Global Information Assurance Certification Paper

GCIA Practical Assignment, v3.0

Intrusion Detection In Depth

Joe Ellis

May 14, 2002

Intrusion Detection Systems: Component Architecture

Abstract
One of the many challenges faced by a security-conscious organization is the deployment and management of an intrusion detection system (IDS) for the comprehensive detection of intrusions in a dynamic environment. This paper will show the ways in which a component-based IDS allows for greater network coverage, and increased room for growth as the network and security requirements change.

Introduction
An IDS is designed to monitor computer or network resources for unwanted, and potentially adverse, activity. It is a complementary technology to intrusion prevention, which includes devices such as firewalls, network address translation (NAT) routers, and encryption for discouraging unwanted use of an organization's computing resources. Or, as Abhijit Sarmah says[1], "An IDS can be compared with a burglar alarm…the lock system protects the car…but it is the burglar alarm that detects the broken lock and raises the alarm". Development of intrusion detection systems has been ongoing since the 1970s, but it was not until the 1990s that intrusion detection became a viable commercial product, and was able to significantly advance outside of government-funded research[2].

The current state of intrusion detection systems can be described as one of rapid growth and expanding capabilities. As computers are becoming more ubiquitous and more capable, new exploits are being discovered at an accelerating pace. Extensive research for extending the capabilities of intrusion detection is being conducted in topics such as data mining, correlation, statistical analysis, attack prediction, and attack response. An IDS operates within a very dynamic environment, requiring quick and easy modifications in response to varying security policies, network topologies, bandwidth requirements, and attack methods. The software of a successful intrusion detection system must be designed to meet these diverse needs. One way of doing so is separating the software into components.

A component architecture is one in which diverse pieces of functionality can be added and removed without affecting the core system. Components can be "plugged in" to the core of the IDS as new functionality is desired or becomes available. Components can be placed in many different locations on the network. This gives users a high level of customization for using the IDS in the manner required by their particular network configuration.

Goals
The lifetime of the IDS can be greatly extended by the ability to add components as research and development produce new applications capable of detecting intrusions. Components for monitoring application-specific log files and data formats can be developed and simply added to the already functioning IDS. Components capable of processing the data in an offline fashion, such as searching for low-and-slow attacks,

statistical anomalies, patterns unnoticeable by human operators, or correlation among varying data sets can be added to the IDS as needed. As new reporting standards are defined, components can be created to "shoehorn" compliance into legacy products by mapping their output to the new standards, thus making the job of understanding attacks easier.

Each system will have its own security requirements based on the network topology, importance of devices and services on the network, and the organization's security policy. With the use of a component architecture, IDS software can be used in a larger variety of networks by allowing the user to choose from a library of components. As an added advantage, deploying only those components that are required will increase the efficiency of the IDS, allowing the IDS to perform its job better with less resources. Additionally, components themselves can be configured based upon the needs of a particular network.

The most comprehensive network monitoring is achieved by placing intrusion sensors at multiple points on the network. Sensors should be able to communicate amongst themselves, with a minimum of burden in their installation, use, and effect on network throughput. To achieve this, all sensors must contain a common interface transparent to the user. While the best monitoring of the network is distributed, the best management of the network is centralized for both ease of use and maintenance. Furthermore, data processing is made more effective and capable by storing it in one location. Network traffic data comes from many different sources such as firewalls, host-based logs, routers, and IDS sensors. Unfortunately all this information is stored in different log formats. Components can bridge these gaps by reporting back to a centralized location, each one understanding the data format of its particular sensor.

Some solutions, such as NetworkICE, involve placing a sensor on each computer within the network and having the sensor report back to a predetermined centralized server. Other tools, such as Snort, are available as a single sensor, but can be used with third-party tools such as Intrusion Vision and Demarc to support multiple remote sensors. Intrusion Vision (http://www.gd-decisionsystems.com/intrusionvision/main.html) is a meta-IDS by General Dynamics for visualizing and analyzing attacks discovered by commercial and free sensors. Demarc (http://www.demarc.com) is a tool for monitoring network servers and Snort sensors from a centralized console.

Additional new developments and advancements other than what we expect can become reality and should be planned for. By being able to treat these in a component fashion, an IDS can add to its capability without requiring a total replacement or reconfiguration. Current efforts, which can be planned for but have not yet been fully implemented, comprise a long list.

- Offline data mining (such as correlation, relational analysis, and abnormal usage analysis)
- Standards for attack reporting and descriptions

- Tighter integration with intrusion prevention, network visualization, and network device (firewalls, computers, routers) monitoring tools
- Increased bandwidth and data archival
- Automated response
- Reduction of false positives and false negatives
- Integrated network vulnerability analysis

Each of these is a piece in the larger puzzle of intrusion detection, and as such, can be broken down into another component of the IDS. By using a component architecture, pieces can be mixed and matched as more technologies become available, and as networks change.

Enablers

The component-based IDS did not grow out of a vacuum. It would not be a practical option today without the development of many prior innovations in the technology field. Some of the diverse technologies useful for developing components include Jini, RMI, SSL, and open source software. Jini is part of the Java family, a mechanism by which components can automatically discover and connect with each other on the network. The advantage of using Jini is that IP addresses of the components do not have to be hard-coded or known ahead of time. RMI used in conjunction with SSL provides the mechanism for the remote components to communicate securely across the network. Intrusion Vision has used these tools extensively. Open source software has been invaluable in allowing software tools, such as Snort, to grow quickly and dynamically. Developers have added new components as various needs arise, such as enhanced input stream processors, a variety of output formatting, and statistical analysis engines.

Examples

Snort and General Dynamics Intrusion Vision are two intrusion detection systems in existence today that stand as models of the component architecture. At its release, Snort consisted of three major components. Open source has allowed many more components to be added to Snort by developers all over the world. Intrusion Vision was created from the ground up based upon the component architecture, and, while released as closed source, includes an open API for allowing others to develop components without requiring a view into the core of the system.

Snort is based on the GNU general public license. Users of the product are allowed to freely modify and add to the source code as long as the modifications are distributed under the GNU general public license as well[3]. Since its release, many new components have been added, including a variety of stream and output processors, which can be used based on the needs of the user's network.

Snort consists of three major components: the packet decoder, the detection engine, and the logging and alerting subsystem. The packet decoder is responsible for decoding the data of each layer of the packet, from link layer to application layer. The detection

engine is a list of rules specifying certain conditions that, when found in the packet, signify an alert. Finally, the logging and alerting subsystem determines the output of Snort[4]. This is a great example of a component architecture, where each piece is an autonomous unit, not all of which are required for Snort to function- the detection engine can be omitted if one simply wishes to capture and decode packets.

A second example of component architecture is Intrusion Vision. Intrusion Vision uses Java's Jini and RMI technologies to facilitate remote component communication. Jini is responsible for automatically discovering all components on the network. Once discovered, RMI and SSL allow the components to securely communicate with a centralized manager. The centralized manager includes a database containing severity, category, and CVE/CERT number assignments for the alert.

Intrusion Vision, while a closed-source product, has an open API for the component interface. The API allows software developers to add new components, such as a sensor or correlation engine, as needed, without requiring a change to the core system. Intrusion Vision comes with an API allowing it to integrate with other commercial IDSs, but also provides the open API to allow developers to add their own support[5]. Jini provides a mechanism for this new component to be automatically discovered, wherever it may reside on the home network.

Potential Issues

A component-based IDS should not be used without consideration of a few cautionary notes. First is the amount of work involved in configuring and constantly tweaking the IDS. Intrusion detection systems themselves are complex and operate in a complex environment; a beginner cannot be expected to determine the best IDS setup without additional training and domain knowledge. Issues to consider are installation of the software on various network nodes, configuration of the components themselves, and verifying the communication between components. Significant time can be spent just on a single sensor, in an effort to reduce false positive and false negative alerts. Furthermore, a decentralized component architecture leaves the door wide open for a beginning user to easily misconfigure the system- causing the software to run slowly, generate false positives, or miss many attacks.

Components communicating across a network present a problem as well. For instance, will the components be allowed to communicate through a firewall? Will intruders be able to easily determine where the "master console" component is located and disable it, thus rendering the entire IDS inoperable? Can certain sensors be jammed or disabled in a way that is not noticeable to the administrator, perhaps by disabling its input or output interfaces? These issues need to be considered by a knowledgeable security officer to maintain system security.

Conclusion

Intrusion detection systems exist in a dynamic network environment, with rapidly expanding capabilities. Anyone wishing to produce a successful IDS must allow for as much flexibility as possible, while trying to reduce exposure to configuration and security issues. Software products existing today have shown that the idea of components can be used to successfully allow for dynamic configuration and integration of new technology without requiring a significant rewrite of the software. Until the general level of intrusion detection technology has stabilized enough to allow for a single, comprehensive solution, the advantages of an IDS built with components is significant.

References

1. Sarmah, Abhijit. "Intrusion Detection Systems: Definition, Need and Challenges" October 3, 2001. URL: http://rr.sans.org/intrusion/IDS_definition.php (May 5, 2002).
2. Bruneau, Guy. "History and Evolution of Intrusion Detection" October 13, 2001. URL: http://rr.sans.org/intrusion/evolution.php (May 5, 2002).
3. Roesch, Martin. "Snort - Lightweight Intrusion Detection for Networks" URL: http://www.snort.org/docs/lisapaper.txt (May 5, 2002).
4. GNU General Public License. June 1991. URL: http://www.gnu.org/copyleft/gpl.html(May 5, 2002).
5. General Dynamics Decision Systems. URL: http://www.gd-decisionsystems.com/intrusionvision/ (May 5, 2002).

Assignment 2: Network Detects

Detect 1: Public FTP and FTP Bounce Reconnaissance

L 03/20/02 17:41:22 FTPD:System:0002 (User=21474836485 Anonymous-->Visitor-->System) [WarFTPDControlSck::OnLoggedIn()] User logged in

c 03/20/02 17:41:24 FTPD:System:0002 (User=21474836485 Anonymous-->Visitor-->System) [WarFTPDControlSck::OnMkd()] Directory "file://C:\Program Files\War-ftpd\FTPRoot\Upload\020321013809p" created.

S 03/20/02 17:41:25 FTPD:System:0002 (User=21474836485 Anonymous-->Visitor-->System) [WarFTPDControlSck::OnPort()] (207,46,133,140,1,21): refused PORT 8C852ECF,277

O 03/20/02 17:41:26 FTPD:System:0002 (User=21474836485 Anonymous-->Visitor-->System) [WarLoginHandle::Logout()] User logged out.

1. Source of trace

My home system, running Windows 98SE and connected to the Internet via a cable modem.

2. Detect was generated by

Wu-ftp log file

3. Probability source address was spoofed

Low. To perform the attack, the attacking computer needs to complete a connection, as well as receive replies from the ftp server.

4. Description of the attack

The source IP is scanning for anonymous ftp servers. If one is found, it logs in and attempts to create a directory. Next the PORT command is issued to test for FXP, and for a possible ftp bounce. FXP allows files to be copied from one ftp server to another. FTP bounce allows an attacker to connect to any port of any remote machine.

The attack is used because an anonymous ftp server with permissions to upload files, download files, and initiate a remote connection is very useful for storing and transferring illegal binaries. FTP bounce is useful for bypassing firewalls by masquerading as a trusted host within the network.

CERT advisory describing how ftp bounce can be misused to allow an attacker to gain access to a remote machine that would not normally be available: http://www.cert.org/advisories/CA-1997-27.html.

5. Attach mechanism

Let's start with a line-by-line analysis of each entry in the log file listed above.
Log Entry 1: Attacker begins by logging in anonymously at 17:41:22.
Log Entry 2: Attacker creates a directory named 020321013809p. The numbers before the "p" are a timestamp. Now the attacker knows that directories can be created anonymously.

Log Entry 3: PORT command is issued, in an attempt to open a connection to a remote host. Note the IP address 207.46.13.140 is microsoft.com, which is rather suspicious.
Log Entry 4: PORT command failed, so user logs out.

Note the timestamp in the log entries. Since each command is performed so quickly, an automated scanner must have performed this attack. The most likely candidate is a tool called Grim Ping which is a popular scanner for public ftp sites.

6. Correlations
Numerous. There's a good analysis at http://www.eyeonsecurity.net/papers/ftpscanning.html. Log files are shown which include the strangely named directory "020612105639p" (the timestamp in the directory name is different than mine, of course) and the PORT command with the microsoft.com IP address.

7. Evidence of active targeting
None. This is most likely a wide scan for public ftp servers. Many attackers scan the well-known range of cable modem IP addresses.

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

Criticality: 1 – My home computer is not used for anything critical.
Lethality: 3 – Attack is reconnaissance. Some FTP vulnerabilities allow for more serious compromises; that was not attempted here, but now that the unprotected public ftp server has been found a more serious attack could be coming. Since the ftp server allows anyone to upload files, it could be used to house illegal programs or data.
System: 2 – The ftp server is the latest version. However, it is using a default configuration which is probably allowing for too much access.
Network: 0 – No countermeasures.

Severity = 2

9. Defensive recommendation
Restrict anonymous users from uploading files and creating directories. These activities should require a username/password. If not needed, don't allow anonymous users on the ftp server at all. Fortunately, this ftp server did not allow the PORT command to be used to connect to a remote computer. Logging performed by the ftp server can be configured to be more verbose.

10. Multiple choice test question
FTP bounce can be used to:
A) Perform a denial of service attack
B) Masquerade as a trusted computer
C) "Jam" in-transit email by reporting it as bounced
D) Crash a machine by "bouncing" packets between two ftp servers

Answer: B

References:
http://www.eyeonsecurity.net/papers/ftpscanning.html
http://www.cert.org/advisories/CA-1997-27.html

Detect 2: Grim's Ping Companion
I 03/21/02 05:23:06 FTPD:System:0002 (User=18446744073709551615) [WarFTPD::OnAccept()] Client (80.135.106.160:2555->HOME.PC:21) is connected to the FTP server.

L 03/21/02 05:23:07 FTPD:System:0002 (User=21474836485 Anonymous-->Visitor-->System) [WarFTPDControlSck::OnLoggedIn()] User logged in

r 03/21/02 05:24:15 FTPD:System:0002:data:0001 (User=21474836485 Anonymous-->Visitor-->System) [FTPDataSocket::OnTransferComplete()] Uploaded file "file://C:\Program Files\War-ftpd\FTPRoot\Upload\1mbtest.ptf". 1048578 bytes in 66.13 sec. (15.485 Kb/s)

r 03/21/02 05:24:18 FTPD:System:0002:data:0003 (User=21474836485 Anonymous-->Visitor-->System) [FTPDataSocket::OnTransferComplete()] Uploaded file "file://C:\Program Files\War-ftpd\FTPRoot\Upload\space.asp". 2648 bytes in 0.38 sec. (6.805 Kb/s)

O 03/21/02 05:24:29 FTPD:System:0002 (User=21474836485 Anonymous-->Visitor-->System) [WarLoginHandle::Logout()] User logged out.

1. Source of trace
My home system, running Windows 98SE and connected to the Internet via a cable modem.

2. Detect was generated by
Wu-ftp log file

3. Probability source address was spoofed
Low. To perform the attack, the attacking computer needs to complete a connection, as well as receive replies from the ftp server.

4. Description of the attack
This is an attack by Grim Ping with Ping Companion, a popular tool for finding and collecting data on public ftp servers. When an ftp server allowing anonymous logon is found, a 1 Megabyte file called 1mbtest.ptf is uploaded to test connection speed. Then a file named space.asp is uploaded, which allows the attacker to gather information about the ftp server and web server (if one is present). Information includes "IPs, writable directories, and OS types from Ping's log and then checks their upload access, upload speed, download access, download speed, list access, delete access, fxp access, fxp speed, and available hard drive space" (http://grimsping.cjb.net/downloads.htm)

5. Attach mechanism
Let's start with a line-by-line analysis of each entry in the log file listed above.
Log Entry 1: Attacker client connects to FTP server (3-way handshake)
Log Entry 2: Attacker begins by logging in anonymously at 03/21/02 05:23:07.
Log Entry 3: Attacker uploads 1mbtest.ptf file to test connection speed.
Log Entry 4: Attacker uploads space.asp file, used to gather information on ftp and http servers.
Log Entry 5: Attacker logs out.

6. Correlations
Numerous. There's a good analysis at http://www.eyeonsecurity.net/papers/ftpscanning.html. Log files are shown which include the use of the files 1mbtest.ptf and space.asp. Also included is a log file showing an HTTP request for space.asp. Had my computer been running a webserver, tcpdump, or an IDS, I probably would have seen the HTTP request.

7. Evidence of active targeting
None. Grim Ping is generally used as a wide scan for anonymous ftp servers. Many attackers scan the well-known range of cable modem IP addresses.

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

Criticality: 1 – My home computer is not used for anything critical.
Lethality: 3 – Attack is reconnaissance. Grim's ping companion discovers a lot of information about the host.
System: 2 – The ftp server is the latest version. However, it is using a default configuration which is probably allowing for too much access.
Network: 0 – No countermeasures.

Severity = 2

9. Defensive recommendation
Restrict anonymous users from uploading files and creating directories. These activities should require a username/password. If not needed, don't allow anonymous users on the ftp server at all.

10. Multiple choice test question
Grim's Ping companion is a well-known public ftp reconnaissance tool which can NOT be used to discover which of the following about the host:
A) IP address
B) Root password
C) ftp permissions (such as anonymous file upload or deletion)
D) fxp access

Answer: B

References:
http://www.eyeonsecurity.net/papers/ftpscanning.html
http://grimsping.cjb.net/downloads.htm

Detect 3: SYN FIN Scan
[**] [111:13:1] spp_stream4: STEALTH ACTIVITY (SYN FIN scan) detection [**]
03/22-18:36:51.693037 211.218.149.56:21 -> A.B.C.D:21 TCP TTL:23 TOS:0x0 ID:39426 IpLen:20 DgmLen:40 ******SF Seq: 0x228CF381 Ack: 0x1B70181E Win: 0x404 TcpLen: 20

[**] [100:1:1] spp_portscan: PORTSCAN DETECTED to port21 from 211.218.149.56 (STEALTH) [**]
03/22-18:36:47.450000

[**] [100:2:1] spp_portscan: portscan status from 211.218.149.56: 2 connections across 1 hosts: TCP(2),UDP(0) STEALTH [**] 03/22-18:36:51.950000

1. Source of trace
My home system, running Windows 98SE and connected to the Internet via a cable modem.

2. Detect was generated by
Snort1.8.1: spp_stream4 and spp_portscan processors.

3. Probability source address was spoofed
Low. The attacker needs a reply to gather information on the target computer.

4. Description of the attack
This packet was most likely generated by synscan 1.5/1.6, which is known to use a constant id of 39426 and uses equal source and destination ports, seen in Snort's output listed above. Donald Smith has a very helpful analysis of synscan and related tools in his GCIA paper at http://www.giac.org/practical/donald_smith_gcia.doc.
Many networks block ICMP echo replies, thus blocking attacker's attempts at finding open ports. "SynFin" crafted packets are considered a stealthy alternative method for mapping ports and have been in use since the late 90's.

5. Attach mechanism
This is a reconnaissance attack. The attacker is using synscan to determine if port 21(ftp) is active on the target host. If so, the attacker may attempt an anonymous login to the ftp server. It could also be used for OS fingerprinting, by capturing and analyzing the host's response to the malformed packet.

This is definitely a crafted packet. Note that the Syn and Fin flags are both set, which is invalid. Also notice that only one packet was sent- normally we would expect a series of retries. Normally one would not expect to see a packet with a SYN flag set originating from a low port like 21(ftp).

6. Correlations
Numerous. Donald Smith's GIAC paper lists many captured SynFin packets, and the Intrusion Detection Track "IDS Signatures and Analysis" book has many examples.

7. Evidence of active targeting
Difficult to determine. My home PC had been running an anonymous FTP server (see detects 1 and 2, which occurred on previous days). Luckily for me, on the day of this attack the ftp server was not running. Had the previous scans resulted in my computer being placed on some sort of hacker's list of public ftp servers? A Google search did not turn up my IP address, but there's really no way to know. Given the popularity of crafted SYNFIN packets, this scan was probably from an unrelated source.

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

Criticality: 1 – My home computer is not used for anything critical.
Lethality: 2 – Attack is reconnaissance, which failed to turn up a public ftp server.
System: 3 – System is not running any services, and is using the latest version of Snort monitor for attacks.
Network: 0 – No countermeasures.

Severity = 0

9. Defensive recommendation
Consider using a firewall or software to block TCP packets with "SF" flags set, in addition to (outgoing) ICMP Echo replies, to limit the ability of others to gain information on the host.

10. Multiple choice test question
A TCP packet with Syn and Fin flags set is useful for:
A) Crashing vulnerable TCP/IP stack implementations.
B) Starting the 3-way handshake process.
C) Stealth scanning and finding of vulnerabilities
D) Stealth traceroute

Answer: C

References:
http://www.giac.org/practical/donald_smith_gcia.doc

Detect 4: Scanning for BIND exploit

Jan 20 20:39:36 10.1.3.15 ipmon[1514]: 20:39:36.581154 fxp0 @48:2 b 202.107.236.180,20 -> A.B.C.D,12300 PR tcp len 20 40 -S IN

Jan 20 20:39:52 10.1.3.15 ipmon[1514]: 20:39:52.204625 fxp0 @48:2 b 202.107.236.180,20 -> A.B.C.D,12300 PR tcp len 20 40 -S IN

Jan 20 20:40:08 10.1.3.15 ipmon[1514]: 20:40:07.756299 fxp0 @48:2 b 202.107.236.180,20 -> A.B.C.D,12300 PR tcp len 20 40 -S IN

Jan 20 20:40:24 10.1.3.15 ipmon[1514]: 20:40:23.371742 fxp0 @48:2 b 202.107.236.180,20 -> A.B.C.D,12300 PR tcp len 20 40 -S IN

Jan 20 20:40:39 10.1.3.15 ipmon[1514]: 20:40:38.954448 fxp0 @48:2 b 202.107.236.180,20 -> A.B.C.D,12300 PR tcp len 20 40 -S IN

Jan 20 20:40:55 10.1.3.15 ipmon[1514]: 20:40:54.570916 fxp0 @48:2 b 202.107.236.180,20 -> A.B.C.D,12300 PR tcp len 20 40 -S IN

etc….

1. Source of trace
Posted by Erik Fichtner of Server Vault, Inc. on Incidents.org archive:
http://www.incidents.org/archives/intrusions/msg02843.html

2. Detect was generated by
ipmon, a UNIX tool which monitors /dev/ipl for packets in binary format, and decodes them into a human readable format (listed above).

3. Probability source address was spoofed
Low. Attacker needs a reply to determine if target has been compromised by a previous BIND exploit.

4. Description of the attack
BIND is a widely used implementation of DNS, known to have many significant vulnerabilities.
A search for the keyword "BIND" on CVE returns 22 search results. A search for the keyword "BIND" returns 161 results on CERT.
A very good analysis of BIND 8 vulnerabilities is listed on the BIND web page at http://www.isc.org/products/BIND/bind-security.html.

5. Attach mechanism
The logs above show the attacker sequentially scanning port 12300 over a large number of addresses within the same class C network. Notice that each scan is 15 seconds apart. This could be "low and slow" to foil an IDS, or it could just be latency since the source address is from China.

This is not an actual attack. The scanning IP is searching for previously compromised hosts, most likely by someone else.

6. Correlations
This link (http://boudicca.tux.org/mhonarc/ma-linux/2001-Feb/msg00569.html) shows the results of running 'lsof -i -n -P' on a compromised host. Of particular interest is this line:

*in.amdq 27400 root 3u IPv4 2695932 TCP *:12300 (LISTEN)*

which shows a process called in.amdq listening on port 12300, the same port that is being scanned for in the capture shown above. According to the link, in.amdq is "A customized ssh daemon of sorts that allows anyone to connect as root, or so it appears".

7. Evidence of active targeting
None. The attacker is scanning a large number of hosts in a sequential manner.

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

Criticality: 5 – I don't know anything about the network, but I can assume that it must contain something important, since it is being protected by a security company called Server Vault, Inc.
Lethality: 4 – This is reconnaissance, rather than an actual attack. However, if a compromised host is discovered, then the attacker can gain root. I have no idea if this network contains a DNS server running BIND; if not, then this would not be considered lethal.
System: 4 – Server Vault, Inc. provides managed hosting. Thus, they control the servers, which I will assume have been secured.
Network: 4 – Server Vault, Inc. claims 99.999% reliability, and are obviously a very security-conscious organization.

Severity: 1

9. Defensive recommendation
The posted logs did not show any hosts returning a response from the port 12300 probe, so hopefully the scan was in vain; this can be verified by running an internal scan on port 12300. If BIND is running on a server, the administrators are strongly recommended to upgrade to the latest version (currently 9.2.0) and check the system logs for any unusual activity. Verify that the network monitoring system in use will record an internal response from a port 12300 probe.

10. Multiple choice test question
BIND is an implementation of what protocol:
A) Berkeley TCP/IP
B) DHCP
C) VPN
D) DNS

Answer: D

References:
http://www.incidents.org/archives/intrusions/msg02843.html
http://boudicca.tux.org/mhonarc/ma-linux/2001-Feb/msg00569.html
http://www.isc.org/products/BIND/bind-security.html

Detect 5: MS SQL Server worm

"50494" "30Jan2002" "10:16:03" "log" "1433" " A.B.C.D" "tcp" "49027"
"50526" "30Jan2002" "10:16:04" "log" "1433" " A.B.C.D" "tcp" "49028"
"53781" "30Jan2002" "10:18:02" "log" "1433" " A.B.C.D" "tcp" "49030"
"53807" "30Jan2002" "10:18:03" "log" "1433" " A.B.C.D" "tcp" "49031"

"71704" "30Jan2002" "11:06:44" "log" "1433" "E.F.G.H" "tcp" "3582"
"71705" "30Jan2002" "11:06:44" "log" "1433" " E.F.G.H " "tcp" "3585"
"71706" "30Jan2002" "11:06:44" "log" "1433" " E.F.G.H " "tcp" "3583"

…many more scans from 212.237.144.11…

1. Source of trace
Jill Treu posted this trace to Incidents.org:
http://www.incidents.org/archives/intrusions/msg02818.html

2. Detect was generated by
Firewall of unknown type.

3. Probability source address was spoofed
Low. Attacker is probing for Microsoft SQL servers.

4. Description of the attack
Microsoft SQL Server is a database. It is vulnerable because the default installation has an Administrator account with no password. A good description is located here: http://securityresponse.symantec.com/avcenter/venc/data/w32.cblade.worm.html. The attack is performed by w32.cblade.worm. When a successful infection occurs, the worm reports itself to an IRC server, and uses ftp to download an executable, which it the runs. Only SQL servers with a null administrator password are affected.

5. Attach mechanism
The worm is searching for default installations of MS SQL server, on port 1433. If found, the target will be infected by the worm.

6. Correlations
Many sites describe this worm and its use of port 1433:
http://www.osborne.com/virus_alert/voyager_alpha.shtml

http://securityresponse.symantec.com/avcenter/venc/data/w32.cblade.worm.html

7. Evidence of active targeting
None. The worm is just scanning the internet for vulnerable MS SQL servers.

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

Criticality: 4 – I don't know what sort of hosts are on the network, but I can assume that it is something important if they have a firewall.
Lethality: 4 – If a vulnerable MS SQL installation is found, it will be infected.
System: 3 – An attempt was made to scan many hosts on the network, but I do not know if the firewall blocked the scan.
Network: 4 – Since a firewall log was posted, I do know that at least a firewall is in place and its log files are being analyzed.

Severity: 1

9. Defensive recommendation
The firewall should be configured to block port 1433 unless there is a specific reason to keep it open. Also verify that, if MS SQL Server is required, the administrator has set a good password. Remove MS SQL Server if it is not needed.

10. Multiple choice test question
The w32.cblade worm operates by infecting MS SQL Server. Once infected, it reports itself to an IRC server and uses ftp to download an executable. Without knowing how the worm operates, choose the vulnerability that the worm could NOT use to initially infect the server and report back to an IRC server:
A) SQL server allows execution of SQL commands submitted on port 1592.
B) Buffer overflow causing the host machine to crash.
C) Default installation has no password for the administrator account.
D) Using a null field in the SQL command, allowing the worm to execute arbitrary code.

Answer: B (If the server crashed, the worm would not be able to execute)

References
http://www.incidents.org/archives/intrusions/msg02818.html
http://securityresponse.symantec.com/avcenter/venc/data/w32.cblade.worm.html
http://www.osborne.com/virus_alert/voyager_alpha.shtml

Assignment 3: "Analyze This" Scenario
This section is an analysis of log files captured by Snort from March 2nd to March 6th, 2002. The logs are in 3 sets of files:

653,972 Snort alerts(not including port scans) in files alert.0203[02-06]
338 Out of Spec packets in files oos.Mar.[02-06].2002
3,044,488 Scan alerts in: scans.0203[02-06].

Executive Analysis

This network appears to have no security in place, other than extensive logging of traffic by the Snort IDS. Due to a lack of implemented security policy, all hosts on the network are open to attack. The network administrators should immediately decide on a security policy and begin implementation of such. Preliminary recommendations include adding a properly configured firewall to protect from intruders, and network address translation to minimize successful reconnaissance. Configuration of the firewall will be dependent upon the security policy. The issues raised from an analysis of network logs captured over a five-day period should help the administrators in determining a security policy by first focusing on the current problems of the network.

Before beginning a detailed analysis, a number of statements can be made about the hosts on this network. The administrators will want to consider the acceptability of a number of services currently in use on the network:

- The use of P2P applications such as Kazaa and Morpheus.
- Anonymous FTP servers
- IRC
- "Unregulated" web servers, particularly those using Microsoft IIS, which has recently been responsible for the spread of the malicious worms Code Red and Nimda.
- Instant Messenger and ICQ
- Streaming media

Alerts Analysis

The following table shows all of the captured alerts sorted by occurrence from the alert files dated March 2nd to March 6th, 2002.

All logged alerts

242878 connect to 515 from inside
99715 MISC Large UDP Packet
88424 spp_http_decode: IIS Unicode attack detected
73327 SMB Name Wildcard
37524 SNMP public access
35855 ICMP Echo Request L3retriever Ping

15665 INFO MSN IM Chat data
12996 spp_http_decode: CGI Null Byte attack detected
12911 High port 65535 udp - possible Red Worm - traffic
8106 Watchlist 000220 IL-ISDNNET-990517
6604 ICMP Echo Request Nmap or HPING2
4287 ICMP Fragment Reassembly Time Exceeded
2941 INFO Inbound GNUTella Connect request
2130 WEB-IIS view source via translate header
1383 Tiny Fragments - Possible Hostile Activity
1303 ICMP Router Selection
1220 WEB-CGI scriptalias access
1105 WEB-MISC Attempt to execute cmd
791 INFO Outbound GNUTella Connect request
544 FTP DoS ftpd globbing
495 WEB-IIS _vti_inf access
489 WEB-FRONTPAGE _vti_rpc access
354 INFO FTP anonymous FTP
331 Null scan!
246 Watchlist 000222 NET-NCFC
231 NMAP TCP ping!
171 INFO napster login
152 ICMP Echo Request Delphi-Piette Windows
126 WEB-MISC http directory traversal
123 ICMP traceroute
122 MYPARTY - Possible My Party infection
122 ICMP Echo Request Windows
99 TFTP - Internal UDP connection to external tftp server
93 SCAN Proxy attempt
90 INFO Possible IRC Access
86 ICMP Destination Unreachable (Communication Administratively Prohibited)
83 Back Orifice
71 INFO Inbound GNUTella Connect accept
51 Possible trojan server activity
50 Port 55850 tcp - Possible myserver activity - ref. 010313-1
41 SCAN Synscan Portscan ID 19104
41 ICMP Destination Unreachable (Protocol Unreachable)
40 WEB-MISC 403 Forbidden
39 WEB-CGI rsh access
38 SCAN FIN
35 MISC traceroute
35 INFO - Possible Squid Scan
33 SUNRPC highport access!
33 Attempted Sun RPC high port access
28 WEB-MISC compaq nsight directory traversal
28 WEB-MISC ICQ Webfront HTTP DOS
27 High port 65535 tcp - possible Red Worm - traffic

26 EXPLOIT NTPDX buffer overflow
20 IDS552/web-iis_IIS ISAPI Overflow ida nosize
20 EXPLOIT x86 NOOP
17 RPC tcp traffic contains bin_sh
14 Queso fingerprint
11 WEB-IIS Unauthorized IP Access Attempt
11 Incomplete Packet Fragments Discarded
10 TCP SRC and DST outside network
9 INFO Outbound GNUTella Connect accept
9 EXPLOIT x86 setuid 0
8 x86 NOOP - unicode BUFFER OVERFLOW ATTACK
8 Russia Dynamo - SANS Flash 28-jul-00
8 Port 55850 udp - Possible myserver activity - ref. 010313-1
8 INFO Napster Client Data
7 WEB-CGI redirect access
7 RFB - Possible WinVNC - 010708-1
7 EXPLOIT x86 setgid 0
6 WEB-MISC cd..
6 ICMP Echo Request BSDtype
6 FTP CWD / - possible warez site
6 EXPLOIT x86 stealth noop
5 WEB-MISC /....
5 WEB-IIS asp-dot attempt
3 WEB-MISC webdav search access
3 EXPLOIT x86 NOPS
3 BACKDOOR NetMetro Incoming Traffic
2 WEB-MISC whisker head
2 WEB-MISC Lotus Domino directory traversal
2 WEB-CGI formmail access
2 TFTP - External UDP connection to internal tftp server
2 RPC udp traffic contains bin sh
2 INFO napster new user login
2 ICMP Address Mask Reply
1 WEB-MISC ftp attempt
1 WEB-FRONTPAGE author.exe access
1 ICMP Echo Request CyberKit 2.2 Windows

The following table shows the top ten source and destination hosts for the alert logs (not including port scans). These addresses will be covered in the following alerts analysis section.

Top 10 Alert Destination Hosts	Top 10 Alert Source Hosts
241517 MY.NET.150.198	39908 61.150.5.19
40048 MY.NET.153.206	24388 MY.NET.153.119
33680 MY.NET.11.7	16897 MY.NET.70.177
31679 MY.NET.11.6	15360 MY.NET.11.7
18074 MY.NET.153.184	14483 MY.NET.11.6
11330 209.10.239.135	12471 MY.NET.153.136
10329 MY.NET.11.5	12145 MY.NET.153.146
7634 MY.NET.5.96	11739 MY.NET.153.126
7371 MY.NET.151.114	10988 MY.NET.153.113
6197 MY.NET.152.109	10879 209.177.65.18

The alert files contain 653,972 alerts, not including port scans, and 88 unique types of alerts. With so many alerts, it is not feasible (from a cost standpoint) that we analyze every single one. Some method of prioritizing is necessary. The top 10 alerts alone make up 627,401, or 96% of the logged alerts. We can maximize our time by initially concentrating on the top occurring alerts, thereby handling over 90% of the discovered alerts. However, we cannot assume that the most common alerts are the indeed the most serious, so we cannot simply analyze the most frequent alerts and ignore the rest. A good place to start is in resolving the "cruft" before drilling down to the less-numerous alerts. Consequently, the first goal will be to reduce the most numerous alerts through an understanding of their characteristics, especially if the most numerous alerts aren't the most serious. Vigilance in removing the distractions presented by a large amount of unimportant data will go a long way towards minimizing the amount of time needed to analyze alerts, and maximize time spent focusing on the most important, destructive problems. This can be accomplished through better configuration of the IDS rules, or by handling the problem itself; for instance, by removing a noisy virus from the infected host.

Less-numerous attacks will be given priority as well, based on criteria such as the severity of the alert type, its correlation with other alerts, and type of the service. A prioritization process can be automated through the use of meta-IDS tools such as Intrusion Vision, or by using the latest version of Snort, which provides a method for assigning severity.

Alert: Connect to 515 from inside

Top 5 of 158 Sources	All Destination
23785 MY.NET.153.119	241469 MY.NET.150.198:515
11301 MY.NET.153.126	1268 MY.NET.1.63:515
10102 MY.NET.153.136	141 MY.NET.153.152:515
8807 MY.NET.153.146	
8627 MY.NET.153.164	

Note that four source IP addresses in this table, MY.NET.153.119, MY.NET.153.126, MY.NET.153.136, and MY.NET.153.146, are in the overall top ten source IP address list. The majority of recorded alerts for these four IP addresses is "connect to 515 from inside". MY.NET.150.198 is the top alert destination on the entire network.

This alert is produced by a Snort rule watching for a connection attempt to port 515 from inside the MY.NET network. Port 515 is used by the UNIX service LPRng, which provides print spooling. LPRng listens on port 515 for connections from clients such as lpr, and once a host makes a connection it can enter a variety of print-related commands. The service has a known vulnerability, posted in October 2000, which allows attackers to execute any command, according to CVE-2000-0917. CERT Advisory CA-2000-22 provides detailed information on this as well. The Red Had Ramen worm is known to exploit this vulnerability. The default Snort rules contain two very specific rules for catching an LPR exploit:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 515 (msg:"EXPLOIT
redhat 7.0 lprd overflow"; flags: A+; content:"|58 58 58 58
25 2E 31 37 32 75 25 33 30 30 24 6E|"; classtype:attempted-
admin; sid:302; rev:1;)

alert tcp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"EXPLOIT
named tsig infoleak"; content: "|AB CD 09 80 00 00 00 01 00
00 00 00 00 00 01 00 01 20 20 20 20 02 61|";
reference:cve,CAN-2000-0010; reference:bugtraq,2302;
reference:arachnids,482; classtype:attempted-admin;
sid:303; rev:3;)
```

If they are enabled, we can assume that an exploit was not attempted. Since all source ports were located within the network, this is not indicative of a scan for the exploit. Furthermore, the source connection was initiated from an anonymous high port, which is consistent with the behavior of a client application. The large number of internal hosts going to a small number of internal suggests that the 3 destination hosts are acting as network print spoolers.

However, there is another possibility. The following is an example set from the alert logs, showing a pattern that occurs many times in the alert log. The interesting thing to note is the large number of connection attempts within a short time period. The time between connections is often a tenth or a hundredth of a second. This seems like an inordinate amount of print connections repeatedly coming from the same host in a very short time. Also note that all connections are coming from the same IP and same port. The data is indicative of a denial of service attack, where the attacker has chosen to spoof the source address as one residing within the network. A "connect" would refer to a SYN packet being sent to the destination to initiate a connection. The packets have many source hosts, while the destination involves only 3 hosts, with the largest number by far going to one host, MY.NET.150.198. The attacker may be hoping that by sending a large number of connections to a port with a known service, the listening service will cause the machine to crash. This was reported here:

http://archives.neohapsis.com/archives/bugtraq/2000-09/0212.html, as an attack against winCOM LPD service (vendor is at http://www.wincom.com/).

3/02-10:15:27.277844 [**] connect to 515 from inside [**] MY.NET.152.158:2702 -> MY.NET.150.198:515
03/02-10:15:27.279075 [**] connect to 515 from inside [**] MY.NET.152.158:2702 -> MY.NET.150.198:515
03/02-10:15:27.280306 [**] connect to 515 from inside [**] MY.NET.152.158:2702 -> MY.NET.150.198:515
03/02-10:15:27.281536 [**] connect to 515 from inside [**] MY.NET.152.158:2702 -> MY.NET.150.198:515
03/02-10:15:27.282779 [**] connect to 515 from inside [**] MY.NET.152.158:2702 -> MY.NET.150.198:515
03/02-10:15:27.283997 [**] connect to 515 from inside [**] MY.NET.152.158:2702 -> MY.NET.150.198:515
03/02-10:15:27.285227 [**] connect to 515 from inside [**] MY.NET.152.158:2702 -> MY.NET.150.198:515
03/02-10:15:27.286458 [**] connect to 515 from inside [**] MY.NET.152.158:2702 -> MY.NET.150.198:515
03/02-10:15:27.287688 [**] connect to 515 from inside [**] MY.NET.152.158:2702 -> MY.NET.150.198:515
etc….

From the given logs files it is not possible to determine exactly what is occurring with this alert. If a firewall or NAT were put in place, we could eliminate the possibility of an externally generated denial of service attack. The 3 destinations should be verified as having the latest version of LPRng. Verify that the Snort rules listed above are enabled; and, if possible, make the existing rule which triggered the alert more specific. An improperly configured Snort rule could be causing the rule to trigger more often than desired. A rule set to trigger on a destination port of 515 would trigger on the connection attempt, but also during normal traffic between the server/client. That would explain the small time period between alerts, as essentially every non-fragmented packet would be logged.

Alert: MISC Large UDP Packet

Top 5 of 30 Source IP	Top 5 of 23 Destination IP
39905 61.150.5.19	39934 MY.NET.153.206
10879 209.177.65.18	18038 MY.NET.153.184
9353 63.250.205.8	5695 MY.NET.153.185
8728 63.250.205.44	5199 MY.NET.153.187
5792 202.30.244.134	4772 MY.NET.153.136

Note that two of the source IP addresses, 61.150.5.19 and 209.177.65.18, are in the overall top ten source IPs. Also, two of the destination IP addresses, MY.NET.153.206 and MY.NET.153.184, are in the overall top ten destination IPs.

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

Attack tools such as Stacheldraht can generate a large UDP packet, or it could simply be normal traffic where the size of the UDP packet happens to be larger than 4000 bytes. A valid use of UDP is for applications requiring large amounts of data to be transferred where data integrity is not that important. This includes games and streaming media traffic. Performing a DNS lookup on these source addresses may give us helpful information as to why these packets are entering our network.

Whois performed on www.geektools.com:

61.150.5.19:
inetnum: 61.150.0.0 - 61.150.31.255
netname: SNXIAN
descr: xi'an
data branch,XIAN CITY SHAANXI PROVINCE
country: CN
admin-c: WWN1-AP
tech-c: WWN1-AP
mnt-by: MAINT-CHINANET-SHAANXI
mnt-lower: MAINT-CN-SNXIAN
changed: ipadm@public.xa.sn.cn 20010309 source: APNIC
person: WANG WEI NA address: Xi Xin street 90# XIAN
country: CN phone: +8629-724-1554 fax-no: +8629-324-4305
e-mail: xaipadm@public.xa.sn.cn nic-hdl: WWN1-AP
mnt-by: MAINT-CN-SNXIAN changed: wwn@public.xa.sn.cn 20001127 source: APNIC

This is an address from China, owning an IP address range which has been documented in other GIAC papers: http://www.giac.org/practical/Rick_Yuen_GCIA.doc. I can't read Chinese, so I don't know what the web site contains.

209.177.65.18:
NCI Technologies, Inc.
(NETBLK-NCI-BLK-1) PO Box 376 Philipsburg, PA 16866 US
Netname: NCI-BLK-1
Netblock: 209.177.64.0 - 209.177.95.255
Maintainer: NCIT Coordinator: Bezilla, Daniel B. (DB1208-ARIN)
dan@NCITECH.COM +1-814-342-7030 ext. 7102 (FAX) 814-342-7033

NCI Technologies, Inc(http://nci01web.ncitech.com/index.html) is a company which provides "communications solutions". Further research shows IP telephony as one of provided services. IP telephony allows users to make regular phone calls over the Internet, and could certainly use UDP as its transport mechanism.

63.250.205.8 and 63.250.205.44:
Yahoo! Broadcast Services, Inc. (NETBLK-NETBLK2-YAHOOBS)
701 First Avenue Sunnyvale, California 94089 US
Netname: NETBLK2-YAHOOBS Netblock: 63.250.192.0 - 63.250.223.255
Maintainer: YAHO
Coordinator: Admin, Netblock (NA258-ARIN) netblockadmin@yahoo-inc.com 1-408-349-5555

According to their website http://broadcast.yahoo.com/home.html,
"Yahoo! Broadcast offers a wide variety of on-demand audio and video content, from space shuttle launches to full-length movies. We have sports heroes in action, bands performing in concert, and TV shows that haven't been on the air in years." Once again, we see streaming media, a legitimate use of UDP.

202.30.244.134:
[ISP member ORG information]
Org Name : ONSE Telecom Service Name : SHINBIRO
Org Address : 192-2 Kumi-dong Bundang-ku Sungnam-si

[Admin Contact Information]
Name : Sungin Kim Phone : 031-738-6411 Fax : 031-738-6430 E-Mail : ip@mgate.shinbiro.com

[IP Manager Contact Information] Name : Sungin Kim Phone : 031-738-6411
Fax : 031-738-6430 E-mail : ipadm@mgate.shinbiro.com

[Hacking/SPAM Contact Information]
Name : Sungin Kim Phone : 031-738-6411 Fax : 031-738-6430 E-mail : ip@mgate.shinbiro.com

Shinbiro (www.shinbiro.com) appears to be a Korean web portal. Unfortunately, the web site is in Korean, but it is not far-fetched to assume that it provides services similar to Yahoo!, including streaming media and games.

Four of the top five source addresses (209.177.65.18, 63.250.205.8, 63.250.205.44, 202.30.244.134) can be linked to services making valid uses of UDP. The other address, 61.150.5.19, could investigated through the help of someone who can speak Chinese. The use of these services should be addressed in the security policy, but this does not constitute an attack on the network.

Alert: spp_http_decode: IIS Unicode attack detected
Alert: WEB-MISC Attempt to execute cmd
Alert: IDS552/web-iis_IIS ISAPI Overflow ida nosize
Alert: High port 65535 udp - possible Red Worm - traffic

Combined, these alert types make up 102,460 alerts, or 15% of the non-port scan detects. These alerts are well-known signatures of Code Red and NIMDA (http://www.incidents.org/react/code_redII.php, and http://www.incidents.org/react/nimda.pdf) worm infections of Microsoft's IIS web server. Once infected, the host will scan port 80, looking for other web servers to infect.

The *WEB-MISC Attempt to execute cmd* and *web-iis_IIS ISAPI Overflow ida nosize* alerts all came from an external network; these hosts are probably infected and are randomly scanning for other hosts to infect.

The following table list hosts on MY.NET found to be the source of traffic to destination port 80, which generated the *spp_http_decode: IIS Unicode attack detected* alert. This alert is generated by Nimda; the hosts should be checked for Nimda infection. The hosts corresponding to a low number alerts are most likely false positives, and any host not running IIS will not be affected. The entire list of alerts is included here to illustrate the implicit difficulties in using a signature-based IDS like Snort. Rules which are not, or can not be, made specific enough will result in many false positives. The end result being increased time and complexity in tracking down potential intrusions.

4778 MY.NET.153.123	542 MY.NET.152.11	83 MY.NET.153.162
4701 MY.NET.153.169	532 MY.NET.152.10	83 MY.NET.151.73
4063 MY.NET.153.202	494 MY.NET.153.165	80 MY.NET.152.21
3659 MY.NET.153.171	474 MY.NET.153.164	78 MY.NET.153.207
3440 MY.NET.153.143	442 MY.NET.153.190	75 MY.NET.152.144
3249 MY.NET.153.142	441 MY.NET.152.164	70 MY.NET.152.171
3064 MY.NET.153.106	438 MY.NET.151.108	69 MY.NET.152.213
2780 MY.NET.153.113	437 MY.NET.153.126	68 MY.NET.153.203
2678 MY.NET.153.111	435 MY.NET.153.159	64 MY.NET.153.205
2505 MY.NET.153.210	420 MY.NET.153.180	56 MY.NET.152.216
2496 MY.NET.153.197	414 MY.NET.153.174	55 MY.NET.152.172
2335 MY.NET.153.136	398 MY.NET.153.154	53 MY.NET.151.64
1919 MY.NET.153.115	374 MY.NET.153.198	52 MY.NET.88.194
1890 MY.NET.153.211	373 MY.NET.153.120	45 MY.NET.153.208
1556 MY.NET.153.110	344 MY.NET.153.187	44 MY.NET.152.17
1521 MY.NET.153.112	339 MY.NET.153.122	37 MY.NET.152.158
1517 MY.NET.153.108	332 MY.NET.152.182	33 MY.NET.152.22
1499 MY.NET.88.254	296 MY.NET.153.149	31 MY.NET.153.168
1460 MY.NET.153.182	289 MY.NET.153.150	28 MY.NET.152.249
1239 MY.NET.153.161	270 MY.NET.153.163	26 MY.NET.152.186
1215 MY.NET.153.105	256 MY.NET.153.193	25 MY.NET.253.10
1127 MY.NET.152.244	238 MY.NET.153.184	24 MY.NET.88.230
1084 MY.NET.153.127	218 MY.NET.153.179	22 MY.NET.88.222
1071 MY.NET.153.107	208 MY.NET.153.172	17 MY.NET.150.103
1045 MY.NET.153.114	202 MY.NET.152.250	15 MY.NET.152.48
967 MY.NET.153.167	197 MY.NET.153.125	14 MY.NET.88.232
918 MY.NET.153.141	186 MY.NET.88.189	14 MY.NET.153.71
893 MY.NET.153.194	181 MY.NET.153.199	9 MY.NET.150.165
890 MY.NET.88.148	170 MY.NET.153.146	8 MY.NET.152.142
857 MY.NET.153.181	166 MY.NET.153.176	6 MY.NET.150.235
839 MY.NET.152.247	162 MY.NET.153.118	5 MY.NET.152.159
829 MY.NET.153.147	158 MY.NET.152.19	5 MY.NET.152.148
813 MY.NET.153.153	155 MY.NET.153.177	4 MY.NET.152.20
784 MY.NET.153.135	134 MY.NET.152.214	3 MY.NET.153.206
776 MY.NET.153.185	129 MY.NET.153.175	3 MY.NET.153.160
773 MY.NET.153.189	128 MY.NET.153.157	3 MY.NET.152.180
759 MY.NET.153.148	127 MY.NET.153.109	3 MY.NET.152.179
743 MY.NET.152.248	119 MY.NET.153.188	3 MY.NET.152.177
703 MY.NET.152.251	117 MY.NET.150.97	3 MY.NET.152.176
697 MY.NET.153.170	114 MY.NET.152.165	3 MY.NET.152.145
663 MY.NET.153.117	111 MY.NET.153.166	2 MY.NET.153.195
650 MY.NET.153.46	108 MY.NET.153.124	2 MY.NET.153.137
616 MY.NET.153.121	108 MY.NET.150.77	1 MY.NET.97.197
600 MY.NET.153.119	95 MY.NET.153.200	1 MY.NET.88.132
589 MY.NET.153.152	92 MY.NET.153.196	1 MY.NET.153.45
559 MY.NET.153.144	88 MY.NET.152.174	1 MY.NET.153.209

The following hosts on MY.NET generated the *High port 65535 udp - possible Red Worm – traffic* alert. These hosts should be checked for Red Worm infection.

3839 MY.NET.6.52	207 MY.NET.6.53
3062 MY.NET.6.49	179 MY.NET.6.60
2482 MY.NET.6.48	100 MY.NET.60.43
1817 MY.NET.6.50	98 MY.NET.6.45

Alert: SMB Name Wildcard

Top Source hosts	Top Destination Hosts
15360 MY.NET.11.7	15311 MY.NET.11.7
14483 MY.NET.11.6	14514 MY.NET.11.6
5182 MY.NET.11.5	5162 MY.NET.11.5

A normal Windows Netbios request for host information, such as workstation name, domain, and users logged in, triggers this alert. This alert has two malicious sources: intruders using Netbios can discover information about the network, and a worm known as network.vbs. The worm begins with the typical Netbios request, then proceeds to connect with port 139 and attempts to mount drive "C" with a null password. Fortunately, this behavior on port 139 was not found in the alert logs, though that may be because there isn't a Snort rule in place to detect this worm. Further analysis showed that all of the source and destination hosts reside in MY.NET. The strong correlation between identical source and destination hosts in the address list above leads me to believe that these are Windows servers. This can be classified as normal behavior by the Windows hosts on the network. It may be helpful to refine the Snort rule to only trigger when the source or destination address is external to this network. See the explanation of *Alert: ICMP Echo Request L3retriever Ping* below for more information.

Sources:

http://www.sans.org/newlook/resources/IDFAQ/port_137.htm.
http://archives.neohapsis.com/archives/snort/2000-01/0222.html.

Alert: ICMP Echo Request L3retriever Ping

Top 3 of 159 Source Hosts	Top 3 of 21 Destination Hosts
1083 MY.NET.152.163	15412 MY.NET.11.7
857 MY.NET.152.166	14582 MY.NET.11.6
848 MY.NET.152.168	5167 MY.NET.11.5

The Snort rule is quite specific, note the interesting payload (ABCDEFGHIJKLMNOPQRSTUVWABCDEFGHI) of the packet:

alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP L3retriever Ping"; content: "ABCDEFGHIJKLMNOPQRSTUVWABCDEFGHI"; itype: 8; icode: 0; depth: 32; reference:arachnids,311; classtype:attempted-recon; sid:466; rev:1;)

http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids311&view=event says that the alert is caused by either the L3 "Retriever 1.5" security scanner, or by a Windows 2000 host. An analysis of the alert logs shows an interesting correlation between this alert and the *SMB Name Wildcard* alert. Notice that the top destination hosts listed here are also the top source and destination hosts of the *SMB Name Wildcard* alert. The following pattern is seen repeatedly in the alert logs:

03/02-00:06:57.496871 [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.177 -> MY.NET.11.7
03/02-00:06:57.497427 [**] SMB Name Wildcard [**] MY.NET.152.177:137 -> MY.NET.11.7:137
03/02-00:06:57.497671 [**] SMB Name Wildcard [**] MY.NET.11.7:137 -> MY.NET.152.177:137

Based on this pattern, MY.NET.152.177 is most likely a Windows 2000 host, querying MY.NET.11.7 for Netbios domain information. A correlation is in this statement from the Whitehats URL mentioned above: "This type of ICMP ping seems to be also generated by (plain) Win2K host talking to Win2K domain controllers." --nnposter

Alert: SNMP public access

Top Source hosts	Top Destination Hosts
16878 MY.NET.70.177	7319 MY.NET.151.114
10511 MY.NET.150.198	6197 MY.NET.152.109
2444 MY.NET.150.41	3363 MY.NET.5.247

A recent advisory at http://www.cert.org/advisories/CA-2002-03.htmldescribes problems with many SNMP implementations, which can cause devices to crash or allow an attacker to gain access to the device. This alert is caused when a connection attempt is made to SNMP listener port 161, with default community string on “public”. The community string is sort of a password, but most devices default to “public” and their users don’t change it. If a SNMP agent is on the host, its reply with notify the scanner of its existence, thereby opening itself to the attacks mentioned above. Hosts using SNMP can include printers, routers, workstations, and any other device where remote monitoring is desirable. Resolution of the advisory is difficult at best, as most affected devices will require a software upgrade. The best solution is to use a firewall to block the 161 and 162 ports, and change the default community strings from “public” to something more

secure. Since all of the many source and destination addresses are internal, this is most likely normal traffic, and does not constitute a serious alert.

Alert: INFO MSN IM Chat data

Top 3 of 115 Source Hosts	Top 3 of 111 Destination Hosts
1545 MY.NET.153.107	1025 MY.NET.153.107
660 MY.NET.153.46	527 64.4.12.160
575 MY.NET.153.211	495 MY.NET.153.211

MSN IM, or Microsoft Networks Instant Messenger, is an application which allows users to trade text messages and data files, similar to ICQ and AOL Instant Messenger. The application connects to a central server, seen as 64.4.12.X in the alert logs. A Whois lookup resolves this address to:
MS Hotmail (NETBLK-HOTMAIL)
1065 La Avenida Mountain View, CA 94043 US
Netname: HOTMAIL

MY.NET.153.107 and MY.NET.153.211 appear to be very active users of this application. A vulnerability does exist for this service, though the worm currently doesn't engage in any mischief (other than replicate itself and use up system memory): http://www.symantec.com/avcenter/venc/data/w32.choke.worm.html

Usage of this service on the network should be reviewed.

Alert: spp_http_decode: CGI Null Byte attack detected

Top Source Hosts	Top Destination Hosts
5170 MY.NET.153.171	11330 209.10.239.135
3168 MY.NET.153.146	780 209.185.162.149
1782 MY.NET.153.197	204 209.143.193.79

CGI, or Common Gateway Interface, allows the web server to return dynamic data to the user, by processing information on the web server and returning it in HTML format to the user's web browser. If the CGI form should contain a %00, this alert will trigger. The problem is that %00 could be valid, if that is how the form is constructed. This alert can trigger many false positives, and can be turned off by adding the "-cginull" option to the line "preprocessor http_decode: " in Snort's alert.ids file.

The top destination host, involving over 90% of the alerts, resolves to Globix Corporation http://www.globix.com/. From their website: "Globix Corporation is a leader in complex hosting, network services and advanced Internet applications for enterprises that seek performance advantages and cost-efficiency through the strategic outsourcing of their IT infrastructure." http://209.10.239.135 turns out to be iFilm, a movie guide which allows

for searches on a movie database and current movie times. It is almost certain that they are using CGI on their website. This alert is a false positive.

The only destination within the MY.NET network is MY.NET.5.96. This host should be checked for a web server using CGI scripts- if so, check the host for any problems.

Alert: Watchlist 000220 IL-ISDNNET-990517

Top 5 Source Host:Port	Top 5 Destination Host:Port
6470 212.179.35.118:80	1778 MY.NET.153.166:1478
633 212.179.27.176:80	1175 MY.NET.153.141:1140
332 212.179.3.218:80	1171 MY.NET.153.142:1811
192 212.179.35.118:1214	884 MY.NET.153.152:1224
102 212.179.35.119:1214	567 MY.NET.153.163:1145

Watchlists are for monitoring traffic with IP addresses from a particular network. This watchlist appears to be for 212.179.0.0/17.
Whois shows:
inetnum: 212.179.0.0 - 212.179.1.255
netname: AREL-NET
descr: arel-net
country: IL
dmin-c: TP1233-RIPE
tech-c: TP1233-RIPE
status: ASSIGNED PA
notify: hostmaster@isdn.net.il
mnt-by: RIPE-NCC-NONE-MNT
changed: hostmaster@isdn.net.il 19990624 source:
RIPE route: 212.179.0.0/17
descr: ISDN Net Ltd.
origin: AS8551
notify: hostmaster@isdn.net.il
mnt-by: AS8551-MNT changed: hostmaster@isdn.net.il 19990610
source: RIPE
person: Tomer Peer
address: Bezeq International
address: 40 Hashakham St.
address: Petakh Tiqwah Israel phone: +972 3 9257761
e-mail: hostmaster@isdn.net.il nic-hdl: TP1233-RIPE
changed: registrar@ns.il 19991113
source: RIPE

A look at the top ports in use shows 80 (webserver), and 1214 (P2P). Port 1214 is used by P2P applications Kazaa, Morpheus, and Grokster. These applications allow users to share and download files after performing a keyword search. They are very popular, and notorious for copyright problems. Their use on the network should be reviewed.

Alert: Web-IIS view source via translate header
This is an attempt to view scripts on the server, through a vulnerability in the default installation of Microsoft IIS 5.0. It could be a false positive, as it is a valid use of WebDAV. Microsoft has a patch at:
http://download.microsoft.com/download/win2000platform/Patch/Q256888/NT5/EN-US/Q256888_W2K_SP1_x86_en.EXE

Source: http://www.whitehats.com/info/IDS305.

Alert: Possible trojan server activity
This is not a standard Snort rule, so I cannot determine what caused this event. Many trojans, once they have infected a host, will automatically report their existence back to a fixed location, such as an IRC channel, webserver, or email address. The trojan will also listen on a specific port, waiting for a command to activate some action.

Alert: INFO Outbound GNUTella Connect Request
Alert: INFO Napster login
Alert: INFO Inbound GNUTella Connect accept
Alert: INFO OutboundGNUTella Connect attempt
Alert: INFO Napster Client Data
Alert: INFO Napster new user login
These alerts refer to the use of P2P file-sharing services, such as Napster, Morpheus, Kazaa, and the GnuTella network. These services allow users to search for, download, and host files. A large number of copyrighted materials are being traded in this manner, resulting in litigation from the copyright owners. Use of these services on the network should be reviewed.

Alert: FTP DoS ftpd globbing
Alert: INFO FTP anonymous FTP
Alert: TFTP- Internal UDP connection to external tftp server
Alert: TFTP – External UDP connection to internal tftp server
These alerts deal with the use of FTP servers on the MY.NET network. FTP, or File Transfer Protocol, is a method for allowing clients to connect with a server and download or upload files. The problem with running an FTP server is twofold, in that the software contains well-known vulnerabilities, and it often allows anonymous users to upload and download files on the host.

The first alert, *DoS ftpd globbing*, is an attempt to crash the server by issuing a command like "LIST */../*../*/../*/../*". This will often overload the FTP server software, causing it to crash. An analysis of the log files shows that the following hosts were involved in this alert, meaning that they are most likely running an FTP server: MY.NET.150.145, MY.NET.151.63, MY.NET.152.173, MY.NET.153.146, MY.NET.153.170.

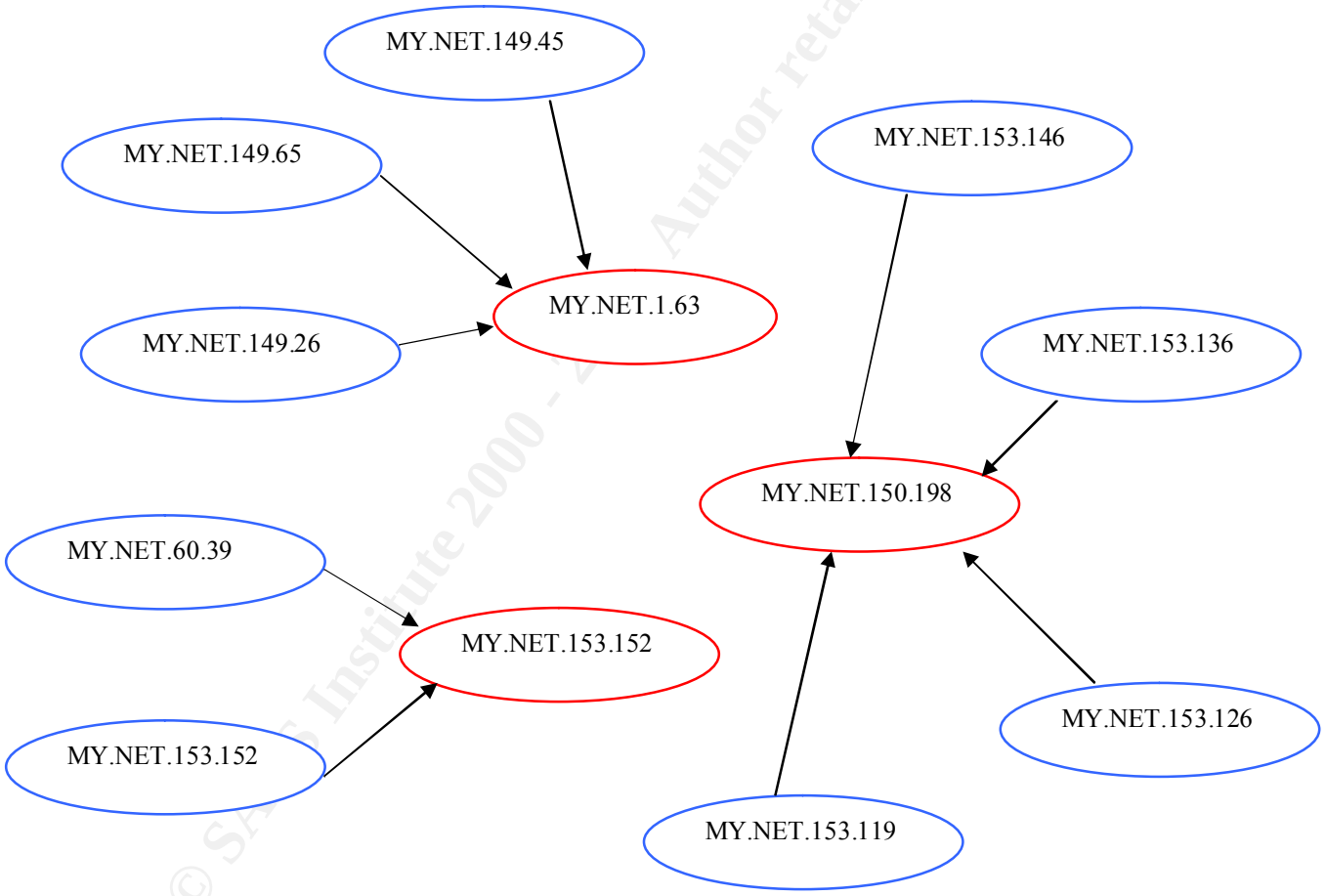
The second alert, *INFO FTP anonymous FTP*, triggers when an connection is made to an FTP server allowing anonymous access from the external network. The acceptability of allowing this service on the network should be reviewed. An analysis of the logs files shows that the following hosts scanned MY.NET for an anonymous ftp server.172.191.15.203, 65.93.244.26, 80.14.111.233, 62.163.117.199, 172.190.217.6, 217.86.27.141. Apparently nothing was found, as a scan of this type, when successful, is usually followed by a known set of commands being sent to the FTP server, which would have generated alerts by the standard Snort rules. None of the source/destination hosts involved with this alert were connected with the *DoS ftpd globbing* alert.

The TFTP alerts refer to a function of ftp servers which allows the server to open a connection to any remote IP address/port. This is dangerous, because it allows someone to indirectly attack another host through your own server, and it allows for the user to pretend to be acting from a trusted source (known as an ftp bounce). An internal network may allow any actions to occur if the source address is on the internal network (the source address would be the ftp server). A valid use of these commands is with FXP, which allows files to be copied from one FTP server to another.

Link Analysis

I would like to show a graph of the "connect to 515 from inside" alert, to illustrate a key finding that better characterizes this alert. The visual aid makes the client/server architecture quite apparent, which is what one would expect for the use of network printers. Through this aid, I realized that this detect, the most numerous in the alert log files, was normal and not a cause for alarm. The arrows always point towards the destination hosts because the alert rule (I assume) is set to trigger only on connection attempts. All destination/source addresses for this alert are shown, with the exception of 149 additional source hosts to MY.NET.150.198.

Source hosts are in blue. Destination hosts are in red.



Scans Analysis

The scan files were created by Snort's "spp_portscan" processor. Individual Scan alerts are generally not that useful. Their use is in characterizing "normal" behavior and monitoring for a change in overall patterns. It is useful to maintain a list of top scans, source host, destination host, and destination ports. Any change should be investigated as possibly suspicious behavior. A previously unseen host in the list may indicate a potential compromise, or an increase in scanning for a certain port may indicate a new vulnerability. If a top source or destination is on the external network, this would definitely be cause for immediate investigation.

Scans by occurrence

2489674	UDP
554015	SYN
352	VECNA
192	NULL
162	UNKNOWN
27	NOACK
25	INVALIDACK
19	FIN
2	XMAS
2	SPAU
1	SYNFIN
1	NMAPID
1	FULLXMAS

Top 10 Destination ports

407611	80
373438	7001
218729	7000
199537	53
136839	0
100342	137
84982	514
56686	6970
39776	88
36921	7003

Top 10 destinations

116152	MY.NET.1.3
84856	MY.NET.1.7
80966	MY.NET.1.4
62513	MY.NET.6.45
57478	MY.NET.11.7
54625	MY.NET.11.6
54066	MY.NET.60.43
38375	MY.NET.153.172
30851	MY.NET.5.55
30761	MY.NET.153.209

Top 10 source

465252	MY.NET.60.43
218814	MY.NET.6.52
197386	MY.NET.6.49
173042	MY.NET.6.48
132676	MY.NET.6.45
121148	MY.NET.6.50
61962	MY.NET.6.60
54320	MY.NET.6.53
25806	MY.NET.11.6
25454	MY.NET.153.175

The most common scans by far are UDP and SYN. UDP is an alternate transport mechanism to TCP, where delivery of each packet is not guaranteed. UDP packets are very common for streaming applications, of which the alerts analysis showed many are in use. SYN scans, where the TCP SYN flag is set, is a reconnaissance technique. However, malicious use of this is easy to hide, given the large number of valid SYN entries.

The other scan types are directly related to which TCP flags are set. The following table shows a key (from http://www.giac.org/practical/Christof_Voemel_GCIA.txt):

VECNA	One of the following: P, U, PU, FP, FU
NULL	None of SFRPAU
UNKNOWN	See spp_portscan.c source code
NOACK	A flag is missing
INVALIDACK	ACK set, not 'normal', no SPAU or FULLXMAS
FIN	F flag
XMAS	FPU flags
SPAU	SPAU flags
SYNFIN	SF flags
NMAPID	SFPU flags
FULLXMAS	SFRPAU flags

Many of the unusual flag combinations are attributed to fingerprint scanning by tools such as nmap and queso. These tools will send unusual (invalid) packets to a host, and compare its response to a known list of responses to determine information about the host, such as operating system type.

Popular destination ports include:

80: webserver

7001: Freak88 trojan, and by the afs3 fileserver.

7000: streaming video, use of this seen in the alerts analysis. Also used by some trojans, notably subseven 2.1, and by the afs3 fileserver.

53: DNS. Scanning for DNS is not a surprise, given their importance and many BIND vulnerabilities.

0: This is valid, occuring when a packet is fragmented.

137: Used by NETBIOS, a Windows naming service.

514: Used by RCP backdoor trojan, syslog, and BSD shell.

6970: Gatecrasher trojan

88: Kerberos: a network authentication protocol, using keys to provide secure authentication of client/server hosts.

7003: afs3 fileserver

It appears that the afs3 fileserver is in use on the network. If not, there may be some real trojan problems. Use of port 6970, by hosts MY.NET.151.MY.NET.151.70, MY.NET.151.125, and MY.NET.151.85 to AOL addresses 205.188.228.X should be investigated for a possible trojan.

OOS Analysis

The Out of Spec(OOS) files contains a list of out of spec packets received from external sources.

All Source	All Destination	All Destination Ports
8 65.28.222.108	9 MY.NET.153.175	13 6346
8 150.176.136.121	8 MY.NET.152.11	6 2418
4 67.112.202.91	4 MY.NET.88.162	5 0
3 193.232.252.34	4 MY.NET.150.41	3 6348
1 80.24.115.162	3 MY.NET.88.222	3 1214
1 80.11.140.100	2 MY.NET.150.209	2 80
1 68.50.154.196	2 MY.NET.150.145	1 64490
1 64.166.85.237	1 MY.NET.5.96	1 6442
1 62.7.15.62	1 MY.NET.153.185	1 27960
1 62.30.110.169	1 MY.NET.150.226	1 1970
1 24.232.159.104	1 MY.NET.150.133	
1 24.144.35.30		
1 217.228.124.20		
1 213.241.35.129		
1 203.59.228.98		
1 132.66.95.92		
1 129.118.174.34		

Of the 36 recorded packets, 15 have the TCP flags 21S***** set. This corresponds to SYN and reserved bits. The reserved bits are used for Explicit Congestion

Notification(ECN). The source of these packets is initiating a connection to a host on our network, and specifying that it supports ECN. This is a valid use of the TCP flags, but was probably recorded because some hacker tools (Queso) use this combination of flags. Examination of the alert logs shows that these packets correspond to the use of Gnutella on port 6346, a P2P application for sharing files.

Removing the "21S*****" packets and re-analyzing the OOS files produces different results:

All Source	All Destination	All Destination Ports
8 150.176.136.121	8 MY.NET.152.11	6 2418
4 67.112.202.91	4 MY.NET.88.162	5 0
1 80.24.115.162	3 MY.NET.150.41	3 6346
1 80.11.140.100	2 MY.NET.150.209	2 1214
1 68.50.154.196	1 MY.NET.5.96	1 80
1 64.166.85.237	1 MY.NET.153.185	1 64490
1 62.7.15.62	1 MY.NET.150.145	1 6442
1 24.232.159.104	1 MY.NET.150.133	1 27960
1 24.144.35.30		1 1970
1 217.228.124.20		
1 132.66.95.92		

Packets received from the top source, 150.176.136.121, are all going to the top destination, MY.NET.152.11, using destination ports 2418 and 6346(Gnutella). Again, we see strange (but different) packets involved with the use of Gnutella. These packets occurred on March 5th, between 20:41 and 20:58.

The packet capture of this traffic is very odd, showing all sorts of different, incompatible flags being set, as well as suspicious sequence numbers.

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/05-20:41:09.520266 150.176.136.121:228 -> MY.NET.152.11:2418
TCP TTL:109 TOS:0x0 ID:37889  DF
21SFRP** Seq: 0x18CA017C   Ack: 0x71023D4C   Win: 0x5010
TCP Options => EOL EOL EOL EOL EOL EOL WS: 1 NOP TS: 0 0 EOL EOL EOL EOL EOL EOL
 EOL EOL

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/05-20:45:03.042927 150.176.136.121:128 -> MY.NET.152.11:2418
TCP TTL:109 TOS:0x0 ID:29206  DF
*1SF**A* Seq: 0x18CA017D   Ack: 0x88983D4F   Win: 0x5010
88 98 3D 4F 25 93 50 10 B6 D2 B0 43 00 00 00 00  ..=O%.P....C....
00 00                                            ..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/05-20:48:23.895467 150.176.136.121:2427 -> MY.NET.152.11:6346
TCP TTL:109 TOS:0x0 ID:11566  DF
21S*R*** Seq: 0x17ECA1E   Ack: 0x3D50   Win: 0x5010
```

```
TCP Options => EOL EOL EOL EOL EOL EOL SackOK NOP NOP TS: 0 0 EOL EOL EOL EOL EO
L EOL EOL EOL EOL EOL EOL EOL EOL EOL EOL EOL EOL EOL EOL

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/05-20:48:49.099396 150.176.136.121:6 -> MY.NET.152.11:2418
TCP TTL:109 TOS:0x0 ID:28721  DF
*1SF*P*U Seq: 0x18CA017E   Ack: 0xE1D93D51   Win: 0x5010
E1 D9 3D 51 28 AB 50 10 B3 06 57 B3 00 00 00 00  ..=Q(.P...W.....
00 00                                            ..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/05-20:49:10.136060 150.176.136.121:0 -> MY.NET.152.11:2418
TCP TTL:109 TOS:0x0 ID:54323  DF
*1SFRP** Seq: 0x18CA017E   Ack: 0xF6123D51   Win: 0x5010
00 00 00 00 00 00                                ......

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/05-20:56:01.533708 150.176.136.121:2427 -> MY.NET.152.11:6346
TCP TTL:109 TOS:0x0 ID:8036  DF
21SF**AU Seq: 0x9F0185   Ack: 0xBA853D57   Win: 0x5010
TCP Options => EOL EOL EOL EOL EOL EOL SackOK NOP NOP SackOK

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
03/05-20:58:17.749970 150.176.136.121:0 -> MY.NET.152.11:2418
TCP TTL:109 TOS:0x0 ID:52341  DF
21*FRP** Seq: 0x18CA0187   Ack: 0xE1F3D5C   Win: 0x5010
0E 1F 3D 5C 2E CD 50 10 B5 C2 22 7C 00 00 00 00  ..=\..P..."|....
00 00                                            ..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

03/05-20:58:50.000427 150.176.136.121:0 -> MY.NET.152.11:2418
TCP TTL:109 TOS:0x0 ID:43385  DF
21*F*PA* Seq: 0x18CA0187   Ack: 0x6ACB3D5C   Win: 0x5018
TCP Options => EOL EOL
```

I could not find an exact match for this behavior, other than theorizing that a tool such as nmap or queso is being used to fingerprint MY.NET.152.11. During the same time period, the source host triggered many scanning type alerts on MY.NET: stealth portscans, null scans, and scan FINs. The best option is to attempt contacting the owner of the host and perhaps receive useful information as to what the host is, or what it is trying to do. If this traffic continues, consider blocking the host for a short time period, or at least placing it on a watchlist. A whois query provides the following contact information:

Florida Information Resource Network
325 W. Gaines Street B1-14 FEC
Tallahassee, FL 32399

Maintainer: FIRN
Coordinator: Florida Information Resource Network
hostmaster@POPMAIL.FIRN.EDU 850-487-8657

MY.NET.152.11 should be checked for possible compromise, as it was the source address for various alerts, including Red Worm/Nimda alerts, Gnutella, and Nmap.

Analysis Process

Manipulation of the data files consisted of using UNIX-based text processing tools wc, uniq, grep, sed, awk, cat, cut, and sort. Knowledge on the use of these tools is required. I would recommend this method, if only for the learning opportunity, as these tools have many applications and are certainly worthwhile addition to one's toolbox.

The research and writing portions of the analysis utilized a Windows PC. Research on the attacks was accomplished through the extensive use of many Internet resources, including: www.google.com, www.snort.org, http://cve.mitre.org/cve/, www.cert.org, www.whitehats.com, and www.geektools.com.

The following commands, adapted from Gregory Lojan's GCIA paper, were used to process the data.

Alerts

Put all alert files into one for easier processing:

```
> cat alert.* > alert.txt
```

Cleanup the alerts (removes "start" and "stop" comments and portscans):

```
> cat alert.txt | sed -e 's/\[\*\*\]/\*\*/g' > sed.out
```

Creation of alerts sorted by occurrence:

```
> cat sed.out |awk -F'**' '{print $3}' | sort | uniq -c |
sort -nr > top10
```

Creation of source addresses sorted by occurrence:

```
> cat sed.out |awk -F'**' '{print $5}' | awk -F '->'
'{print $1}' | awk -F ':' '{print $1}' | sort | uniq -c |
sort -nr > top10src
```

Creation of destination addresses sorted by occurrence:

```
> cat sed.out |awk -F'**' '{print $5}' | awk -F '->'
'{print $2}' | awk -F ':' '{print $1}' | sort | uniq -c |
sort -nr > top10dst
```

Script file for finding source and destination addresses of hosts involved with a particular alert, sorted by occurrence:

```
grep 'CGI Null Byte'  sed.out  |awk -F'**' '{print $5}' |
awk –F '->' '{print $1}'    | awk –F ':' ' {print $1}' |
sort | uniq -c | sort -nr > src.txt

grep 'CGI Null Byte'  sed.out  |awk -F'**' '{print $5}' |
awk –F '->' '{print $2}'    | awk –F ':' ' {print $1}' |
sort | uniq -c | sort -nr > dst.txt

cat src.txt dst.txt > sorted.txt
```

Scans

Cleanup the scans:

```
> grep  'Mar' scans.txt |awk '{print $4" " $6" "$7" " $8}'
allscans
```

Creation of top scans sorted by occurrence:

```
> cat allscans | awk '{print $3}' | sort | uniq -c | sort -
nr > top10
```

Creation of scans by source sorted by occurrence:

```
> awk '{print $1}' allscans | awk -F':' '{print $1}' |sort
| uniq -c | sort -nr >src_scans
```

Creation of scans by destination sorted by occurrence:

```
> awk '{print $2}' allscans | awk -F':' '{print $1}' |sort
| uniq -c | sort -nr >dst_scans
```

Creation of scans by destination port sorted by occurrence:

```
> awk '{print $2}' allscans | awk -F':' '{print $2}' |sort
| uniq -c | sort -nr > dst_port
```

Out of Spec (OOS)

These scripts are based on Mike Poor's GCIA paper.

Creation of destination addresses sorted by occurrence:

```
> grep "..\/..\-..\:..\:"  oos.txt | cut -d \> -f 2 | cut -
d \: -f 1 | sed s/\ //g | sort | uniq -c | sort -nr >
dst_ips
```

Creation of destination ports sorted by occurrence:

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

Creation of source addresses sorted by occurrence:

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

References

1. Sarmah, Abhijit. "Intrusion Detection Systems: Definition, Need and Challenges" October 3, 2001. URL: http://rr.sans.org/intrusion/IDS_definition.php (May 5, 2002).
2. Bruneau, Guy. "History and Evolution of Intrusion Detection" October 13, 2001. URL: http://rr.sans.org/intrusion/evolution.php (May 5, 2002).
3. Roesch, Martin. "Snort - Lightweight Intrusion Detection for Networks" URL: http://www.snort.org/docs/lisapaper.txt (May 5, 2002).
4. GNU General Public License. June 1991. URL: http://www.gnu.org/copyleft/gpl.html(May 5, 2002).
5. General Dynamics Decision Systems. URL: http://www.gd-decisionsystems.com/intrusionvision/ (May 5, 2002).
6. Bruneau, Guy. "History and Evolution of Intrusion Detection" October 13, 2001.
7. Obscure. "When your server ends up on a Warez site". URL: http://www.eyeonsecurity.net/papers/ftpscanning.html (May 5, 2002).
8. "CERT Advisory CA-1997-27 FTP Bounce". April 03, 2002. URL: http://www.cert.org/advisories/CA-1997-27.html. (May 5, 2002).
9. "Grim's Ping". URL: http://grimsping.cjb.net/downloads.htm (May 5, 2002)
10. Smith, Donald. "GIAC assignments for GCIDS". URL: http://www.giac.org/practical/donald_smith_gcia.doc (May 5, 2002).
11. Fichtner, Erik. "BIND exploit backdoor scanning". Jan 21, 2002. URL: http://www.incidents.org/archives/intrusions/msg02843.html (May 5, 2002).
12. "BIND vulnerabilities". URL: http://www.isc.org/products/BIND/bind-security.html (May 5, 2002).
13. Olsen, Jim. "BIND compromise". Feb 20, 2001. URL: http://boudicca.tux.org/mhonarc/ma-linux/2001-Feb/msg00569.html (May 5, 2002).
14. Treu, Jim. "Port 1433 Scans". Jan 18, 2002. URL: http://www.incidents.org/archives/intrusions/msg02818.html (May 5, 2002).
15. Symantec. "W32.Cblade.Worm". April 15, 2002. URL: http://securityresponse.symantec.com/avcenter/venc/data/w32.cblade.worm.html (May 5, 2002).
16. Slade, Robert. "Latest Virus Alert!" URL: http://www.osborne.com/virus_alert/voyager_alpha.shtml (May 5, 2002).
17. "CVE 2000-0917". March 9, 2002. URL: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0917 (May 5, 2002).
18. "CERT® Advisory CA-2000-22 Input Validation Problems in LPRng". December 12, 2002. URL: http://www.cert.org/advisories/CA-2000-22.html (May 5, 2002).
19. Grundl, Peter. "VIGILANTE-2000013: WinCOM LPD DoS". September 19, 2001. URL: http://archives.neohapsis.com/archives/bugtraq/2000-09/0212.html (May 5, 2002).
20. Yuen, Rick. "GIAC Intrusion Detection, Practical Assignment". October 11, 2001. URL: http://www.giac.org/practical/Rick_Yuen_GCIA.doc (May 5, 2002).
21. The SANS Institute. "Code Red II Worm Analysis Update". August 7, 2002. URL: http://www.incidents.org/react/code_redII.php (May 5, 2002).

22. Alexander, Bryce. "Port 137 Scan". May 10, 2000. URL: http://www.sans.org/newlook/resources/IDFAQ/port_137.htm (May 5, 2002).
23. Forster, Jim. "Re: [snort]'SMB Name Wildcard". January 17, 2000. URL: http://archives.neohapsis.com/archives/snort/2000-01/0222.html (May 5, 2002).
24. Whitehats, Inc. "IDS311/SCAN_PING-SCANNER-L3RETRIEVER" URL: http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids311&view=event (May 5, 2002).
25. "CERT® Advisory CA-2002-03 Multiple Vulnerabilities in Many Implementations of the Simple Network Management Protocol (SNMP)". April 28, 2002. URL: http://www.cert.org/advisories/CA-2002-03.html (May 5, 2002).
26. Symatec. "W32.Choke.Worm". April 15, 2002. URL: http://www.symantec.com/avcenter/venc/data/w32.choke.worm.html (May 5, 2002).
27. Whitehats, Inc. "IDS305/WEB-IIS_HTTP-IIS_TRANSLATE_F ". URL: http://www.whitehats.com/info/IDS305 (May 5, 2002).
28. Voemel, Christof. "SANS Intrusion Detection Practical". September 7, 2001. URL: http://www.giac.org/practical/Christof_Voemel_GCIA.txt (May 5, 2002).
29. Lojan, Gregory. "GIAC Intrusion Detection In Depth". August 18, 2001. URL: http://www.giac.org/practical/Gregory_Lajon_GCIA.txt (May 5, 2002)
30. Poor, Mike. "Intrusion Detection In Depth". August 18, 2001. URL: http://www.giac.org/practical/Mike_Poor_GCIA.txt (May 5, 2002).