x.23:80
TCP TTL:41 TOS:0x0 ID:2938 IpLen:20 DgmLen:289 DF
***AP*** Seq: 0xEF818D34  Ack: 0x844F3E92  Win: 0x7D78  TcpLen: 32
TCP Options (3) => NOP NOP TS: 15433327 0
47 45 54 20 2F 6D 73 61 64 63 2F 2E 2E 25 63 30  GET /msadc/..%c0
25 61 66 2E 2E 2F 2E 2E 25 63 30 25 61 66 2E 2E  %af../..%c0%af..
2F 2E 2E 25 63 30 25 61 66 2E 2E 2F 77 69 6E 6E  /..%c0%af../winn
74 2F 73 79 73 74 65 6D 33 32 2F 63 6D 64 2E 65  t/system32/cmd.e
78 65 3F 2F 63 2B 64 69 72 2B 63 3A 5C 20 48 54  xe?/c+dir+c:\ HT
54 50 2F 31 2E 30 0D 0A 56 69 61 3A 20 31 2E 30  TP/1.0..Via: 1.0
20 50 72 6F 78 79 3A 33 31 32 38 20 28 53 71 75   Proxy:3128 (Squ
69 64 2F 32 2E 33 2E 53 54 41 42 4C 45 31 29 0D  id/2.3.STABLE1).
0A 58 2D 46 6F 72 77 61 72 64 65 64 2D 46 6F 72  .X-Forwarded-For
3A 20 36 32 2E 34 31 2E 33 38 2E 31 30 0D 0A 48  : 62.41.38.10..H
6F 73 74 3A 20 31 34 30 2E 31 37 38 2E 33 33 2E  ost: x.x.x.
32 33 0D 0A 43 61 63 68 65 2D 43 6F 6E 74 72 6F  23..Cache-Contro
6C 3A 20 6D 61 78 2D 61 67 65 3D 32 35 39 32 30  l: max-age=25920
30 0D 0A 43 6F 6E 6E 65 63 74 69 6F 6E 3A 20 6B  0..Connection: k
65 65 70 2D 61 6C 69 76 65 0D 0A 0D 0A           eep-alive....

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

What evidence is shown that the source IP address might not be the attacker?

a) The attack is hidden in a TCP Syn packet, so it doesn't require a three way handshake and could therefore be spoofed.
b) The attacker is using a proxy server.
c) The packet ID's are jumping too much not to be crafted packets.
d) The source ports are all the same.
Answer: b

Part III Analyze This Scenario

Executive Summary of Analysis

Most traffic on any network is good intention or innocent traffic. If it wasn't, less companies would have an Internet connection, or allow their employees use of that connection. However, due to the malicious behavior of some individuals, and the vast damage they can cause, all traffic must be analyzed and compared to a list of known "signatures" of packets that are potentially malicious to avoid loss of business due to compromised systems. The university files that make up the analysis of this paper are no exception.

The traffic on the network at this university that is being monitored is quite extensive, and so therefore is the potential for a malicious packet. In order to capture the potentially malicious traffic, three machines were placed on the network to gather data and compare that data to known potential "signatures" as mentioned above. The data analyzed represents 5 days of packets gathered from these source machines. To give an idea of the amount of alerts and scans recorded, the total number of records or packets that have flagged some form of alert to be analyzed is over 800,000.

In order to analyze such a large amount of data, the data must be pared down into manageable chunks. In this process, unfortunately some data might get lost if the paring down is not done correctly. In any forensic study it is important to not only pull out the glaring by sheer amount of packets or occurrences, but to also look closely at the severity no matter how few the packets.

As an example, thousands of port scans don't harm the network directly. They may allow hackers to gain valuable information for the future, but one small well-crafted exploit packet that you are vulnerable to, is far more dangerous in the short term. If not scrutinized properly it's these packets that could slip through your fingers.

In order to gain knowledge about the data in this practical, I went through several steps to pare down the information into manageable chunks as well as weed out the small but important packets. The first step was to look at the total number of occurrences of what the Snort Intrusion Detection System labeled as an alert. From that, I was able to get a good understanding of how often and with what types of attacks the network is being hit.

From a different perspective, I filtered out the top talkers in the network from the stand point of port scans, alerts and out of spec files (definitions of each to follow) to allow me to use that data from which to further pare down the list of alerts gathered.

Once that list was comprised, a careful examination of the source and destination ports of these packets was investigated. From that port list, the number and types of machines the source IP addresses were hitting and the comparison to the other files, a hypothesis could be created about the true intention of the traffic that flagged the alert.

Throughout the rest of this paper, you will see the specifics on how traffic was either dismissed as innocent, or flagged for further investigation. If flagged for further investigation, a decision was made on how to proceed with those specific IP addresses or ports, and it was addressed by the paper's end.

Five particularly suspicious machines were further analyzed and tracked to their source networks in this country and in others. Several machines were flagged as possible compromised hosts, and several machines were listed as machines that could be dismissed with a simple discussion with the end user to answer a question about how that machine is being used.

It was this process that allowed me to come to some conclusions about certain machines listed at the end of this document. Overall the network management seems very loose. It appears that traffic going out of the network is not blocked at all, allowing university students to use Internet bandwidth for Internet radio, heavy Internet gaming, IRC and MSN chat among others. It is also the opinion of this analyst that many students are performing network port scans, either learning hacking tools, or actually utilizing them to do investigative work. A policy about proper Internet usability should be created and put in force.

From the external world, a more restrictive firewall set would also be extremely helpful in blocking access to internal machines over the higher, less documented ports.

Both of these recommendations as well as specific machines to look into for compromise are listed in the last section of this document. The information between this and the list of machines to check into is the specific analysis methods and reasoning. Let's start by identifying the files that were analyzed in this section.

Files Analyzed

The list of files analyzed were comprised of consecutive days from October 5th through October 9th, 2001. They included Scans, Alerts and Out of Spec files and are listed below:

10/5/01: scans_011005_gz.txt
alert_011005_gz.txt
oos_Oct_5_2001_gz.txt

10/6/01: scans_011006_gz.txt
alert_011006_gz.txt
oos_Oct_6_2001_gz.txt

10/7/01: scans_011007_gz.txt
alert_011007_gz.txt
oos_Oct_7_2001_gz.txt

10/8/01: scans_011008_gz.txt
alert_011008_gz.txt
oos_Oct_8_2001_gz.txt

10/9/01: scans_011009_gz.txt
alert_011009_gz.txt
oos_Oct_9_2001_gz.txt

Computers Gathering Logs

There were three distinct machines gathering data on the network. One machine was setup with a standard Snort IDS rule set, one machine was setup to capture port scans and one was setup to simply collect files that contained oddities not able to be accounted for in a standard IP packet, called Out of Spec (OOS). An example of the OOS packets would be a packet that has all the TCP reserved bits turned on. Since the bits are reserved they are not found in a normal IP packet, and therefore generate a need for immediate attention.

Network Detects by Number of Occurrences

From over 320,000 total alerts in the alert file, there were 14,854 different types of alerts recognized in the combined 5 days worth of data. Obviously we can't examine all 14,854 different types, so we must begin by looking at the most common.

Paring out the top 10 types of alerts allows us to see what the most common alert types are. From this we can find out what IP addresses are both the source and destinations of the most common alerts to identify patterns. From there we will take a look at the top 10 talkers without looking at the alerts themselves.

The top 10 Alerts by occurrence are listed below:

Alert Type	Number of Occurrences
WEB-MISC Attempt to execute cmd	59208
MISC Large UDP Packet	37442
spp_http_decode: IIS Unicode attack detected	30855
ICMP Echo Request speedera	7257
INFO MSN IM Chat data	6932
ICMP Echo Request Nmap or HPING2	6228
spp_portscan: portscan status from MY.NET.160.114: 4 connections across 4 hosts: TCP(0), UDP(4)	5720
Spp_portscan: portscan status from MY.NET.160.114: 5 connections across 5 hosts: TCP(0), UDP(5)	5157
WEB-MISC prefix-get //	4873
Spp_portscan: portscan status from MY.NET.160.114: 3 connections	4395

Alert Type	Number of Occurrences
across 3 hosts: TCP(0), UDP(3)	

As noticed above, there are quite a few port scans coming from the MY.Net.160.114 machine. Without any other data, this looks very suspicious. Later in the document we will discuss this in more detail. However, for now, since port scans show up both in the scans tables and the alerts tables, factoring out the port scan alerts into one alert, rather than so many unique alerts gives us a better understanding of the type and frequency of the other of attacks.

After filtering out the port scans there were only 128 different alert types. This is still a few too many to identify in one paper, so again, let's take the top 10 of those.

The top 10 alerts by occurrence filtering out all port scans are listed below:

Alert Type	Number of Occurrences
WEB-MISC Attempt to execute cmd	59208
MISC Large UDP Packet	37442
Spp_http_decode: IIS Unicode attack detected	30855
ICMP Echo Request speedera	7257
INFO MSN IM Chat data	6932
ICMP Echo Request Nmap or HPING2	6228
WEB-MISC prefix-get //	4873
ICMP Destination Unreachable (Communication Administratively Prohibited)	4340
MISC traceroute	4219
MISC source port 53 to <1024	3802

Without more data on who is doing the talking, this information is insufficient to determine whether traffic can be innocently explained or not. Further analysis by pulling out the top talkers and cross-referencing these talkers with the most common alerts will provide us much better data. The most effective way to do this is to find out who the top talkers were.

Top Talkers

Below are three tables showing the top 10 talkers in by source IP address by scans, alerts and out of spec files. Each of the different types of files are shown by source IP addresses are represented. This top talkers list, allows us to further analyze the serious offenders by using those IP addresses which show up more regularly to concentrate our analysis.

Top 10 Scanning IPs

Source IP	Number of Occurrences
MY.NET.160.114	237857
MY.NET.221.250	23868
205.188.244.121	19388
205.188.246.121	18767
MY.NET.233.246	18760
205.188.233.121	17952
205.188.233.153	17005
205.188.233.185	16165
205.188.244.57	16101
MY.NET.223.58	12613

Top 10 Alerts generators

Source IP	Number of Occurrences
212.29.222.114	16747
130.58.144.65	8260
MY.NET.205.30	7263
209.190.237.123	6413
61.153.17.244	6199
61.153.17.246	5189
61.153.17.188	5115
MY.NET.14.1	3523
211.90.120.41	3102
MY.NET.225.6	3097

Top 10 OOS Source IP's

Source IP	Number of Occurrences
206.65.191.129	19
MY.NET.237.182	16
199.183.24.194	9
MY.NET.241.230	7
24.0.154.106	5
130.207.193.70	4
194.82.103.75	3
198.186.202.147	3
63.202.233.119	3
208.63.188.106	2

Scanning IP's investigated a little further:

MY.NET.160.114 only shows up with minimal Alert's but with the sheer number of scans to odd port number and thousands of hosts, make this a machine to look into.

MY.NET.221.250 only shows up with 1 traceroute alert, so chances are this machine is not running any exploits, but the sheer number of scans with odd port numbers and thousands of hosts, in very short time bursts throughout the five days make this a machine to look into.

All of the IP address 205.188.244.121, 205.188.246.121, 205.188.233.121, 205.188.233.153, 205.188.233.185 and 205.188.244.57 had no alerts in the alert table. After looking at the associated machines and the fact that the scans are all "from" port 6970, it appears as though these machines are not scanning the network, but instead are Real Audio servers and many machines internally are visiting them. Most probably this is innocent traffic. However, it wouldn't hurt to double check the numbers. Below in another section, I will do just that.

MY.NET.233.246 also appears to be innocent traffic as it had only 1 ICMP packet in the alerts table and all of the "scans" seemed to be the same port 28800, a well-known gaming port. Chances are as the machines above, this is innocent traffic.

MY.NET.223.58 had only one event in the alert log. All of the ports were either 6699 and 6257. A little investigation revealed that this is probably a Microsoft Proxy server sitting behind the firewall as noted in the web site listed below:

http://www.linxent.com/articles/Proxy%20Server%20Setup%20for%20WinMX.htm

Alert generating IP's investigated further:

Utilizing the top 10 talkers' IP addresses displayed in the above alerts table, allows us to then determine what types of attacks those machines are using. It also gives us the ability to find out what types of machines they are attacking.

Source IP	Alert Type	Number of Occurrences	Number of different Hosts
212.29.222.114	WEB-MISC Attempt to execute cmd	11111	8389
MY.NET.205.30	ICMP Echo Request speedera	7257	2
209.190.237.123	MISC Large UDP Packet	6389	2
61.153.17.244	MISC Large UDP Packet	6193	3
212.29.222.114	spp_http_decode: IIS Unicode attack detected	5634	3221
130.58.144.65	WEB-MISC Attempt to execute cmd	5435	3753
61.153.17.246	MISC Large UDP Packet	5189	1
61.153.17.188	MISC Large UDP Packet	5115	6
MY.NET.14.1	ICMP Destination Unreachable (Communication Administratively Prohibited)	3522	217
130.58.144.65	spp_http_decode: IIS Unicode attack detected	2825	1701
211.90.120.41	WEB-MISC Attempt to execute cmd	2001	1306
211.90.120.41	spp_http_decode: IIS Unicode attack detected	1101	638
209.190.237.123	ICMP Fragment Reassembly Time Exceeded	20	1

Source IP	Alert Type	Number of Occurrences	Number of different Hosts
MY.NET.205.30	INFO Possible IRC Access	6	2
61.153.17.244	High port 65535 udp - possible Red Worm – traffic	5	1
209.190.237.123	High port 65535 udp - possible Red Worm – traffic	4	1
61.153.17.244	Port 55850 udp - Possible myserver activity - ref. 010313-1	1	1
212.29.222.114	beetle.ucs	1	1
MY.NET.14.1	ICMP Destination Unreachable (Host Unreachable)	1	1
212.29.222.114	SCAN Synscan Portscan ID 19104	1	1

Now let's begin to analyze each of these to find out the severity.

The **"WEB-MISC Attempt to execute cmd"** is an attack that seeks to gain command access on an unprotected web server. Once the cmd.exe (command interpreter) is launched, many exploits can be taken advantage of.

http://www.incidents.org/react/dosstormworm.php had this to say about this alert:

"DoS.Storm.Worm is a worm that seeks out Microsoft Internet Information Services (IIS) systems that have not applied the proper security patches. Any such systems that it finds are then infected with the worm. The payload of this worm performs a denial of service attack on http:/ /www.microsoft.com 1 (http://www.symantec.com/avcenter/venc/data/dos.storm.worm.html)"

To summarize, one type of this attack is after a TCP session is established with an MS IIS 4.0 or 5.0 Server. The packet that triggers the alert is sent with a specific payload that executes a worm. This worm emails the address of this newly compromised box to the original attacker and begins a JavaScript that executes and starts a denial of service on Microsoft.

There are several other vulnerabilities that can be exploited by this cmd access. This same vulnerability is what was used by both codered and nimda worms.

This alert was logged 59208 times. Looking a little further, that alert came specifically from over 9,000 different hosts over this 5 day period to over 19,000 internal hosts.

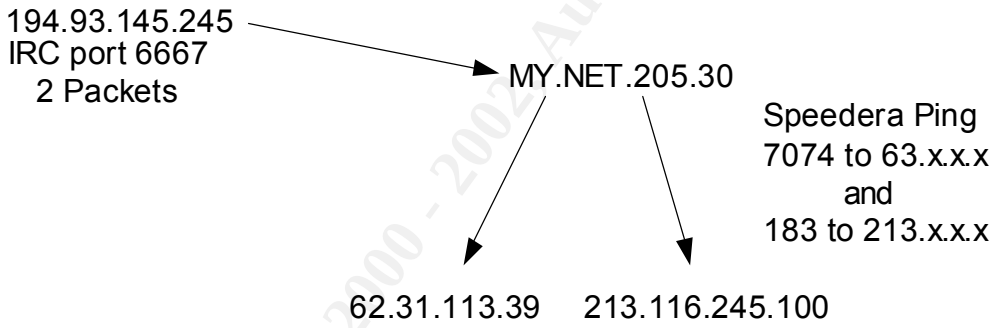
Three IP addresses stick out from the rest as there contribution to this particular attack was almost 30% of the total. They are 212.29.222.114, 130.58.144.65 and 211.90.120.41. Incidentally, those same three IP addresses also ran the "spp_http_decode: IIS Unicode attack detected" attack as listed below, but yet didn't appear anywhere else in the alert logs.

Considering that these are files from October of 2001 with both nimda and codered still running amuck in the wild, it would be a good guess that these machines are in fact infected with one of these two viruses.

Furthermore, looking at the times these attacks came in from all three of these hosts, this couldn't be a specific attacker using any intelligence at the console; this has to be a script as it runs 24 hours per day once it begins. This is very consistent with both of these viruses, and is therefore the conclusion of this analyst.

The next entry in our table above is **"ICMP Echo Request speedera"** alert. This alert is generated by a specific type of ping. What is odd is, unlike the "WEB-MISC Attempt to execute cmd" alert, this alert is only perpetrated by one host: MY.NET.205.30. In addition, this host only sends this attack to 2 other external hosts. What makes this even more suspect is that there are 2 short IRC packets that come in just before the speedera pings are released. A very strong case for a compromised host.

A link graph of that data would look like this:



The next entry in our table above is **"MISC Large UDP Packet"**. This alert came from 74 different hosts and went to 94 different hosts. They also ran across 3100 different ports. Had this been to a grouping of ports, like to 12345 or 27374 like sub seven or any other known ports, or if the attacking sites or the attacked sites been in some mathematical order, we could have a more concrete sign of a problem.

Based on the results, without the packets themselves it is difficult to tell what this traffic is. This could simply be a whole lot of false positives. However, given the number of ports, it appears more like something is in fact going on.

The biggest offenders of this are 209.190.237.123 and 61.153.17.244 and to only 5 hosts total. This is really the only credible alert, as the Red Worm alerts all seem to just be coincidental, as all the ports in the traffic are very high. Without more data, I would simply have to mark these machines as suspect. I have flagged these to be traced in another section of the paper.

The next entry in our table above is **"spp_http_decode: IIS Unicode attack detected"** attack. This attack is a common attack where by the attacker uses a mal formed URL

with Unicode text to a web machine forcing its way to get access to directories out of the web root. This attack is more clearly defined above in section 2 as it is one of the attacks analyzed in this practical.

Similar to the "WEB-MISC Attempt to execute cmd" attack, this attack came from over 4,800 different source IP addresses and went to almost 11,000. However, the same three IP addresses stick out from the rest as there contribution to this attack also was almost 30% of the total. They are 212.29.222.114, 130.58.144.65 and 211.90.120.41. This only shows more clearly that these machines are compromised. Either that, or we are dealing with a very unsophisticated script-kiddie that is not aware of how to hide his IP address. These addresses will be researched below to find out their origin.

The next entry in our table above is **"ICMP Destination Unreachable (Communication Administratively Prohibited)"**. The reason this alert is triggered is simple. A ping command from a source address is aimed at a machine that is blocked by an access list on a router. The router responds with this message. This particular ICMP response is usually in great numbers seen when a compromised host is attempting to gather reconnaissance for later attacks.

This alert was logged by 37 different source addresses to 301 destination addresses. 81% of these requests came from the internal address of My.Net.14.1. Not having a network map this analyst can't be sure, but this sounds like this host might be an internal router. Since this IP address doesn't appear in the port scan log, nor have any other alerts associated with it, it seems even more likely. A review of the network map would solve this problem, but for now, I will chalk this up a network router.

The next entry in our table above is **"ICMP Fragment Reassembly Time Exceeded"** This alert is generated when a packet comes through with fragment pieces but never gets all the pieces to assemble. Eventually the buffer that the packets are sitting in time runs out. The packets are dumped. While this can happen with congestion on the Internet, it can also be the cause of a Denial of Service by filling up those buffers preventing other packets from being seen by the operating system. Further research on this is necessary.

Neither source nor destination IP addresses in this alert table appear enough times to warrant a Denial of Service attack. There are no other known vulnerabilities with this time exceeded accept as a recon tool. Since neither the source nor destination appeared many times in the table, it is my opinion that this traffic can be dismissed as congested innocent traffic forcing retransmission. This is further demonstrated when looking at the packet times. They are very spread out across all 5 days with no groups suggesting a DOS.

The next entry in our table above is **"INFO Possible IRC Access"** This alert is generated by port only. It is listed as port 6667. It is a possible false positive as this is a valid port number, but requires some research in relation to this network as referenced below.

It was logged 3638 times across 99 different machines. My.Net.217.30 and My.Net.217.242 were the only machines that logged more than one or two instances. The next step for these other internal 97 hosts would be to research each of these machines to determine their intention. Since the source port is not 80 and the dest port is 6667, we can conclude that this is not web innocent web traffic. However, in the interest of time, we will focus on the top couple of machines mentioned above.

My.Net.217.30 and My.Net.217.242 should probably be looked into to see if they are being used as IRC bots, or innocently used as IRC communication channels for simple chatting online. Since both machines only communicates to 4 different external hosts, it was easy to track down those host. One was a legitamate IRC server, of the other three, one was a nameserver, one a United Kingdom host and one didn't respond to a ping. From this we can conclude that these are most probably IRC Bots taking commands through IRC channels doing dirty work for the true attackers.

The only other times that My.Net.217.30 was seen was in a few "ICMP Echo Request CyberKit 2.2 Windows" alerts. A closer look at CyberKit revealed that it could be false positives at the site:

http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids154&view=event

My.Net.217.242 is only seen as IRC alerts. A quick talk with the end user to double check their habits might be a good idea. But if (as I suspect would be the case) the user doesn't communicate through IRC, then we definitely have 2 compromised hosts.

The next entry in our table above is **"High port 65535 udp - possible Red Worm – traffic"** This alert is generated by the use of the port 65535. While an unlikely port, it is a valid port. Since the traffic from this machine that comes before this alert is generated from very high ports, it is likely that this is simply a false positive.

The next entry in our table above is **"Port 55850 udp - Possible myserver activity - ref. 010313-1"** This alert is generated by the use of the port 55850. While an unlikely port, it is a valid port. Since the traffic from this machine that comes before this alert is generated from very high ports, it is likely that this is simply a false positive.

The next entry in our table above is **"beetle.ucs"** This attack only showed up a few times across only a few machines. There was one internal machine My.Net.70.69 that showed up in all communications marking the "beetle.ucs" alert. With no other signs of compromise in this machine, this attack is probably a false positive of web traffic between two hosts. My default rule set from Snort did not have this alert, and a web search on Whitehats.com and Google.com produced nothing. I am unfamiliar with this attack and can only guess that it is a rule added into this university's rule set.

The next entry in our table above is **"ICMP Destination Unreachable (Host Unreachable)"** This alert is generated as a warning on a possible recon mission. This ICMP message is a valid message for any mistyped host, or disconnected web site, but is sometimes used as a collection tool to help gain valuable information to help future

attacks. It only appears in this table 1 time, and does not make the top 10 alerts generated so is easily dismissed as innocent.

The next entry in our table above is **"SCAN Synscan Portscan ID 19104"** This alert is generated when the packet ID is 19104. While an unlikely port, the port number is randomly generated and could easily be 19104 at some time. Since this alert is only seen this one time from one machine, and all of our logs, it is very likely that this is a false positive.

For more information about this alert, look at the following web site:
http://www.whitehats.com/IDS/521

Looking back at our top 10 Alerts table from the top, it looks as though the top talkers of alerts also happened to answer the questions of where the top 10 alerts came from and why, except for a couple of attacks listed below:

The "INFO MSN IM Chat data" alert, was rampant throughout the organization. A look at the source and destination IP addresses, looks as though all of the internal addresses are connecting to 64.4.12.x machines which are all registered to Hotmail. This can easily be dismissed as legitimate MSN Chat traffic, as these Hotmail servers are run by MSN. Considering this is a university, this is a very likely occurrence.

The "ICMP Echo Request Nmap or HPING2" is a special alert that refers to a specific type of ICMP message that is indicative of the Nmap or HPING2 port scanning tools. Although it can produce a false positive, the MY.NET.225.6 and MY.NET.219.38 machines have over 4,600 alerts associated with them to the same 3 hosts on the external network, making a good case for these machines running nmap scans. Maybe a talk with these two individuals would be in order.

The "WEB-MISC prefix-get //" alert was seen many times but only to two differet hosts. MY.NET.253.114 is the large majority of those packets, MY.NET.253.115 is the other only real looks as though it is a possible issue.

Further investigation of the MY.NET.253.114 and MY.NET.253.115 machines, reveals several other alerts, such as "WEB-CGI rsh access", "Watchlist 000222 NET-NCFC" and "Possible trojan server activity". This leads me to believe that these machines are compromised and require investigation. Especially since the source IP addresses that are utilizing these machines are not part of one single net block.

The "MISC traceroute" alert shows up enough times to be questionable, especially since it's destination is almost entirely MY.NET.140.9. More importantly, upon further research of this machine, it is obvious that it has been the destination of many different types of alerts including "WEB-MISC Attempt to execute cmd", "Port 55850 udp - Possible myserver activity - ref. 010313-1", and "ICMP Destination Unreachable (Communication Administratively Prohibited)". This machine also requires further research.

Lastly, the "MISC source port 53 to <1024" alert appeared several thousand times, but mostly to the Destinations of MY.NET.1.2, .3, .4 and .5, and without knowing the network, I am assuming that with these low numbers, that they are in fact DNS servers. If that is the case we are probably looking at innocent traffic considering all of the traffic is to destination port 53 or DNS.

OOS Files investigated further:

A further investigation of the OOS files has a direct correlation with the alerts source IP addresses. Overall there were 8,143 different source IP addresses in the alert files, and 67 source IP addresses in the OOS files. Together, 24 addresses were common between the two. There were 561 different Source IP's in the Scans table, and 44 addresses were common between the scan files and alert files, of which 19 were the same as the ones in the alerts files.

Of the 67 common source IP addresses in the alerts files and the OOS files, the top 10 were:

Source IP
206.65.191.129
199.183.24.194
MY.NET.241.230
24.0.154.106
130.207.193.70
198.186.202.147
MY.NET.235.186
MY.NET.206.82
194.82.103.75
198.82.75.200

Of the 24 common source IP addresses in the alert files and the OOS files, the top 10 were:

Source IP
206.65.191.129
199.183.24.194
MY.NET.241.230
24.0.154.106
130.207.193.70
198.186.202.147
MY.NET.235.186
MY.NET.206.82
194.82.103.75
198.82.75.200

Of the 44 common source IP addresses in the scans files and the OOS files, the top 10 were:

Source IP
206.65.191.129
MY.NET.237.182
MY.NET.241.230
199.183.24.194
24.0.154.106
130.207.193.70
194.82.103.75
24.112.130.237
MY.NET.235.186
65.15.76.184

There were 19 common source IP addresses between all three tables. Not surprisingly some of the same IP addresses that come to the top of the other lists appeared in the list below are the top 10 of those common to all three.

Source IP
206.65.191.129
199.183.24.194
MY.NET.241.230
24.0.154.106
130.207.193.70
MY.NET.235.186
194.82.103.75
24.112.130.237
198.82.75.200
213.237.116.91

Before, we focus our time on these IP addresses, a quick look at the OOS files reveals the reasons for the placement in these files. The reserved bits on a majority of the packets in these files are turned on. This obviously presents a problem, as these bits are not supposed to be on as they are in fact reserved. However, this fact alone is not enough to quantify all these addresses as suspect.

As mentioned in the practical of Chris Baker dated May 29th, 2001, "It is worthwhile to note that the Out of Spec logs have captured demon.net malformed IP packets. Packets that come through the demon.net network often show up malformed. This problem has been acknowledged by Demon Internet as a software issue on their routers, yet the problem has not been resolved."

Utilizing that knowledge, I believe that it is a reasonable assumption that the remainder of the study of these OOS files be concentrated the same as the previous analysis, by source and destination IP address and port.

206.65.191.129 had all of its' entries in the Alerts table for the Queso Figerprint. And all for MY.Net.212.190. Queso is a tool similar to Nmap, a port scanning tool. I would be concerned if there was any other traffic between these two hosts, or if MY.NET.212.190 appeared anywhere else, but the only other traffic noted in any of the files are three entries in the alerts table for IRC access.

199.183.24.194 communicated to port 25 on three internal servers. From their numbers being so close together, I will assume these are the three mail servers in this environment. There is also however some traffic that generates the Queso Fingerprint alert, which is basically a port mapper like Nmap. While this is odd, there isn't enough traffic to make my ears perk up. It also looks as though a false positive to me as there are only 3 machines listed as the recipients, and they are the same three machines as noted above, the mail servers.

MY.NET.241.230. That address was in the OOS files several times, and in each instance either the source port or destination port was 6346 – Gnutella peer to peer sharing utility. As for the scans files and alert files, oddly enough all contained port 6346 either in the source or destination ports. Considering that no other alert was set for this IP address, and no other scan, it is this analysts opinion that this is a pure case of this end user sharing files with a couple of buddies through Gnutella, or a false positive based on port number alone. A discussion, still isn't a bad idea to have with the user of this machine.

24.0.154.106 appeared in all three tables with all different port numbers, but all to the same address: MY.NET.218.254. The odd thing about this is simply that this address had no report in the scans table and only three entries in the alerts table for IRC access. Don't really see an issue with this machine because of the limited packets and flags caught.

130.207.193.70 the next one in our list, also generated some Queso Fingerprints and those of course appeared in both the scans and alerts table as basically the same thing. 5 alerts to two different IP addresses, one at port 25 one at port 113. Neither a cause for alarm. Most likely we are looking at real communication to a mail server and another mail server running ident.

194.82.103.75 appered in all three tables going to a couple of internal web servers. While this is a guess, there were no other alert generated, nor scans, nor OOS info, so we will make the assumption that it was in fact web traffic.

24.112.130.237 is the first machine in this list that showed some possible signs of compromise. The alerts that were generated were all in relation to FIN Scans from port 3130 – a known Squid Proxy port to port 6346, a known Gnutella peer to peer sharing port. Fin scans are a little more difficult to be found in normal traffic, so that by itself sparks some concern. But the source and destination ports are what tickled my ears here.

Those packets were all going to the machine with IP address MY.NET.238.90. While it had no other alerts, it was loaded with Gnutella alerts. Looks like this machine may be a peer to peer sharer, but more likely a compromised peer to peer sharer.

198.82.75.200 is reference a few times in the alerts table and the scans table all for the same Gnutella alerts. This one however, is all to the same host. My guess is a false alarm as this alert is generated by port only.

213.237.116.91 lastly has the same issues as above. A couple of entries, no other alerts. My guess, this is a false positive. Either that or some hacker is trying to get valuable pieces of data off your network one packet at a time using several dozen hosts to avoid being caught. This I think is unklikely, but then again all of these files were flagged because of the TCP flags and reserved bits. A close watch of these IP addresses might not be a bad idea.

Specific Machines worth investigating:

1. 205.188.244.121 and 205.188.246.121 had several hundred machines noted as being "scanned" by this address for port 6970. Since this port is known to be Real Audio, I though it wise to investigate these.

A ping –a command returned the following DNS registrations:

205.188.244.121 is registered as G2LB2.spinner.com
205.188.246.121 is registered as G2LB3.spinner.com

A www.networksolutions.com whois lookup revealed this about Spinner.com:

Domain Name: SPINNER.COM

Registrant:

Spinner Networks, Inc.

1209 Howard Ave Suite 200

Burlingame, CA 90410

US

Created on.....: Dec 23, 1999

Expires on.....: Dec 23, 2001

Record Last Updated on..: Jan 05, 2000

Registrar.....: America Online, Inc.

http://whois.registrar.aol.com/whois/

Administrative Contact:

Domain Administration, Spinner

Spinner Networks, Inc.
1209 Howard Ave Suite 200
Burlingame, CA 90410
US
Email. hostmaster@SPINNER.COM
Tel. 415 934 2700
Fax. 415 934 2756
Technical Contact:
Domain Administration, Spinner
Spinner Networks, Inc.
1209 Howard Ave Suite 200
Burlingame, CA 90410
US
Email. hostmaster@SPINNER.COM
Tel. 415 934 2700
Fax. 415 934 2756
Domain servers:
dns-01.spinner.net
152.163.159.239
dns-02.spinner.net
205.188.157.239

www.spinner.com is an Internet radio site. These machines were picked up by the scan but are most probably innocent Internet radio traffic.

2. 209.190.237.123 appeared to be a host that sent many Misc UDP packets, a possible Red Worm packet and a few time exceed packets. I thought maybe this one was worth a look.

Turns out this is registered to an ISP called At-Lan-Tec corp, a DSL provider. This particular IP address has a DNS registration of 7b.edbed1.client.atlantech.net, which did respond to ping but didn't respond to FTP or HTTP.

A www.networksolutions.com whois lookup revealed this about the ISP:

Registrant:
At-Lan-Tec, Incorporated (ATLANTEC-DOM)
16021 industrial dr. suite
Gaithersburg, MD 20877

US

Domain Name: ATLANTEC.NET

Administrative Contact, Technical Contact:

Cornell, Dennis (DC1088) DCORNELL@ATLANTEC.NET

At-Lan-Tec, Incorporated

16021 industrial dr.

suite 12

Gaithersburg, MD 20877

(301) 590-9090

Billing Contact:

Thomas, Douglas W (DWT3) dthomas@ATLANTEC.NET

At-Lan-Tec, Incorporated

16021 industrial dr.

suite 12

Gaithersburg, MD 20877

(301) 948-7070

Record last updated on 29-Jan-2001.

Record expires on 21-Dec-2001.

Record created on 21-Dec-1995.

Database last updated on 10-Nov-2001 09:40:00 EST.

Domain servers in listed order:

NS1.ABAC.COM	216.55.128.4
NS2.ABAC.COM	216.55.144.4

Considering the commonality of compromised machines on home connections like DSL and cable modems, this is probably a compromised host. Contacting this ISP would be a good idea to let them know what is being sent across their network.

3. 212.29.222.114 It had many of the "WEB-MISC Attempt to execute cmd" and "spp_http_decode: IIS Unicode attack detected" across many hosts. As mentioned above, it is probably an infected host, and the ISP should be contacted.

It did not respond to ping, nor have a web site or ftp site at that IP address, nor have a registered DNS name.

A traceroute revealed that it is a .il host or associated with Isreal as determined by http://www.domainit.com/country-domains.htm web site.

A Whois lookup on www.ARIN.net revealed the following ISP that owns the Net block:

European Regional Internet Registry/RIPE NCC (NET-RIPE-NCC-)
These addresses have been further assigned to European users.
Contact info can be found in the RIPE database, via the
WHOIS and TELNET servers at whois.ripe.net, and at
http://www.ripe.net/db/whois.html
NL
Netname: RIPE-NCC-212
Netblock: 212.0.0.0 - 212.255.255.255
Maintainer: RIPE
Coordinator:
Reseaux IP European Network Co-ordination Centre Singel 258 (RIPE-NCC-ARIN)
nicdb@RIPE.NET
+31 20 535 4444
Domain System inverse mapping provided by:

NS.RIPE.NET	193.0.0.193
NS.EU.NET	192.16.202.11
AUTH03.NS.UU.NET	198.6.1.83
NS2.NIC.FR	192.93.0.4
SUNIC.SUNET.SE	192.36.125.2
MUNNARI.OZ.AU	128.250.1.21
NS.APNIC.NET	203.37.255.97

To search on arbitrary strings, see the Database page on
the RIPE NCC web-site at http://www.ripe.net/db/
Record last updated on 16-Oct-1998.
Database last updated on 10-Nov-2001 19:54:27 EDT.

Using this information and going to that ISP's Whois database at www.ripe.net/db revealed this:

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:        212.29.222.96 - 212.29.222.127
netname:        EFRAT-1
descr:          Efrat
```

```
country:        IL
admin-c:        OH624-RIPE
tech-c:         DB1523-RIPE
status:         ASSIGNED PA
mnt-by:         RIPE-NCC-NONE-MNT
changed:        dbenjamin@barakitc.co.il 19981018
source:         RIPE

person:         Oleg Hanokov
address:        Efrat
address:        Israel
phone:          + 972 3 6452222
fax-no:         + 972 3 6452333
nic-hdl:        OH624-RIPE
notify:         dbenjamin@barak.net.il
changed:        dbenjamin@barak.net.il 19981119
source:         RIPE

person:         Dana Benjamin
address:        Barak I.T.C
address:        15 Hmelacha St Rosh Ha'ayin
address:        Israel 48091
phone:          + 972 3 9001102
fax-no:         + 972 3 9001515
e-mail:         dbenjamin@barakitc.co.il
nic-hdl:        DB1523-RIPE
changed:        dbenjamin@barakitc.co.il 19981126
source:         RIPE
```

Considering the source looks to be a home user and the attacks looking like a codered or nimda type worm, a call to this ISP is probably in order.

4. 130.58.144.65 also has the same 2 specific attacks that have been launched at many different sites. As mentioned above, it is probably an infected host, and the ISP should be contacted.

Further research shows that it has a default web page from a Windows 2000 IIS server or personal web server, although it didn't respond to a ping. It has a DNS registered name as Alumni-web.swarthmore.edu it is registered as a true college in Pennsylvania. Seeing the default page, makes me believe that this is an unmanaged machine sitting on the Internet, probably compromised and being used as a drone to launch attacks.

www.networksolutions.com revealed this information about the domain name:

Registrant:

Swarthmore College (SWARTHMORE-DOM)

500 College Avenue

Swarthmore, PA 19081

US

Domain Name: SWARTHMORE.EDU

Administrative Contact, Billing Contact:

Dumic, Mark J. (MD83) dumic@SWARTHMORE.EDU

Swarthmore College

500 College Avenue

Swarthmore, PA 19081

(610) 328-8511

Technical Contact:

Preset, Adam (AP13139) apreset1@SWARTHMORE.EDU

Swarthmore College

500 College Avenue

Swarthmore, PA 19081-1397

610-328-8508 (FAX) 610-328-7793

Record last updated on 28-Jun-2001.

Record created on 22-Jun-1987.

Database last updated on 10-Nov-2001 21:08:00 EST.

Domain servers in listed order:

OAK.SWARTHMORE.EDU	130.58.64.20
CS.SWARTHMORE.EDU	130.58.68.10
NS1.YIPES.COM	209.213.223.126
NS2.YIPES.COM	209.50.39.102
NS3.YIPES.COM	209.50.40.102
KALVIN.PITZER.EDU	134.173.112.15

A call to this ISP is probably in order as well for the reasons mentioned above.

5. 211.90.120.41 also has the same 2 specific attacks that have been launched at many different sites. As mentioned above, it is probably an infected host, and the ISP should be contacted. It did not respond to ping, nor have a web site or ftp site at that IP address, nor have a registered DNS name.

The registered names in a trace route ended after a host called chinaunicom-gw.customer.alter.net. There are several hosts with different class c address spaces routing the traffic from that point to the host that doesn't respond.

So a quick look at www.ARIN.net for that IP block revealed the following:

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.

A further look for that specific IP out of the 210.x.x.x block using www.APNIC.net whois database revealed this:

inetnum	211.90.0.0 - 211.91.255.255
netname	UNICOM
descr	China United Telecommunications Corporation
country	CN
admin-c	XL31-AP, inverse
tech-c	XL31-AP, inverse
mnt-by	MAINT-CNNIC-AP, inverse
changed	xiaqing@cnnic.net.cn 20000414
source	APNIC

person	XiaoMing Li, inverse
address	6F Office Tower 3, Henderson Centre, Beijing China
country	CN
phone	+86-10-65181800-291
fax-no	+86-10-65181800-777
e-mail	lxmlxm@public3.bta.net.cn, inverse
nic-hdl	XL31-AP, inverse
mnt-by	MAINT-CNNIC-AP, inverse
changed	wangch@cnnic.net.cn 20000331

source APNIC

This makes this host very suspect as China is a well-known source of malicious attacks as well as a well-known source of un-patched machines, as the knowledge of security is not as well disseminated there as it is in this country.

Correlations with other student's Practicals:

Overall, an observation made about all the practicals examined, is that the sheer number of alerts has gone up dramatically over the course of time. In each of the previous practicals there were more days being examined, and yet much few attacks. This leads me to believe one of three scenarios - the Snort IDS system is getting better at picking up attacks or that it is getting worse at generating false positives or finally that the attacks themselves are actually exponentially climbing. None of these are good outcomes, so I will leave the conclusions to another paper.

To site a specific example of what I mean by this, in Bruno Marien's Practical dated March 2001, he stated this about the total alerts:

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

In the 5 days worth of files that I analyzed in this practical there were over 320,000 total, or approximately 64,000 per day or 1 every 1.35 seconds. This seems to be quite a substantial change in patterns over a short time frame, considering that Mr. Marien's was completed and graded in July.

In *Lone Star SANS 2001 GCIA Practical Version 2.8 by Donald Pitts from May 29, 2001*port 28800 (a gaming port) was referenced as one of the most common source and destination ports. His conclusion was the same as mine; that since this is a university's network, the chances that this is truly gaming traffic is quite likely. None of the machine's IP addresses are the same, however, this can easily be attributed to the fact that these are across two different college semesters.

Tools used in Analysis

There were five different days worth of data from three different sources to analyze. Each data set of course had some similar information in that specific machines' IP addresses would be appearing in all three captures across all five days. In order to do a complete analysis, the data had to be combined and cross-referenced. To pull all the information from the different source files into a comprehensive set of data to be analyzed, I figured the best solution would be some form of relational database.

Since the files were all ASCII text files, a simple import into a known database program appeared to be the most logical way to get all the information together. Once all files were integrated into this database, queries could be written to cross-reference IP addresses or specific port numbers and mathematical functions could be performed. For ease of use I chose Microsoft (MS) Access. However, each file type had it's own delimiters making a quick and dirty import impossible. After several round about sessions working with MS Access import capabilities, I decided a different route was necessary.

I began importing each file into MS Excel. Unfortunately again, the files were too big for MS Excel to handle. To bring them into MS Excel, I began using Notepad to open them, split them into two or three pieces, save them, and import them into MS Excel. Once there, I used the "text to columns" feature to split the text up by different delimiters into a standard column separated file. I recorded the process into a macro and ran it for each of the files to create a standard format to import. From there they were ready for import to MS Access.

Once inside MS Access, I began writing queries to pull out the top talkers by both source IP address and destination IP address answering a specific criteria question in the practical, but also providing some critical information for further queries. With that information, I was able to then query top talkers to determine what port numbers both to and from the scans and other alerts were referencing. For the most part MS Access was the tool for all the queries to pull out relevant data from the text files.

When a port seemed suspicious, or unrecognized, a quick search of www.google.com lead me to web sites that offered their interpretation of the ports. These web sites, often times gave me network traces or other interpretations of the uses of that port. This gave me the necessary information to decide whether traffic looked suspicious and required further investigation, or innocent.

Once a strong understanding of the relationship between IP addresses ports and alerts were gained it was time to move on to searching for the source IP addresses and what networks they belonged to. This was done with a variety of different tools including the ping utility with the –a switch, tracert utility and web sites such as www.networksolutions.com and www.arin.net among others, and attempting web and ftp sessions with the names and IP addresses.

Correlations were done searching on www.google.com, www.securityfocus.com and www.whitehats.com as well as of course www.incidents.org.

Once that process was completed, a reference to previous practical papers was required. From the www.incidents.org web site, other practical papers were downloaded and read for correlations. I have reference all of those above in the documentation.

Defense Recommendations:

Generally speaking it looks as though there needs to be a much stronger policy in place as to what can and can not run on the internal network. Specifically a decision has to be made about whether Microsoft Proxy should be allowed on the network, or if any form of port scanning should be allowed. The policy then has to be distributed to the faculty and students and adhered to.

On a less serious note, there are other such protocols and channels that should be looked at like IRC or Real Audio or Gaming and determined if they are permitted on the network. These do have strong bandwidth requirements and can be misused. However, this is a site to site recommendation, not a hard and fast rule.

Regardless of policy, there are a few machines that should be looked at:

External Machines to watch out for:

212.29.222.114,130.58.144.65 and 211.90.120.41 I'd look into further as mentioned above, they have a large number of different types of alerts to different machines and different ports. There is definitely something suspicious about all these machines and the domains they are associated with. Further research might be necessary, but blocking access from them might be a good preventative measure.

I would talk to the owner of MY.NET.70.134 to find out if they have any connection with 209.190.237.123, or they might know them as 7b.edbed1.client.atlantech.net, which might give insight into the Misc Large UDP packets coming from this external host.

I would also talk with the owner of MY.NET.111.22 to find out if they have any connection to 61.153.17.244, which might give insight into the Misc Large UDP packets from that external host.

Internal machines to look into for further action:

MY.NET.253.114 & MY.NET.253.115 as there is good reason to believe that these are web servers that have been compromised and are being used by hackers to do some dirty work on the Internet.

MY.NET.140.9 is probably a compromised box, as the number of alerts it was the destination for and the different blocks of source IP address make this look like a machine that should definitely be investigated.

MY.NET.160.114 - for coming up so many times as some sort of port scanning tool. It only logged alerts in relation to fragment time reassembly. My guess is that this machine is a rooted box being used as a port scanner, or one of the college kids learning some port scanning tools. A talk with this youngster might be a good start.

MY.NET.221.250 for the tremendous amount of scans to odd ports across many machines. This is probably a rooted box, being used as a drone for port scanning or as above one of the college kid learning port scanning. A talk with this person might also be a good start.

MY.NET.217.30 & My.Net.217.242 - a quick talk with the end users about their IRC habits might do some good. If they are not communicating to friends and family using IRC which his the likely conclusion then these machines are IRC bots.

MY.NET.233.246 I'd ask them if they do a lot of gaming to dismiss this one. Although, truthfully, the others listed above are a higher priority. This one I believe can be dismissed easily, but it never hurts to be thorough.

MY.NET.223.58 looks to be a Microsoft Proxy server, but I would take a closer look at that machine to make sure that it is. As mentioned above, a policy must be put in place to determine if this is allowed in the future.

MY.NET.225.6 and MY.NET.219.38 I'd ask about their use of Nmap or HPING2 tools, and ask them to stop using them.

MY.NET.238.90 for being in so much Gnutella traffic with so many different hosts over such a short period of time.

References:

http://www.cknow.com/vtutor/vthistory.htm
http://inventors.about.com/ library/weekly/aa111598.htm
http://www.cert.org/advisories/CA-2001-26.html
http://www.incidents.org/react/nimda.pdf
http://vil.nai.com/vil/virusSummary.asp?virus_k=99209
http://www.newsfactor.com/perl/story/13607.html
http://www.whitehats.com/IDS/177
http://www.whitehats.com/info/IDS457
http://www.whitehats.com/info/IDS456
http://www.cert.org/advisories/CA-2000-22.html
http://www.securityfocus.com/bid/3252
http://www.securityfocus.com/bid/2894
http://archives.neohapsis.com/archives/incidents/2000-03/0140.html
http://www.securityfocus.com/infocus/1508
http://razor.bindview.com/publish/advisories/adv_ssh1crc.html
http://www.securityfocus.com/archive/1/11018
http://www.securityfocus.com/archive/1/7718
http://www.securityfocus.com/archive/1/8398
http://www.securityfocus.com/cgi-bin/vulns-item.pl?section=info&id=1806
http://support.vigilante.com/support/documents/nx_sans.htm
http://www.linxent.com/articles/Proxy%20Server%20Setup%20for%20WinMX.htm
http://www.symantec.com/avcenter/venc/data/dos.storm.worm.html
http://www.incidents.org/react/dosstormworm.php
http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids154&view=event
http://www.whitehats.com/IDS/521
Chris Baker, Intrusion Detection In Depth GCIA Practical Assignment, v2.8 Lone Star SANS, 29 May 2001
http://www.domainit.com/country-domains.htm
www.ripe.net/db
www.APNIC.net
Bruno Marien, GCIA Certification – Practical Assignment, Version 2.8b, Dallas Lone Star – March 2001
Lone Star SANS 2001 GCIA Practical Version 2.8 by Donald Pitts from May 29, 2001
www.Whitehats.com
www.arin.net
www.apnic.net
www.SecurityFocus.com
www.NetworkSolutions.com
www.Google.com
www.Incidents.org
www.snort.org
http://www.domainit.com/country-domains.htm
http://archives.neohapsis.com/archives/snort/2000-01/0222.html

http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids177&view=event

Last Updated: May 26th, 2017

Upcoming Training

Security Operations Center Summit & Training	Washington, DC	Jun 05, 2017 - Jun 12, 2017	Live Event
SANS Houston 2017	Houston, TX	Jun 05, 2017 - Jun 10, 2017	Live Event
SANS Columbia, MD 2017	Columbia, MD	Jun 26, 2017 - Jul 01, 2017	Live Event
SANS London July 2017	London, United Kingdom	Jul 03, 2017 - Jul 08, 2017	Live Event
SANSFIRE 2017	Washington, DC	Jul 22, 2017 - Jul 29, 2017	Live Event
SANSFIRE 2017 - SEC503: Intrusion Detection In-Depth	Washington, DC	Jul 24, 2017 - Jul 29, 2017	vLive
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
SANS vLive - SEC503: Intrusion Detection In-Depth	SEC503 - 201709,	Sep 11, 2017 - Oct 18, 2017	vLive
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
Baltimore September 2017 - SEC503: Intrusion Detection In-Depth	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
Community SANS Scottsdale SEC503	Scottsdale, AZ	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS October Singapore 2017	Singapore, Singapore	Oct 09, 2017 - Oct 28, 2017	Live Event
Community SANS Boston SEC503	Boston, MA	Oct 09, 2017 - Oct 14, 2017	Community SANS
Community SANS Ottawa SEC503	Ottawa, ON	Oct 16, 2017 - Oct 21, 2017	Community SANS
SANS Seattle 2017	Seattle, WA	Oct 30, 2017 - Nov 04, 2017	Live Event
SANS San Diego 2017	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	Live Event
San Diego Fall 2017 - SEC503: Intrusion Detection In-Depth	San Diego, CA	Oct 30, 2017 - Nov 04, 2017	vLive
SIEM & Tactical Analytics Summit & Training	Scottsdale, AZ	Nov 28, 2017 - Dec 05, 2017	Live Event
SANS Cyber Defense Initiative 2017	Washington, DC	Dec 12, 2017 - Dec 19, 2017	Live Event
SANS OnDemand	Online	Anytime	Self Paced
SANS SelfStudy	Books & MP3s Only	Anytime	Self Paced