SANS

GIAC Certified Intrusion Analyst (GCIA) Practical

Jalal Moloo

SANS Network Security 2001: October 15-22, 2001 (San Diego, CA)

Version 3.0

Contents

Assignment 1 - Describe the state of Intrusion Detection

Topic Overview

The purpose of this white paper is to describe the importance of host based Intrusion Detection System (IDS). It also outlines various open sources host based IDS used in the industry. I am currently faced with a challenge of recommending the best commercial host based IDS for my employer. After reviewing several host based IDS currently available in the industry, I recommend using Entercept host based IDS. This paper also explains in detail the architecture of Entercept Host based IDS and how it protects host systems.

IDS Background

The proliferation of Internet has made all networked systems widely accessible to anyone in the world. As companies become more Internet-reliant the same systems by the virtue of being widely accessible by anyone world tends to become vulnerable to attacks. These systems, which tends to be crown jewels in the e-commerce era of business needs to be protected against any vulnerable exploits. Thus securing these crown jewels is an important task. To protect these systems from any security threats, various commercial vendors have brought forward various security technologies to the market, such as firewalls, encryption and authentication, and access control lists. These technologies, although providing a certain measure of security, contain certain limitations that may allow attackers to get past them.

We build fortress of Firewall's to protect our private network from being exposed to the Internet but as claimed by a lot of surveys most threat comes from within an internal network compared to the external attacks. Firewall is a perimeter device that separates the untrusted public network with the private trusted internal network. Firewall does not inspect any payload data except in case were it is passed through some content filtering devices to filter out certain type of traffic. Thus Firewall inspects & detects the traffic based on rule set defined. To complement to this role of inspecting and detecting network traffic based on header information, IDS can be deployed to examine the payload. This payload information can be run through various decode engine to determine the traffic type. IDS are generally designed for this task and are more efficient in decoding traffic through various engines without interrupting the traffic flow. In an overall security framework one needs to build a mechanism that prevents, tracks, record and detect the traffic flow for any anomalous behavior. Firewall configured with its rules/policy complemented with IDS can fully provide this security framework of tracking, recording and inspecting/detecting traffic flow, along with decoding the traffic to decipher and log anomalous activity. The basic purpose of IDS is to listen to the network traffic and watch for any anomalous behavior in the traffic flow. There are several solutions available for network based IDS. Several open source systems i.e. snort, shadow do an excellent job. There are also several commercial product available i.e. Cisco's IDS Netranger and ISS RealSecure to name a few.

The IDS used to protect the Network operates at a similar fashion as a burglar alarm system that you can put in your windows & doors of your house. Just as the alarm detects a burglar, IDS detects attacks against the Network. They detect any intrusive activity that enters your environment by examining network traffic, host's logs, systems calls, and any other area that

would signal an attack. Having said that, on the networking aspect there are several documented ways to evade these networks based IDS. Thus once its past the network the attack basically compromises the host which contains the information wealth.
Reference: http://www.robertgraham.com/pubs/network-intrusion-detection.html
http://www.insecure.org/presentations/CanSecWest01/
http://www.insecure.org/nmap/OSDEM_Presentation/

An IDS can also be deployed on a host system, i.e. a Host, which actually host the crown jewel of any organizations, i.e. corporation, Govt. agencies, etc. This method however is more pro-active in its approach. The systems can monitor and respond to any normal traffic, and denies access to any anomalous activity. Being proactive in nature, it's considered more as an Intrusion Prevention system than Intrusion Detection.

There are two approaches to a host based IDS, i.e. host wrappers/personal firewalls and agent-based software. Both these are effective in maintaining the integrity of the file system as well as watching for any suspicious processes.

Host wrappers/personal firewalls can be configured to monitor traffic flow as well as login attempts. Programs such as TCP Wrapper, adds a layer of security by intercepting calls and determining whether or not, the service will be allowed to run. This tool can be downloaded from http://ciac.llnl.gov/ciac/ToolsUnixNetSec.html.

Other Firewall software such as ZoneAlarm for Windows and Ipchains/IPTables on *nix platform can also be configured for access control. Few other security tools such as the Abacus Project tools i.e. HostSentry can monitor login attempts and LogSentry monitors system logs for any security violations.

There are also several commercial host based intrusion prevention systems. One such host based Intrusion prevention system is the Entercept. Entercept is agents based software that reports alerts back to a console of any anomalous activity. Entercept protects the operating system, applications and system resources by identifying and preventing both known as well as unknown attacks. This software agent basically shims its self between the TCP/IP stack and the kernel, watching all the system calls placed to the kernel from the network. What this actually indicates is that it actually installs itself adjacent to the Operating System and intercepts any calls made into the OS. It also has a signature-based database to reference any anomalous behavior. If an attack is found, Entercept can take pre-emptive action to protect the system.

What does the Entercept protects?

- **Operating System**

Entercept protects the operating system from any hostile activity, by shielding critical files, registry, and other system resources. It also prevents attackers from gaining unauthorized access to servers.

Entercept prevents any perpetrator to take advantage of the Operating systems vulnerabilities and gain root/admin access. This agent software also prevents the systems from buffer overflow exploit, which currently is the biggest security concern in the industry.

Entercept monitors the code, which is about to be executed by the OS. It then denies access to the kernel memory space if the code is generated by a buffer overflow. However if the code is generated as a result of normal application it will allow the code to be processed. Thus it protects any host from being compromised as a result of buffer overflow. According to CERT of Carnegie Mellon University, over 60% of their alerts deals with buffer overflow exploits. Please refer http://www.cert.org.

- **Application/Communication layer**

Entercept protects applications by monitoring the application behavior pattern. If an attacker compromises an application, it will deviate from the normal behavior. Entercept prevent these abnormal behaviors and shields the system from being compromised. Entercept Web agent intercepts and analyzes incoming HTTP requests and blocks any malicious access to the web server.

How does Entercept protect?

- **System calls**

To protect the core operating system from being damaged by any malicious programs the operating system architecture separates the code executed from users and the code executed by the operating system itself. To achieve this objective the processor includes a mode bit that specifies if the processor is executing kernel-mode code or user-mode code. When the mode bit is set, the processor then is executing user-mode code. Thus when user-mode code is being executed, the processor hardware prevents any access to kernel memory space. In the instance when user-mode code tries to access kernel memory space, the processor generates an illegal access exception.

The only interface between user-mode to kernel mode is the system call table. Once installed, Entercept adjust the entries in the system call table, pointing towards the Entercept's kernel mode drivers. This puts Entercept entries in the command chain. System calls are now intercepted by Entercept and analyzed to determine if access to kernel mode should be allowed. An Entercept kernel driver then calls the original kernel functions.

- **Behavioral rules**

Entercept's powerful behavioral rules set prevents servers from being attacked. This rule set also prevents from new as well as previously known attacks. Entercept is not only signature based but it also analyzes traffic based on rule set. If there are no signatures found in the database, it enforces correct behavior pattern and prevent new attacks from succeeding.

- **HTTP Interception**

Entercept's Web Server agent monitors and examines the incoming HTTP requests. If these requests are malicious in nature, Entercept then goes ahead and blocks the request. It sends an error code to the offending browser program.

In an SSL based environment the front-end processing on a web server includes the encrypting/decrypting process. The Web based Entercept agent generally sits between the SSL front-end engine and the back end-processing engine. Once the traffic is normalized by SSL this clear communication has to traverse through this Web agents analysis process, on its way to the back end-processing engine. If any malicious code is found it is then blocked by this agent. Moreover since this Web agent resides after the front-end processing engine it is not susceptible to IDS evasion technique most commonly used by an attacker. As Entercept generally resides after the http request has been converted to normalized clear-text traffic, it receives the request as "htr" extension. Using this HTTP interception it blocks the attacks before the Web server processes them.

Functionality:

Entercept has two components:

Agent – This basically detects and protects the systems against any attack.
Console – This management console is used to monitor and configure agents. It is used also to generate reports and is a source where all the agents send their notification. Console can also be used to push new signature updates to the agents.

Thus as mentioned Entercept has two components Console and an Agent. All control of the Entercept product is done via this GUI based Windows console. There are two main services to Entercept that serves the internal services: Watchdog and Notification Manager. There is also a Server service, which is designed to coordinate any communication between the Console and the Agents. All communication between the two is encrypted to ensure secure communication. When the Console is installed a public key and a server key is generated. The public key is used by all agents and is needed to successfully install an agent. The agent to communicate to the console uses this public key. Thus this set of public key as well as server key ensures an encrypted secure communication path between the console and agent. The console also holds a database, which stores related data including Policies, Exceptions, Security events and agents configuration. The agent when installed by default comes up in warning mode. Besides warning mode the other option/mode available for this agent is to be in prevent mode. This is the mode in which it detects any malicious behavior. It then examines this activity to its signature database, thus blocking any questionable traffic. It is recommended to leave the system running in warning mode initially to trend the traffic and create exceptions for the traffic. The exceptions are the traffic pattern that would be allowed when the systems is set to prevent mode.

The console provides centralized management and configuration of agents. Policies can be defined and agents can be grouped for easier handling. Agent's alarms levels can also be customized, along with any exception rules that can be defined for the agent.
The Console also has the capability to define customizable reports. E-mail and Pager notification can also be configured via this console. Also SNMP traps can be sent to any Network Management station, along with an option to spawn any other processes.

Summary:

Entercept can be classified as an Intrusion prevention product for its pre-emptive action. The agents provide layers of protection through signature shielding as well through kernel protection.

References:

Robert Graham FAQ on Intrusion Detection System
http://www.robertgraham.com/pubs/network-intrusion-detection.html

Fyodor
http://www.insecure.org/nmap/OSDEM_Presentation/

Computer Incident Advisory Capability – U.S. Dept. of Energy.
http://ciac.llnl.gov/ciac/ToolsUnixNetSec.html

Psionic Technologies – Abacus Project
http://www.psionic.com/products/index.html

Carnegie Mellon – Software Engineering Institute
http://www.cert.org

Entercept Security Technologies
http://www.entercept.com

Assignment 2 – Network Detects

Detect 1 – RPC Portmap request

```
[**] [1:583:1] RPC portmap request rstatd [**]
[Classification: Attempted Information Leak] [Priority: 3]
02/08-18:52:54.386871 203.85.35.1:692 -> a.b.c.20:111
UDP TTL:47 TOS:0x0 ID:1907 IpLen:20 DgmLen:84
Len: 64
[Xref => http://www.whitehats.com/info/IDS10]

[**] [1:583:1] RPC portmap request rstatd [**]
[Classification: Attempted Information Leak] [Priority: 3]
02/08-18:52:59.284269 203.85.35.1:692 -> a.b.c.20:111
UDP TTL:47 TOS:0x0 ID:2869 IpLen:20 DgmLen:84
Len: 64
[Xref => http://www.whitehats.com/info/IDS10]

[**] [1:583:1] RPC portmap request rstatd [**]
[Classification: Attempted Information Leak] [Priority: 3]
02/08-18:53:04.294567 203.85.35.1:692 -> a.b.c.20:111
UDP TTL:47 TOS:0x0 ID:3779 IpLen:20 DgmLen:84
Len: 64
[Xref => http://www.whitehats.com/info/IDS10]

[**] [1:583:1] RPC portmap request rstatd [**]
[Classification: Attempted Information Leak] [Priority: 3]
02/08-18:53:09.303699 203.85.35.1:692 -> a.b.c.20:111
UDP TTL:47 TOS:0x0 ID:3780 IpLen:20 DgmLen:84
Len: 64
[Xref => http://www.whitehats.com/info/IDS10]

[**] [1:583:1] RPC portmap request rstatd [**]
[Classification: Attempted Information Leak] [Priority: 3]
02/08-18:53:14.312786 203.85.35.1:692 -> a.b.c.20:111
UDP TTL:47 TOS:0x0 ID:3951 IpLen:20 DgmLen:84
Len: 64
[Xref => http://www.whitehats.com/info/IDS10]
```

Here is the format of the above traces

[**] Name of Alert [**]
[Classification given to this alert in the rule definition] [Priority: Priority rank given by rule definition]
date-time Source IP Address : Port <direction> Destination IP Address: Port UDP Protocal Time-to-live Type of Service IP Packet Length Datagram Length
UDP datagram length

1. **Source of Trace:**

The data was gathered from my employers network using Snort IDS sensors

2. **Detect was generated by:**

This detect was generated by Snort, the open source IDS Version 1.8.1-RELEASE (Build 74), The standard rule set version "Snort 1.8.1 Ruleset" per snort.conf file was used and the rpc.rules file triggered this detect. These detects were seen generated at past 18:00 hours where the network load is considered low, as its after business hour. Thus snort IDS is less susceptible to any kind of packet loss.

The filter that triggered this alert is defined below.

alert udp $EXTERNAL_NET any -> $HOME_NET 111 (msg:"RPC portmap request rstatd"; content: "|01 86 A0 00 00|"; reference:arachnids,10;classtype:attempted-recon; sid:583; rev:1;)

Please make a note the above line is wrapped for readability purpose.

Here is a brief description of how the rule is interpreted.
The rule is in two parts. The rule header and rule options enclosed in parentheses

alert	Send to alert file
udp	Protocol
$EXTERNAL_NET	snort variable specifying any external network
any	any port
$HOME_NET	Defined in the snort.conf file
any	any port
msg	message text to identify rule

3. **Probability the source address was spoofed:**

The source address can be spoofed as its UDP based query that does not require establishing a three-way handshake as in case of TCP. The intruder can be sniffing on the network and can capture the response. This was not the case in this instance as the device probed as well as the sensor is placed on a switched environment. Also the switch needs manual intervention to enable a monitoring port to be able to sniff the network. This appears to be a reconnaissance probe; the intruder in this instance may be trying to gather information, and thus would need to see answers to his query coming back to him in order for him to succeed with his reconnaissance. Therefore the probability of the source address being spoofed is very low. Also this appears to be a stimulus and therefore does not fall under the category of third party effect, i.e. responding to a packet with a spoofed address.

4. Description of the attack:

RPC allow program on one computer to execute a program on another computer. RPC bind is a server that converts RPC program numbers to TCP or UDP port numbers. When a RPC program starts it tells RPC bind the program number it is serving. When a client wishes to make a RPC call to a given program number it will first contact the servers machines RCP bind to determine which port the RPC request needs to be sent. RPC Bind can expose several vulnerabilities, from buffer overflows, to remote command execution, to an attacker access, which can eventually leads to root access compromise.

There are several listing on RPC vulnerabilities on http://cve.mitre.org, they are listed below for references
http://cve.mitre.org/cgi-bin/cvename.cgi?name=1999-0018
http://cve.mitre.org/cgi-bin/cvename.cgi?name=1999-0019
http://cve.mitre.org/cgi-bin/cvename.cgi?name=1999-0493
http://cve.mitre.org/cgi-bin/cvename.cgi?name=2000-0666

SANS also list RPC vulnerabilities as one of the top ten most critical Internet Security Threats.
http://www.sans.org/topten.htm#3
Plus there are several advisories on CERT on RPC
www.cert.org/advisories/CA-1999-05.html
http://www.kb.cert.org/vuls/id/34043
http://www.ciac.org/ciac/bulletins/k-069.shtml

5. Attack mechanism:

The RPC Bind service was being targeted in the above detect on UDP port 111. This portmapper or RPC Bind service not only listens on both TCP as well as UDP port 111, but also UDP ports greater than 32770
The file /etc/rpc contain all the RPC network programs/services running on a particular host. Typically when a client needs to connect to any of the RPC program on a given host/server it first query RPC bind service for the RPC programs port number. The client typically would know the service name it wants to connect to but would not know the port number. RPC bind would then respond with the port number and then the client would make a connection to that port number. Thus RPC bind query is a program/service name to a port number translator. In some flavors of UNIX it is also referred to as portmapper.

In our detect above it appears that this is a stimulus packet to query for RPC program/service to see if the RPC bind responses back with the port number for the given service if its running. There are several flaws in RPC, thus making it extremely vulnerable. RPC is widely used for NFS. NFS locking service rpc.lockd uses rpc.statd services for file lock recovery in instances when the system crashes or reboots. There is various documented vulnerability associated with rpc.statd service. This rpc.statd supplies user data to syslog() function as a format string. If this

input is not validated, any malicious user may be able to execute any arbitrary code with whatever privileges rpc.statd process is running with, its usually root.
This detect can be categorized as reconnaissance as we see numerous attempt to query RPC bind port 111

6. Correlations:

I did not find any references of the source IP address 203.85.35.1 in any GCIA practical. Although I found similar reconnaissance correlation of RCP bind probes in Rueben Rubio's practical found below.

```
[**] RPC portmap request rstatd [**]06/27-06:26:26.128684 200.54.185.51:872->
my.net.work.100:111UDP TTL:48 TOS:0x0 ID:25352 IpLen:20 DgmLen:84Len: 64
[**] RPC portmap request rstatd [**]06/27-06:26:26.340134 200.54.185.51:873->
my.net.work.114:111UDP TTL:48 TOS:0x0 ID:25361 IpLen:20 DgmLen:84Len: 64
```

Dsheild.org database show the below IP information for 203.85.35.1

```
IP Address: 203.85.35.1
HostName: pc001.butterfield.com.hk
DShield Profile: Country:
Contact E-mail:
Total Records against IP:  106
Number of targets:  76
Date Range: 2002-02-06 to 2002-02-25
Ports Attacked (up to 10): Port Attacks
```

Whois:

```
% Rights restricted by copyright. See http://www.apnic.net/db/dbcopyright.html
% (whois7.apnic.net)

inetnum:     203.85.35.0 - 203.85.35.255
netname:     WAYBO-HK
descr:       Waybo Co Ltd
descr:       Hong Kong
country:     HK
admin-c:     PN29-AP
tech-c:      PN29-AP
mnt-by:      MAINT-HK-PSINET
changed:     hostinfo@psinet.com.hk 20010613
source:      APNIC

person:      PSINet HK NOC
address:     PSINet Hong Kong Ltd.
address:     20/F, Lincoln House,
address:     Taikoo Place,
address:     Quarry Bay,
address:     HONG KONG
```

```
country:    HK
phone:      +852-2331-8123
fax-no:     +852-2372-0287
e-mail:     hostinfo@psinet.com.hk
nic-hdl:    PN29-AP
mnt-by:     MAINT-HK-PSINET
changed:    hostinfo@psinet.com.hk 20010710
source:     APNIC
```

RPC bind is considered to be one of the top ten targeted ports probed. See below references.
http://www.dshield.org/ports/port111.html
http://www.cert.org/advisories/CA-2000-17.html
http://www.dshield.org/topports.html

7. Evidence of active targeting:

This RPC bind query activity was being seen as what appears to be for the entire network. This general scan was seen for the whole 25-bit subnet that we own. If the intruder were seen targeting one specific host, this would be an indication that the intruder already has the reconnaissance information about the given host. As the scan appeared for the entire segment, it rules out the possibility of active targeting. Also I did not see this same source IP address appear again in our alert log.

8. Severity:

(Criticality + Lethality) – (System countermeasures + Network countermeasures) = Severity
The severity of the attack is (4+4) – (5+5) = -2
Criticality = 4 - We have several e-commerce web servers deployed on this Internet segment and are critical to the business.
Lethality = 4 - If portmapper would have been running and the intruder would have been successful in his reconnaissance, this could lead to other attacks.
System countermeasures = 5 - None of our servers in that segment are running portmapper service.
Network countermeasures = 5 – Based on the firewall rules the packet was dropped.

9. Defensive Recommendation:

Continue monitoring IDS and Firewall logs. Also if portmapper is required in the future make use of SecureRPC/portmapper instead. And if portmapper is not required block TCP & UDP port 111 as well as 32770-32772 on the border router. Firewalls rules are needed to be hardened to restrict access. Make sure all systems are deployed with their latest security patches. And also look into host based IDS systems for that host.

10. Multiple choice test question:

Which of the following port is used for rpcinfo query to determine the active RPC prog Ids and their port numbers

A) port 113
B) port 111

C) port 119
D) port 137 ……………The answer is B

Detect 2 – SYN/FIN Scan

```
[**] [111:13:1] spp_stream4: STEALTH ACTIVITY (SYN FIN scan) detection [**]
02/11-00:25:23.683237 207.200.55.150:21 -> a.b.c.43:21
TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
******SF Seq: 0x2C1C682B Ack: 0x309D0FD2 Win: 0x404 TcpLen: 20

[**] [111:13:1] spp_stream4: STEALTH ACTIVITY (SYN FIN scan) detection [**]
02/11-00:25:24.003361 207.200.55.150:21 -> a.b.c.59:21
TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
******SF Seq: 0x2C1C682B Ack: 0x309D0FD2 Win: 0x404 TcpLen: 20

[**] [111:13:1] spp_stream4: STEALTH ACTIVITY (SYN FIN scan) detection [**]
02/11-00:25:24.023788 207.200.55.150:21 -> a.b.c.60:21
TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
******SF Seq: 0x2C1C682B Ack: 0x309D0FD2 Win: 0x404 TcpLen: 20

[**] [111:13:1] spp_stream4: STEALTH ACTIVITY (SYN FIN scan) detection [**]
02/11-00:25:24.123353 207.200.55.150:21 -> a.b.c.65:21
TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
******SF Seq: 0x2C1C682B Ack: 0x309D0FD2 Win: 0x404 TcpLen: 20

[**] [111:13:1] spp_stream4: STEALTH ACTIVITY (SYN FIN scan) detection [**]
02/11-00:25:24.243343 207.200.55.150:21 -> a.b.c.71:21
TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
******SF Seq: 0x2C1C682B Ack: 0x309D0FD2 Win: 0x404 TcpLen: 20
```

Here is the format of the above traces

```
[**]  Name of Alert [**]
[Classification given to this alert in the rule definition] [Priority: Priority rank given by rule
definition]
date-time Source IP Address : Port <direction> Destination IP Address: Port
TCP Protocol  Time-to-Live  Type of Service   IP Packet Length  Datagram Length
TCP Flags   Sequence Number     Acknowledge Number    Window size    TCP datagram length
```

1. **Source of Trace:**

The data was captured at my employer's network using Snort IDS

2. **Detect was generated by:**

This detect was generated by Snort, the open source IDS Version 1.8.1-RELEASE (Build 74), The standard rule set version "Snort 1.8.1 Ruleset" per snort.conf file was used and the scan.rules file triggered this detect. These detects were seen generated after midnight where the network load was low, as its after business hour. Once again as the previous detect, Snort is less susceptible to packet loss as the traffic is low.

The filter that triggered this alert is defined below.

alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SCAN SYN FIN";flags:SF;)

Please make a note the above line is wrapped for readability purpose.

Here is a brief description of how the rule is interpreted.
The rule is in two parts. The rule header and rule options enclosed in parentheses

alert	Send to alert file
tcp	Protocol
$EXTERNAL_NET	snort variable specifying any external network
any	any port
$HOME_NET	Defined in the snort.conf file
any	any port
msg	message text to identify rule
flags	TCP flag bits

3. **Probability the source address was spoofed:**

There is a possibility that the source address could be spoofed, as it appears that the packets have been crafted. It appears that Synscan tool was used to craft the packet. This tool is widely used for scanning and probing host. This tool has its distinguished characteristics of IP identification number of 39426 and TCP window size of 1028, which appears in above detect. The TTL value also remains constant at 29, no similar packet traverses and takes the same on Internet and reaches the destination with the same TTL values. This is another sign of this packet being crafted. Thus the packet could probably be crafted using any spoofed IP address. This attempt was made using TCP, and for all purpose the intruder would want to receive answers to his query returned back to him for a successful reconnaissance. Thus the probability of source IP address spoofed is extremely low. This also rules out the category of third party effect, as it does not appear to be a response to a packet with a spoofed address.

4. **Description of attack:**

As we see this was a SYN/FIN scan for an ftp server. SYN is a tcp flag to initiate a three-way hand shake. FIN is a tcp flag to request a graceful shutdown of an established tcp session. Thus SYN and FIN flag combined together is not a normal occurrence. This confirms that these packets were crafted. You will also notice in the trace an acknowledgement number set with a non-zero value without setting the ACK bit, along with reflexive port that is set to port 21. Normal request to ftp server is usually seen on source ports 1024 or higher to the ftp destination port 21. Here is some more indication of these packets being crafted.

5. **Attack mechanism:**

As mentioned earlier this is not a normal occurrence to have both SYN/FIN bit set, it indicates an intentional probe. It appears to be stimulus to probe any active ftp servers. Thus ftp service is being targeted here. Normal TCP/IP stack would respond with either SYN/ACK for an open port or RST/ACK for a closed one. Filtered ports would just drop the packet and may reply with icmp message. This is a technique to subvert an improperly configured packet-filtering device, i.e. only initial SYN may be blocked thus leaving it open for a packet with any other flag bit set along with SYN. In case of a stateful firewall, it will drop this type of packet. This SYN/FIN technique is mostly utilized for reconnaissance. Once the intruder is successful with his reconnaissance what follows next would be any exploits for any vulnerable ftp servers.

6. Correlation:

Since this is an older protocol, before http gain popularity, after http, ftp is the second most probed port out there. http://www.dshield.org/ports/port21.html
http://www.dshield.org/topports.html
There are references of several documented issues with ftp, see..
http://www.cert.org/tech_tips/ftp_port_attacks.html
According to CERT, there are several ftp advisories on its vulnerabilities.
http://www.cert.org/advisories/CA-1993-10.html

I could not find any direct correlation for the same source IP address in any GCIA practical, but Dshield.org shows the below information for this source IP number.

IP Address: 207.200.55.150
HostName: 207.200.55.150
DShield Profile: Country: US
Contact E-mail: chad@ONR.COM
Total Records against IP: 6769
Number of targets: 4452
Date Range: 2002-02-10 to 2002-02-10
Ports Attacked (up to 10): Port Attacks

Whois: Onramp Access, Inc. (NETBLK-ONR-CIDR)
  612 Brazos, Suite 103
  Austin, TX 78701
  US

  Netname: ONR-CIDR
  Netblock: 207.200.0.0 - 207.200.63.255
  Maintainer: ONR

  Coordinator:
    Kissinger, Chad (CK47-ARIN) chad@ONR.COM
    512-322-9200 (FAX) 512-322-9200

  Domain System inverse mapping provided by:

```
   SIERRA.ONR.COM                  199.1.90.2
   FIVER.ONR.COM           199.1.90.162
   SYLVESTER.ONR.COM               199.1.90.157

   ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

   Record last updated on 26-Sep-2001.
   Database last updated on  2-Mar-2002 19:57:03 EDT.

----------

Steve Axelrod (NETBLK-ONR-445)
   612 Brazos
   Austin, TX 78701
   US

   Netname: ONR-445
   Netblock: 207.200.55.0 - 207.200.55.255

   Coordinator:
      Kissinger, Chad  (CK47-ARIN)  chad@ONR.COM
      512-322-9200 (FAX) 512-322-9200

   Domain System inverse mapping provided by:

   SIERRA.ONR.COM                  199.1.90.2
   FIVER.ONR.COM           199.1.90.162

   Record last updated on 15-Oct-1999.
   Database last updated on  2-Mar-2002 19:57:03 EDT.

The ARIN Registration Services Host contains ONLY Internet
Network Information: Networks, ASN's, and related POC's.
Please use the whois server at rs.internic.net for DOMAIN related
Information and whois.nic.mil for NIPRNET Information.
```

7. Evidence of active targeting:

```
207.200.55.150:21 -> a.b.c.4:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.20:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.31:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.32:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.41:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.43:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.59:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.60:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.65:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
```

```
207.200.55.150:21 -> a.b.c.71:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.74:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.82:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.85:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.86:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.94:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.101:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.110:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.111:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
207.200.55.150:21 -> a.b.c.121:21 TCP TTL:29 TOS:0x0 ID:39426 IpLen:20 DgmLen:40
```

As you can see from the log above, there were several attempts made to probe this ftp port. This does not appear to be targeting a specific host. This appears to be more like a general probe to random IP numbers. This can be interpreted as a hostile approach to look for open ftp servers. I don't consider it a wrong number since the packet, which is destined to our perimeter, was probably addressed to us. But I don't see any evidence of active targeting as these scans stopped coming to our perimeter network after this instance.

8. Severity:

(Criticality + Lethality) – (System countermeasures + Network countermeasures) = Severity
The severity of the attack is (4+1) – (3+5) = -3
Criticality = 4 - We have a few ftp servers deployed that are critical to our business.
Lethality = 1 - Although there were several probes, this was strictly a probes to look for any actively available ftp servers. If these servers had compromised it would have been a nightmare to re-build them. Also the accountability of business loss would be tremendous.
System countermeasures = 3 - The ftp servers are not up to date with their latest security patches.
Network countermeasures = 5 – Our deployed firewall is stateful, thus it will not allow any packet with both SYN/FIN bit flagged.

9. Defensive recommendation:

First of all make sure all the ftp servers are up to date with their latest security patches. Allow anonymous only if it's necessary with passive ftp. Also if anonymous ftp is allowed make sure write access is restricted. Check the Statefulness of the firewall; verify with the logs to make sure packets with SYN/FIN flagged set are being dropped.

10. Multiple choice test question:

What initial flag bit must set for TCP three-way handshake?

A) SYN
B) RST
C) FIN
D) ACK
E) SYN/FIN

Answer is A

Detect 3 – WEB-IIS ISAPI Vulnerability

```
[**] [1:1243:1] WEB-IIS ISAPI .ida attempt [**]
[Classification: Attempted Administrator Privilege Gain] [Priority: 10]
02/12-11:50:15.259701 199.18.64.37:4413 -> a.b.c.67:80
TCP TTL:112 TOS:0x0 ID:38994 IpLen:20 DgmLen:576 DF
***AP*** Seq: 0x6C31E58E Ack: 0x82358A54 Win: 0xB68 TcpLen: 20

[**] [1:1243:1] WEB-IIS ISAPI .ida attempt [**]
[Classification: Attempted Administrator Privilege Gain] [Priority: 10]
02/12-11:50:21.264252 199.18.64.37:4413 -> a.b.c.67:80
TCP TTL:112 TOS:0x0 ID:39150 IpLen:20 DgmLen:576 DF
***AP*** Seq: 0x6C31E58E Ack: 0x82358A54 Win: 0xB68 TcpLen: 20

[**] [1:1243:1] WEB-IIS ISAPI .ida attempt [**]
[Classification: Attempted Administrator Privilege Gain] [Priority: 10]
02/12-11:50:33.281500 199.18.64.37:4413 -> a.b.c.67:80
TCP TTL:112 TOS:0x0 ID:39491 IpLen:20 DgmLen:576 DF
***AP*** Seq: 0x6C31E58E Ack: 0x82358A54 Win: 0xB68 TcpLen: 20

[**] [1:1243:1] WEB-IIS ISAPI .ida attempt [**]
[Classification: Attempted Administrator Privilege Gain] [Priority: 10]
02/12-11:50:57.313174 199.18.64.37:4413 -> a.b.c.67:80
TCP TTL:112 TOS:0x0 ID:40190 IpLen:20 DgmLen:576 DF
***AP*** Seq: 0x6C31E58E Ack: 0x82358A54 Win: 0xB68 TcpLen: 20
```

Here is the format of the above traces

```
[**]  Name of Alert [**]
[Classification given to this alert in the rule definition] [Priority: Priority rank given by rule definition]
date-time Source IP Address : Port <direction> Destination IP Address: Port
TCP Protocol  Time-to-Live Type of Service   IP Packet Length    Datagram Length
TCP Flags   Sequence Number     Acknowledge Number     Window size    TCP datagram length
```

1. **Source of Trace:**

The data was captured at my employer's network using Snort IDS

2. **Detect was generated by:**

This detect was generated by Snort, the open source IDS Version 1.8.1-RELEASE (Build 74), The standard rule set version "Snort 1.8.1 Ruleset" per snort.conf file was used and the web-iis.rules file triggered this detect. The second complementing trace was detected from TCPDUMP running on LaBrea. LaBrea as you know is a Linux based tool that can make all unused IP addresses on the network to use, by creating a “tarpit” that can slow down or even stop scans destined to your address block.

The filter that triggered this alert is defined below.

alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB-IIS ISAPI .ida access"; uricontent:".ida"; nocase; flags:A+; reference:arachnids,552; classtype:web-application-activity; reference:cve,CAN-2000-0071; sid:1242; rev:2;)

Please make a note the above line is wrapped for readability purpose.

Here is a brief description of how the rule is interpreted.
The rule is in two parts. The rule header and rule options enclosed in parentheses

alert	Send to alert file
tcp	Protocol
$EXTERNAL_NET	snort variable specifying any external network
any	any port
$HOME_NET	Defined in the snort.conf file
any	any port
msg	message text to identify rule
flags	TCP flag bits

3. **Probability the source address was spoofed:**

The probability of this source address was spoofed is low, because we see a TCP three-way handshake between the intruder’s machine and the victim host. It was determined that this address was from a network address block assigned to an ISP in Columbus, OH

```
OARnet (NETBLK-OARNET-CBLK2)
  2455 North Star Road
  Columbus, OH 43221
  US

  Netname: NETBLK-OARNET-CBLK2
  Netblock: 199.18.0.0 - 199.18.255.255
  Maintainer: OAR

  Coordinator:
    Steele, Greg  (GS1050-ARIN)  hostmaster@OAR.NET
    +1-800-627-6420 ext. 203 (FAX) +1-614-728-8110

  Domain System inverse mapping provided by:

  NS1.OAR.NET                    192.88.193.144
  NS2.OAR.NET                    192.88.195.10

  *Rwhois information on assignments from this block available from
```

```
* rwhois.oar.net 4321

ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

Record last updated on 27-Jul-1999.
Database last updated on  12-Feb-2002 19:56:22 EDT.
```

4. Description of attack:

Vulnerability exists in Microsoft's Indexing Services (IS) that is used by the Internet Information Server (IIS) v4.0 and v5.0 running on Windows NT, 2000 as well as beta version of Windows XP. This vulnerability is a remotely exploitable buffer overflow in IS Application Programming Interface (API) extension (IDQ.DLL). This vulnerability generally allows an intruder to execute any arbitrary code on the victim host.

The following is the description of this vulnerability as reported by Common Vulnerabilities & Exposures database,

"Microsoft Index Server allows remote attackers to determine the real path for a web directory via a request to an Internet Data Query file that does not exist."
http://cve.mitre.org/cgi-bin/cvename.cgi?name=2000-0071

Microsoft security bulletin describe the same vulnerability at the below url
http://www.microsoft.com/technet/treeview/default.asp?url=/TechNet/security/bulletin/MS01-033.asp
Here is another url from eEye Digital Security, which describe this vulnerability in detail. According to CERT, eEye Digital Security actually discovered this original vulnerability.
http://www.eeye.com/html/Research/Advisories/AD20010618.html

5. Attack Mechanism:

As mentioned in the description this vulnerability is an exploitable buffer overflow in ISAPI installed with Microsoft's IIS v4.0 and v5.0. This detect is past its stage of being an initial stimulus, this detect appears to be a TCP conversation with ACK & PUSH flag set, i.e. beyond the TCP three-way handshake. This detect is targeting Web-IIS service, the Index Service API to run arbitrary code, which is run within the local system security context. This service as mentioned is extremely vulnerable and once exploited, it can give the potential intruder complete control of the victim host.
There have been other Cert Advisories (see below), which reported malicious code that exploits, this known vulnerability. This was categorized as a self-replicating worm, identified as "Code Red"

CERT® Advisory CA-2001-19
"Code Red" Worm Exploiting Buffer Overflow In IIS Indexing Service DLL
<snip>
The "Code Red" worm is self-replicating malicious code that exploits a known vulnerability in Microsoft IIS servers (CA-2001-13).
Attack Cycle
The "Code Red" worm attack proceeds as follows:

1. The "Code Red" worm attempts to connect to TCP port 80 on a randomly chosen host assuming that a web server will be found. Upon a successful connection to port 80, the attacking host sends a crafted HTTP GET request to the victim, attempting to exploit a buffer overflow in the Indexing Service described in CERT advisory CA-2001-13 <snip>

6. **Correlations:**

I did not see any correlation of the same source IP address, although this vulnerability is being documented in many GCIA practical. See below a correlation from Tom Jones's practical

```
195.53.245.250 - - [19/Jul/2001:11:15:57 -0500] "GET
/default.ida?NNNNNNNN...NNNNNNNNNNNNNNNNNNNNNNNN%u9090%u6858%ucbd3%
u7801%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u9090%u819
0%u00c3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a
HTTP/1.0" 404 207
209.167.130.22 - - [19/Jul/2001:11:46:13 -0500] "GET
/default.ida?NNNNNNNN...NNNNNNNNNNNNNNNNNNNNNNNN%u9090%u6858%ucbd3%
u7801%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u9090%u819
0%u00c3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a
HTTP/1.0" 404 207
```

Dshields.org has the below information on the source IP address.

IP Address: 199.18.64.37
HostName: 199.18.64.37
DShield Profile: Country: US
Contact E-mail: steele@OAR.NET
Total Records against IP: 176
Number of targets: 66
Date Range: 2002-02-12 to 2002-02-14
Ports Attacked (up to 10): Port Attacks

There have also been several advisories on CERT web site.
http://www.cert.org/advisories/CA-2001-13.html
http://www.cert.org/advisories/CA-2001-19.html

7. **Evidence of the active attack:**

This appears to be a general scan of our whole entire network. Thus was not targeted to any specific host. We see a lot of re-tries in our logs with the same sequence number as well the ack number. With the same string of GET/default.ida (see below trace) . This does not also appears as a wrong number as it was addressed to host residing on the Internet perimeter. Since this is a malicious code/worm type of activity, I cannot classify this ongoing traffic as evidence of active attack.

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

02/07-22:03:53.171795 0:B0:64:B4:D0:60 -> 0:0:F:FF:FF:FF type:0x800 len:0x24E
156.63.63.109:1320 -> a.b.c.118:80 TCP TTL:110 TOS:0x0 ID:21828 IpLen:20 DgmLen:576
DF
***AP*** Seq: 0xE7407507  Ack: 0xD33B9A1F  Win: 0xB68  TcpLen: 20
47 45 54 20 2F 64 65 66 61 75 6C 74 2E 69 64 61  GET /default.ida
3F 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  ?NNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
4E 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  N...............
C3 03 00 00 00 78 00 FA 20 25 75 39 30 39 30 25  .....x.. %u9090%
75 36 38 35 38 25 75 63 62 64 33 25 75 37 38 30  u6858%ucbd3%u780
31 25 75 39 30 39 30 25 75 36 38 35 38 25 75 63  1%u9090%u6858%uc
62 64 33 25 75 37 38 30 31 25 75 39 30 39 30 25  bd3%u7801%u9090%
75 39 30 39 30 25 75 38 31 39 30 25 75 30 30 63  u9090%u8190%u00c
33 25 75 30 30 30 33 25 75 38 62 30 30 25 75 35  3%u0003%u8b00%u5
33 31 62 25 75 35 33 66 66 25 75 30 30 37 38 25  31b%u53ff%u0078%
75 30 30 30 30 25 75 30 30 3D 61 20 20 48 54 54  u0000%u00=a  HTT
50 2F 31 2E 30 0D 0A 43 6F 6E 74 65 6E 74 2D 74  P/1.0..Content-t
79 70 65 3A 20 74 65 78 74 2F 78 6D 6C 0A 48 4F  ype: text/xml.HO
53 54 3A 77 77 77 2E 77 6F 72 6D 2E 63 6F 6D 0A  ST:www.worm.com.
20 41 63 63 65 70 74 3A 20 2A 2F 2A 0A 43 6F 6E   Accept: */*.Con
74 65 6E 74 2D 6C 65 6E 67 74 68 3A 20 33 35 36  tent-length: 356
39 20 0D 0A 0D 0A 55 8B EC 81 EC 18 02 00 00 53  9 ....U........S
56 57 8D BD E8 FD FF FF B9 86 00 00 00 B8 CC CC  VW..............
CC CC F3 AB C7 85 70 FE FF FF 00 00 00 00 E9 0A  ......p.........
0B 00 00 8F 85 68 FE FF FF 8D BD F0 FE FF FF 64  .....h.........d
A1 00 00 00 00 89 47 08                          ......G.

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

8. Severity:

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

The severity of the attack is (5+5) – (5+5) = 0
Criticality = 5 - We have quite a few web servers deployed that are critical to the business.
Lethality = 5 - If one of our systems had been compromised, this code would have self-replicated to other systems. This is a very lethal code that can cause Denial of service by tying up the network bandwidth. This actually happened and we saw performance degradation on our systems. Although you see several re-tries, it's almost certain that one of our systems got compromised.
System countermeasures = 1 – Most of our systems did not have the latest security patches. Thus were infected. The above trace, you'll see has a fictitious mac address of 0:0:F:FF:FF:FF This is our Honey pot system (LaBrea) responding to this request. LaBrea was used to respond with low windows size to tie down any TCP session destined to our perimeter.
Network countermeasures = 5 – Firewall in this place would allow access on port 80 based on the defined rule. But since this was targeting port 80 it compromised all vulnerable host which were deployed without any security patch.

9. Defensive recommendation:

Make sure all of the servers have all the possible latest security patches deployed. Keep current on information regarding security issues with IIS. There are numerous vulnerabilities found associated with IIS web servers. If you're running Cisco routers deploy Network Based Applications Recognition (NBAR) access list on border router, which looks at the above content string and block access.

10. Multiple choice test question:

Given the content of snort log, what can you say about this GET/default.ida request??

A) Is this an ftp get request
B) Is this IIS vulnerability
C) Is this a TFTP get request

Answer is B

Detect 4 – Attempted Proxy server connection from Internet

```
[**] [1:615:1] SCAN Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 3]
02/11-12:19:36.592665 66.28.178.15:38609 -> a.b.c.4:1080
TCP TTL:48 TOS:0x0 ID:16347 IpLen:20 DgmLen:60 DF
******S* Seq: 0x9BDAFA64 Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 19449279 0 NOP WS: 0

[**] [1:615:1] SCAN Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 3]
02/12-02:09:34.843673 66.28.178.15:52198 -> a.b.c.12:1080
TCP TTL:48 TOS:0x0 ID:10580 IpLen:20 DgmLen:60 DF
******S* Seq: 0xDA5EEBA2 Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 24429302 0 NOP WS: 0

[**] [1:615:1] SCAN Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 3]
02/12-04:30:27.368698 66.28.178.15:39596 -> a.b.c.13:1080
TCP TTL:48 TOS:0x0 ID:15154 IpLen:20 DgmLen:60 DF
******S* Seq: 0xEE1793EE Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 25274588 0 NOP WS: 0

[**] [1:615:1] SCAN Proxy attempt [**]
[Classification: Attempted Information Leak] [Priority: 3]
02/12-05:57:29.721286 66.28.178.15:56036 -> a.b.c.14:1080
TCP TTL:48 TOS:0x0 ID:47221 IpLen:20 DgmLen:60 DF
******S* Seq: 0x36951899 Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 25796844 0 NOP WS: 0
```

Here is the format of the above traces

```
[**]  Name of Alert [**]
[Classification given to this alert in the rule definition] [Priority: Priority rank given by rule
definition]
date-time Source IP Address : Port <direction> Destination IP Address: Port
TCP Protocol  Time-to-Live Type of Service   IP Packet Length    Datagram Length
TCP Flags   Sequence Number      Acknowledge Number     Window size    TCP datagram length
TCP options.
```

1. **Source of Trace:**

The data was captured at my employer's network using Snort IDS

2. **Detect was generated by:**

This detect was generated by Snort, the open source IDS Version 1.8.1-RELEASE (Build 74), The standard rule set version "Snort 1.8.1 Ruleset" per snort.conf file was used and the web-scans.rules file triggered this detect.

The filter that triggered this alert is defined below.

alert tcp $EXTERNAL_NET any -> $HOME_NET 1080 (msg:"SCAN Proxy attempt"; flags:S; reference:url,help.undernet.org/proxyscan/; classtype:attempted-recon; sid:615; rev:2;)

Please make a note the above line is wrapped for readability purpose.

Here is a brief description of how the rule is interpreted.
The rule is in two parts. The rule header and rule options enclosed in parentheses

alert	Send to alert file
tcp	Protocol
$EXTERNAL_NET	snort variable specifying any external network
any	any port
$HOME_NET	Defined in the snort.conf file
any	any port
msg	message text to identify rule
flags	TCP flag bits

3. **Probability the source address was spoofed:**

First let's look at the possibility that the address was spoofed. If you look at the packet it has the IP header length of 20 bytes, the TCP header length is 40 thus making the datagram 60 bytes. The source port changes with each attempt, also the IP Identification number changes at each attempt. Thus what appears is a normal uncrafted packet. Therefore the probability of this being spoofed is very low. Also it is a TCP based communication probing different hosts in our address space with different sequence number. The intruder is trying to probe for a vulnerable proxy server, and I am sure he would like to get a response back from any proxy server listening on that port. It does not even fall into the category of third party effect, as it is not a response to any spoofed address. Therefore, it's a stimulus packet with an attempt of reconnaissance.

4. **Description of attack:**

This user is trying to scans our address space to find a SOCKS proxy sever listening on port 1080. This I would not consider malicious in nature as this user is strictly trying to hide behind a proxy server to surf the Internet anonymously and conceal his/hers location. These probes are now getting very common as many home/Small offices have the default installation of Wingate/SOCKS, which is vulnerable. As mentioned earlier the attacker is just interested in this service so they can bounce of their connection through the vulnerable victim, thus concealing their location/identity.

A search on CVE listed the below vulnerability on Wingate.

http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0291

Cert Vulnerability notes below talk about the default installation of Wingate and how an intruder or potential attacker can conceal his or hers own true location without any need to forge packet.
http://www.cert.org/vul_notes/VN-98.03.WinGate.html

5. **Attack mechanism:**

This appears to be a stimulus probing for port 1080 for Wingate/SOCKS service. As you can see that Wingate/SOCKS proxy service is being targeted. 1080 is a typical port for SOCKS proxy service. Wingate http://www.wingate.net/ is a popular proxy/firewall on Windows platform that listens on the same port by default. As mentioned in description section, this service is known to have vulnerability associated with SOCKS proxy service. A SOCKS service allows a client to access the Internet through SOCKS server. The client then assumes this servers IP address. Attacker can use these vulnerable SOCKS server as proxies to conceal their source address on the Internet. If this Wingate installation is not secured with restrictive access, it can accept connection from anywhere. Attacker can thus hide their identity behind it, if they can get their hands on any such vulnerable proxy server. Therefore this appears as a reconnaissance activity to scan for any vulnerable proxy server.

6. **Correlations:**

"The WinGate proxy is installed without a password, which allows remote attackers to redirect connections without authentication."

There were several references found on www.sans.org on probes on wingate port 1080
See below.

```
11/19-00:07:39.344116  [**] WinGate 1080 Attempt [**] 168.120.16.250:47988 ->
MY.NET.97.187:1080
11/19-00:37:47.119013  [**] WinGate 1080 Attempt [**] 168.120.16.250:49644 ->
MY.NET.97.187:1080
11/19-01:12:27.363966  [**] WinGate 1080 Attempt [**] 168.120.16.155:40984 ->
MY.NET.97.187:1080
11/19-01:12:30.724785  [**] WinGate 1080 Attempt [**] 168.120.16.155:40984 ->
MY.NET.97.187:1080
11/19-01:36:04.580979  [**] spp_portscan: PORTSCAN DETECTED from 35.10.22.101
(STEALTH) [**]
11/19-01:36:05.671231  [**] spp_portscan: portscan status from 35.10.22.101: 1 connections
across 1 hosts: TCP(1), UDP(0) STEALTH [**]
11/19-01:36:07.816344  [**] spp_portscan: End of portscan from 35.10.22.101 (TOTAL
HOSTS:1 TCP:1 UDP:0) [**]
11/19-01:29:03.079870  [**] WinGate 1080 Attempt [**] 208.194.160.1:3962 ->
MY.NET.98.93:1080
11/19-01:32:23.410308  [**] WinGate 1080 Attempt [**] 168.120.16.155:41508 ->
MY.NET.97.187:1080
```

```
11/19-01:39:10.965297  [**] WinGate 1080 Attempt [**] 168.120.16.155:41728 ->
MY.NET.97.187:1080
11/19-01:54:17.594608  [**] WinGate 1080 Attempt [**] 168.120.16.155:42222 ->
MY.NET.97.187:1080
11/19-02:00:25.205816  [**] WinGate 1080 Attempt [**] 216.234.161.197:4098 ->
MY.NET.221.198:1080
11/19-02:17:06.299074  [**] WinGate 1080 Attempt [**] 168.120.16.155:42748 ->
MY.NET.97.187:1080
```

7. Evidence of active targeting:

We saw several scans coming to our network. This probe was a general scan for the entire address space that we own. This was not targeted to any specific host. This is a pretty common scans that is seen all the time. This does not even appear like a probable wrong number, i.e. transpose of IP numbers. I don't see evidence of active targeting as the scans, which I have seen, have been originating from different sources.

8. Severity:

(Criticality + Lethality) – (System countermeasures + Network countermeasures) = Severity
The severity of the attack is (5+3) – (5+5) = -2
Criticality = 5 – We have a Web http/ftp caching host, residing directly on the Internet segment. This device is setup as a proxy server too.
Lethality = 3 - The attack is not lethal in nature as the attacker is strictly trying to hide his identity by going via this proxy server. It just may degrade our http performance.
System countermeasures = 5 – The system had an access list that would restrict access to the box. This had also been kept up to date with its security patches, etc.
Network countermeasures = 5 – There was an access list on the border router which would inspect the source IP before it would let the traffic go to the caching/proxy server.

9. Defensive recommendation:

To begin with verify the restriction deployed on the caching/proxy server. Secondly configure the proxy server to listen on a custom port instead of the default port. Deploy & verify the access list on the router that inspects the source IP address. Monitor the access-list logs to make sure you recognize the source addresses. Make sure you routinely deploy all the security patches and code updates as and when they become available.

10. Multiple choice test question:

What service is port TCP 1080 associated with?

A) Telnet service
B) SOCKS proxy service
C) FTP service
D) TFTP service

The answer is B

Detect – 5 – Scan for SubSeven v2 Trojan

ZoneAlarm log:

```
FWIN,2001/11/16,14:08:13 -8:00 GMT,24.202.209.193:3216,x.y.z.192:27374,TCP (flags:S)
```

Netgear DSL/Cable router log:

```
12/28/2001 11:52:41.464 - Sub Seven Attack Dropped - Source:24.177.160.206, 4180, WAN
- Destination:a.b.c.148, 1243, LAN - -

12/22/2001 01:56:41.208 - Sub Seven Attack Dropped - Source:24.11.147.2, 1867, WAN -
Destination:a.b.c.148, 1243, LAN - -

12/21/2001 18:37:59.544 - Sub Seven Attack Dropped - Source:24.9.194.190, 4188, WAN -
Destination:a.b.c.148, 1243, LAN - -
```

1. **Source of Trace:**

This data was gathered from my home network, the first log is from my ZoneAlarm personal firewall and the next log is from my Netgear DSL/Cable router, which I deployed eventually later on replacing ZoneAlarm. I found the above traces in my logs.

2. **Detect was generated by:**

Rules defined in ZoneAlarm to drop any traffic to port 27374, as well as Netgear to drop any traffic to port 1243

3. **Probability the source address was spoofed:**

Let's first look at the possibility of the address being spoofed. Packet can be crafted with the SYN flag set with spoofed address destined for the above given destination IP number/port. The intruder can then be sniffing the cable network to see the return traffic. Thus there is a possibility of the address being spoofed. These logs (Zone Alarm & Netgear) provide limited information such as there is no sequence number or Acknowledgement number, no IP identification number, or IP, TCP or Datagram length for the packet in the log. Thus making it difficult to judge. The fact that this traffic appears to be trolling for Trojans such as this SubSeven, I'll go with the notion that the address is not spoofed. The intruder may be using some sort of a tool to scan a large address space to find systems, which are already compromised. Therefore I believe the intruder would like seeing responses coming back to him to assist in tracking a list of all compromised host that the intruder can then try and exploit. This appears to be a stimulus packet waiting for a response, thus rules out the possibility of third party effect, i.e. responses generated by forged packet that has spoofed IP addresses.

4. **Description of attack:**

The source host is trolling for Trojans, i.e. SubSeven v2 compromised hosts. SubSeven is categorized as a backdoor and remote administration program similar to Back Orifice and Netbus. Once installed this Trojan program can allows other to access as well as control the compromised system. SubSeven allows a remote attacker to obtain cached password, capture screen shots, etc. Most of the systems that are infected to this date have been home users with Cable/DSL connections to the Internet. Subseven systems can also be controlled via IRC commands and its possible to perform Distributed Denial of Service attack using these SubSeven agents. All this is done with no knowledge of the systems owner.

There are numerous tech tips for Home users on CERT home page, see below.
http://www.cert.org/tech_tips/home_networks.html#III-B-1

A detail description of this Trojan can be found below;
http://www.europe.f-secure.com/v-descs/subseven.shtml
http://www.symantec.com/avcenter/venc/data/backdoor.subseven.html

Some advisories on SubSeven can be found on iss.net site, see below;
http://www.iss.net/security_center/alerts/advise73.php
http://www.iss.net/security_center/alerts/advise65.php

5. Attack mechanism:
It is a Trojan code and can be installed via infected software, e-mail, etc. This appears to be a scan for this Trojan, thus making it a stimulus. The probes were to troll for this Trojan and thus if any compromised host found obviously an exploit would then follow. As mentioned in the description section; SubSeven is a back door/remote administration program. This program is often used as a Trojan horse, which enables an intruder to execute any custom arbitrary command on the victim machine. SubSeven v2.0 is usually controlled via an IRC server. It is found listening on port 27374. A variant of SubSeven BackdoorG can also be found running on port 1243. A pretty popular activity on the Internet these days is to hijack an infected SubSeven-v2.0 system. Therefore this detect can be categorized as a reconnaissance activity. If this were a compromised SubSeven system and the default password has not been changed, then the intruder who is probing can have full access to the system, without the knowledge of this system owner.

6. Correlations:
I did not find any correlation for the source IP address in GCIA practical, although I found few traces documenting similar probes on SANS web site. See trace below.

Dshield.org show the following information for the source IP address;

IP Address: 24.202.209.193
HostName: modemcable193.209-202-24.mtl.mc.videotron.ca
DShield Profile: Country: CA
Contact E-mail: abuse@videotron.ca (bounced)
Total Records against IP:
Number of targets:
Date Range: to
Ports Attacked (up to 10): Port Attacks

Whois: Le Groupe Videotron Ltee (NETBLK-VL-2BL)

```
  2000 Rue Berri
  Montreal, Quebec H1V 2E4
  CA

  Netname: VL-2BL
  Netblock: 24.200.0.0 - 24.203.255.255
  Maintainer: VLCA

  Coordinator:
     Roy, Pierre  (PR163-ARIN)  abuse@videotron.ca
     514 281-0850

  Domain System inverse mapping provided by:

  DNS1.VIDEOTRON.NET          205.151.222.250
  DNS2.VIDEOTRON.NET          205.151.222.251

  ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

  Record last updated on 26-Sep-2001.
  Database last updated on  3-Mar-2002 19:56:53 EDT.

----------

Videotron Ltee (NETBLK-VL-D-MS-18CAD100)
  2000 Rue Berri
  Montreal, QC H2L 4V7
  CA

  Netname: VL-D-MS-18CAD100
  Netblock: 24.202.209.0 - 24.202.209.255

  Coordinator:
     Roy, Pierre  (PR163-ARIN)  abuse@videotron.ca
     514 281-0850

  Record last updated on 06-Dec-2000.
  Database last updated on  3-Mar-2002 19:56:53 EDT.

The ARIN Registration Services Host contains ONLY Internet
Network Information: Networks, ASN's, and related POC's.
Please use the whois server at rs.internic.net for DOMAIN related
Information and whois.nic.mil for NIPRNET Information.
```

This appears to an Internet service provider based in Montreal, Canada

Below is high risk advisory from Internet Security System, X-Force database.
http://www.iss.net/security_center/static/2245.php
Here is another FAQ on SANS web site talking about this infamous Trojan.
http://www.sans.org/newlook/resources/IDFAQ/subseven.htm
http://www.sans.org/newlook/resources/IDFAQ/oddports.htm
http://www.sans.org/newlook/resources/IDFAQ/zombie.htm

Sample of TCPDUMP record which are associated with this probing activity.

```
12:16:31.150575 ool-18bd69bb.dyn.optonline.net.4333 > 192.168.112.44.27374: S
542724472:542724472(0) win 16384 <mss 1460,nop,nop,sackOK> (DF) (ttl 117, id 13444)

12:16:31.160575 ool-18bd69bb.dyn.optonline.net.4334 > 192.168.112.45.27374: S
542768141:542768141(0) win 16384 <mss 1460,nop,nop,sackOK> (DF) (ttl 117, id 13445)

12:16:31.170575 24.3.50.252.1757 > 192.168.19.178.27374: S 681372183:681372183(0) win
16384 <mss 1460,nop,nop,sackOK> (DF) (ttl 117, id 54912)

12:16:31.170575 24-240-136-48.hsacorp.net.4939 >192.168.11.19.27374: S
3019773591:3019773591(0) win 16384 <mss 1460,nop,nop,sackOK> (DF) (ttl 117, id 39621)

12:16:31.170575 ool-18bd69bb.dyn.optonline.net.4335 > 192.168.112.46.27374: S
542804226:542804226(0) win 16384 <mss 1460,nop,nop,sackOK> (DF) (ttl 117, id 13446)

12:16:31.170575 cc18270-a.essx1.md.home.com.4658 > 192.168.5.88.27374: S
55455482:55455482(0) win 8192 <mss 1460,nop,nop,sackOK> (DF) (ttl 117, id 8953)

12:16:31.170575 24.3.50.252.1759 > 192.168.19.180.27374: S 681485650:681485650(0) win
16384 <mss 1460,nop,nop,sackOK> (DF) (ttl 117, id 54914)

12:16:31.170575 cc18270-a.essx1.md.home.com.4659 > 192.168.5.89.27374: S
55455483:55455483(0) win 8192 <mss 1460,nop,nop,sackOK> (DF) (ttl 117, id 9209)

12:16:31.170575 24.3.50.252.1760 > 192.168.19.181.27374: S 681550782:681550782(0) win
16384 <mss 1460,nop,nop,sackOK> (DF) (ttl 117, id 54915)

12:16:31.170575 cc18270-a.essx1.md.home.com.4660 > 192.168.5.90.27374: S
55455484:55455484(0) win 8192 <mss 1460,nop,nop,sackOK> (DF) (ttl 117, id 9465)
```

7. Evidence of attack:

This type of scan is not run on just one solo machine. I don't think I was singled out. Also this has re-occurred several times, although its intensity has declined over time. Thus I don't see any evidence of active targeting towards my host.

8. Severity:

(Criticality + Lethality) – (System countermeasures + Network countermeasures) = Severity
The severity of the attack is (5+5) – (5+5) = 0
Criticality = 5 – My host was being probed continuously. I can't have my system being compromised as it has my personal critical information; such as banking information, previous years Income tax form & other documentation.
Lethality = 5 - This attack is lethal attack, as a compromised host can be abused as being an agent for other DDOS attack, as well as the intruder can get to personal information, password to online banking, etc. A compromised host leaves a backdoor for any intruder to pry in and he can also use this compromised host as a launching pad for any attack on the Internet without the knowledge of the systems owner.
System countermeasures = 5 – In my case here my home network is up to date with virus definitions files and all other related security patches.
Network countermeasures = 5 – The fact that this traffic was dropped says a lot. The border device is configured to drop all traffic to port 27374

9. Defensive recommendation:

I recommend updating all virus definitions files. Block in bound access on the firewall or border router. Deploy personal firewall to control access as well as monitor activity. Monitor traffic going to IRC server. If there is no need for any IRC service then it can also be filter/blocked.

10. Multiple choice test question:

When you run netstat –a on your Windows NT system and if you see port TCP 27374 in listening mode, what type of service is running on that port?

A) BackOrifice Server
B) Doom Server
C) Netbus
D) SubSeven v2.0

Answer is D

Assignment 3 – "Analyze This" Scenario

Executive Summary:

The following is an analysis done for a security audit conducted at GIAC University. The log files were collected for five consecutive days starting from January 23rd thru January 27th. The log files were apparently in Snort Intrusion detection systems format. Thus making it easier to run it against Snortsnarf tool for analysis.
Three different types of files were provided, i.e. Alerts log, OOS (Out-of-Spec) logs, and Scan logs. Thus making it a total of 15 files, one of each kind for five consecutive days. The files are listed in the table below.

Alert Logs	***OOS (Out-of-Spec) Logs***	***Scan Logs***
alert.020123.gz	oos_Jan.23.2002.gz	scans.020123.gz
alert.020124.gz	oos_Jan.24.2002.gz	scans.020124.gz
alert.020125.gz	oos_Jan.25.2002.gz	scans.020125.gz
alert.020126.gz	oos_Jan.26.2002.gz	scans.020126.gz
alert.020127.gz	oos_Jan.27.2002.gz	scans.020127.gz

The analysis consists of meaningful event information generated by different computers system. The entire above logs do not all have the similar trails of events, therefore each log have been separately parsed and analyzed. Although some of the data caused an entry in alert logs as well as appeared in scan logs, while data was unique as seen in the OOS logs. The logs were concatenated into a master alert log; master OOS logs & master scan logs for the analysis process. The logs were then used to identify if any possible internal systems/networks have been compromised.

The logs analyzed do not provide sufficient information to determine the compromise vector. But security recommendations have been provided to take action to limit the vulnerability exposure and remove the threats necessary to protect the internal network.

Summary of analysis methodology is described below. A detail analysis process describing the steps involved appears in Appendix A.

Snortsnarf was used to group all of the alerts generated and rank them by volume of alerts. Most significant activity will stick out more by its sheer volume, thus identifying any misconfigured hosts/systems or any abnormal traffic that are being generated on the network. Also any skilled attacker generates stealthy type of activity, often these activities goes un-noticed as false negative or appears as least occurring alerts. Grouping these alerts by number of occurrences makes it easier to identify any such least frequent alert.

Therefore a review of alerts is done based on severity rather then the volume of alerts. A brief explanation of these severe events has also been provided. If any compromised systems have been found a defensive recommendation is provided.

"Top talkers" lists have been generated from alert & scan logs in terms of traffic generated and any registration information of any external IP addresses has been provided. This information will also provide us with targets hosts within the internal network that need to be scrutinized for any sign of being compromised. The external sources also identify hosts that are being subject to infecting/injecting any malicious type of traffic towards the internal network. Some of these hosts have also been scrutinized and correlated with GIAC analyst work. References have also been made with CERT advisories, SANS alerts, Incidents.org and if found, CVE references have also been made.

An out-of-spec log contains traffic that does not conform to TCP/IP's RFC. This log can contain less known or any unknown attacks and thus will be used to substantiate with the alert/scan logs activity.

Below is a list of all the event signatures that Snortsnarf v020126.1 reported.

All Snort signatures
SnortSnarf v020126.1

119430 alerts found using input module SnortFile Input, with sources:
Earliest alert at **00:00:00**.952321 *on 01/23/2002*
Latest alert at **00:00:00**.367864 *on 01/28/2002*

Signature (click for sig info)	# Alerts	# Sources	# Dests
MISC Large ICMP Packet	1	1	1
INFO Napster Client Data	1	1	1
TFTP - Internal UDP connection to external tftp server	1	1	1
EXPLOIT x86 setgid 0	1	1	1
RPC udp traffic contains bin sh	1	1	1
Back Orifice	1	1	1
WEB-MISC http directory traversal	1	1	1
WEB-IIS encoding access	1	1	1
ICMP Echo Request Cisco Type.x	2	1	1
Watchlist 000222 NET-NCFC	2	1	1
EXPLOIT x86 setuid 0	3	2	3
WEB-CGI formmail access	3	3	1
INFO - Possible Squid Scan	3	2	2
WEB-MISC compaq nsight directory traversal	4	3	3
ICMP SRC and DST outside network	4	2	2
TFTP - External UDP connection to internal tftp server	4	1	1
Attempted Sun RPC high port access	4	4	4
WEB-MISC 403 Forbidden	5	2	4
INFO Outbound GNUTella Connect accept	5	5	2
SUNRPC highport access!	5	2	1
Incomplete Packet Fragments Discarded	6	6	1
Queso fingerprint	7	4	2
EXPLOIT x86 NOOP	7	4	4
Port 55850 tcp - Possible myserver activity - ref. 010313-1	7	2	2
MISC traceroute	8	3	3
MISC source port 53 to <1024	8	8	3
TCP SRC and DST outside network	10	3	7
High port 65535 tcp - possible Red Worm - traffic	11	3	3

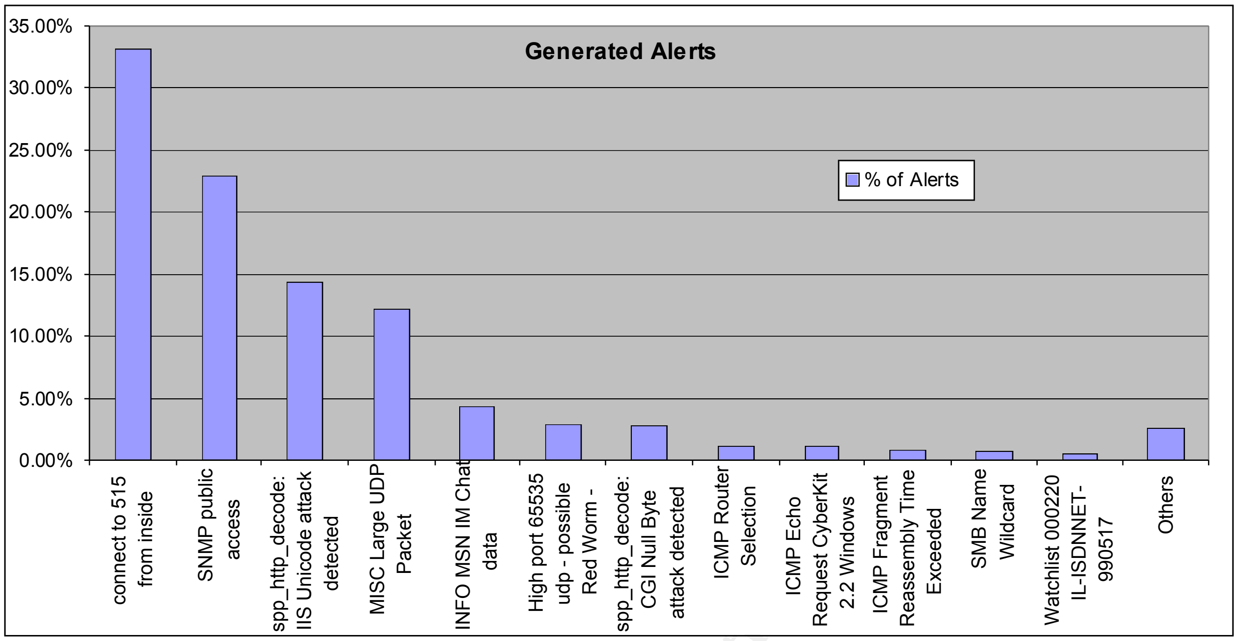
ICMP traceroute	14	8	5
ICMP Destination Unreachable (Host Unreachable)	14	1	5
INFO Inbound GNUTella Connect accept	21	2	20
WEB-FRONTPAGE _vti_rpc access	23	14	1
WEB-IIS _vti_inf access	23	13	1
SCAN Synscan Portscan ID 19104	27	27	2
ICMP Destination Unreachable (Protocol Unreachable)	34	3	4
Possible trojan server activity	40	3	3
WEB-CGI scriptalias access	45	4	1
FTP DoS ftpd globbing	53	1	1
INFO Inbound GNUTella Connect request	57	51	2
INFO Possible IRC Access	70	14	14
INFO FTP anonymous FTP	70	5	21
WEB-MISC Attempt to execute cmd	74	9	7
ICMP Echo Request Nmap or HPING2	77	7	3
EXPLOIT NTPDX buffer overflow	86	14	9
SCAN Proxy attempt	90	13	5
NMAP TCP ping!	92	8	4
ICMP Destination Unreachable (Communication Administratively Prohibited)	196	1	10
WEB-IIS view source via translate header	250	6	1
ICMP Echo Request Windows	262	8	23
Null scan!	328	58	4
ICMP Echo Request L3retriever Ping	404	24	7
ICMP Echo Request BSDtype	729	5	7
Watchlist 000220 IL-ISDNNET-990517	741	19	7
SMB Name Wildcard	822	76	79
ICMP Fragment Reassembly Time Exceeded	1055	17	26
ICMP Echo Request CyberKit 2.2 Windows	1470	3	4
ICMP Router Selection	1474	130	1
spp_http_decode: CGI Null Byte attack detected	3283	10	21
High port 65535 udp - possible Red Worm - traffic	3509	70	115
INFO MSN IM Chat data	5200	78	75
MISC Large UDP Packet	14593	16	6
spp_http_decode: IIS Unicode attack detected	17200	94	370
SNMP public access	27423	15	140
connect to 515 from inside	39465	69	1

Alerts based on occurrences:

Below is a table that list the alert based on the number of occurrences and the overall percentage of these alerts.

Alerts	Total Alerts	% of overall alerts
connect to 515 from inside	39465	33.04%
SNMP public access	27423	22.96%
spp_http_decode: IIS Unicode attack detected	17200	14.40%
MISC Large UDP Packet	14593	12.22%
INFO MSN IM Chat data	5200	4.35%
High port 65535 udp - possible Red Worm - traffic	3509	2.94%
spp_http_decode: CGI Null Byte attack detected	3283	2.75%
ICMP Router Selection	1474	1.23%
ICMP Echo Request CyberKit 2.2 Windows	1470	1.23%
ICMP Fragment Reassembly Time Exceeded	1055	0.88%
SMB Name Wildcard	822	0.69%
Watchlist 000220 IL-ISDNNET-990517	741	0.62%
ICMP Echo Request BSDtype	729	0.61%
ICMP Echo Request L3retriever Ping	404	0.34%
Null scan!	328	0.27%
ICMP Echo Request Windows	262	0.22%
WEB-IIS view source via translate header	250	0.21%
ICMP Destination Unreachable (Communication Administratively Prohibited)	196	0.16%
NMAP TCP ping!	92	0.08%
SCAN Proxy attempt	90	0.08%
EXPLOIT NTPDX buffer overflow	86	0.07%
ICMP Echo Request Nmap or HPING2	77	0.06%
WEB-MISC Attempt to execute cmd	74	0.06%
INFO Possible IRC Access	70	0.06%
INFO FTP anonymous FTP	70	0.06%
INFO Inbound GNUTella Connect request	57	0.05%
FTP DoS ftpd globbing	53	0.04%
WEB-CGI scriptalias access	45	0.04%
Possible trojan server activity	40	0.03%
ICMP Destination Unreachable (Protocol Unreachable)	34	0.03%
SCAN Synscan Portscan ID 19104	27	0.02%
WEB-FRONTPAGE _vti_rpc access	23	0.02%
WEB-IIS _vti_inf access	23	0.02%
INFO Inbound GNUTella Connect accept	21	0.02%
ICMP traceroute	14	0.01%
ICMP Destination Unreachable (Host Unreachable)	14	0.01%

High port 65535 tcp - possible Red Worm - traffic	11	0.01%
TCP SRC and DST outside network	10	0.01%
MISC traceroute	8	0.01%
MISC source port 53 to <1024	8	0.01%
Queso fingerprint	7	0.01%
EXPLOIT x86 NOOP	7	0.01%
Port 55850 tcp - Possible myserver activity - ref. 010313-1	7	0.01%
Incomplete Packet Fragments Discarded	6	0.01%
SUNRPC highport access!	5	0.00%
INFO Outbound GNUTella Connect accept	5	0.00%
WEB-MISC 403 Forbidden	5	0.00%
WEB-MISC compaq nsight directory traversal	4	0.00%
TFTP - External UDP connection to internal tftp server	4	0.00%
Attempted Sun RPC high port access	4	0.00%
ICMP SRC and DST outside network	4	0.00%
WEB-CGI formmail access	3	0.00%
INFO - Possible Squid Scan	3	0.00%
EXPLOIT x86 setuid 0	3	0.00%
ICMP Echo Request Cisco Type.x	2	0.00%
Watchlist 000222 NET-NCFC	2	0.00%
EXPLOIT x86 setgid 0	1	0.00%
TFTP - Internal UDP connection to external tftp server	1	0.00%
RPC udp traffic contains bin sh	1	0.00%
Back Orifice	1	0.00%
MISC Large ICMP Packet	1	0.00%
WEB-IIS encoding access	1	0.00%
WEB-MISC http directory traversal	1	0.00%
INFO Napster Client Data	1	0.00%



Analysis:

Universities by their default policies are open in nature and are very desirable target for crackers. These would be crackers usually look for poorly patched/configured systems to compromise and use them to perform malicious activity against other's network.

The report below will make an attempt to discuss the activity that's taking place on the network and the effect of that activity. Throughout the report as well as towards the end of the report, defensive recommendations will be included that will help to alleviate the risk to these vulnerable systems.

The above Link graph that defines percentage of the network traffic provides some insight into the type of activity being generated on the network. Each of the alert categories found most critical are discussed later on in detail within this report. As you can see 33% of the traffic triggered is generated by "connect to 515 from the inside" alert and 23% is SNMP activity. Although not necessarily malicious in nature it should be scrutinized to verify the source/destination of this traffic to determine if it's legitimate. Lets look at these two alerts to begin with, as more than half (33.04 + 22.96= 51%) of the network activity is triggering these 2 alerts.

Connect to 515 from inside:

Description:
This alert that triggered the rule of "Connect to 515 from the inside" applies to internal host trying to make a connection to open lpd printers. Apparently there is a line spooler control on TCP Port 515 and Spooler on UDP port 515. As mentioned port 515 is used by LPRng print services and some of the versions are vulnerable to attacks.

According to *CERT® Advisory CA-2000-22 Input Validation Problems in LPRng* http://www.cert.org/advisories/CA-2000-22.html,

A popular replacement software package to the BSD lpd printing service called LPRng contains at least one software defect, known as a "format string vulnerability,"[1] which may allow remote users to execute arbitrary code on vulnerable systems.

SANS has also posted an alert referencing the same issue for port 515, you can find this at http://www.sans.org/newlook/alerts/port515.htm

Statistics:

There were 39465 alerts generated from 69 internal hosts trying to connect to one internal host MY.NET.150.198

Top 10 sources triggering this attack signature

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
MY.NET.153.114	6446	6821	1	21
MY.NET.153.111	4308	4316	1	2
MY.NET.153.119	3678	3719	1	9
MY.NET.153.118	2669	2685	1	3
MY.NET.153.202	2242	2243	1	2
MY.NET.153.113	1645	1744	1	10
MY.NET.153.112	1601	1648	1	4
MY.NET.153.123	1106	1295	1	13
MY.NET.153.122	983	1000	1	4
MY.NET.153.173	894	894	1	1

Destinations receiving this attack signature

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
MY.NET.150.198	39465	39465	69	69

Correlation:
Since this was strictly internal host making attempt to internal host. I did not look for any correlations. Below is an example of attempt to port 515; traces from alert logs for the said period defined earlier.

```
01/27-19:47:17.213148 [**] connect to 515 from inside [**] MY.NET.153.114:3987 ->
MY.NET.150.198:515
01/27-19:47:17.214639 [**] connect to 515 from inside [**] MY.NET.153.114:3987 ->
```

MY.NET.150.198:515
01/27-19:47:17.215869 [**] connect to 515 from inside [**] MY.NET.153.114:3987 -> MY.NET.150.198:515
01/27-19:47:17.218467 [**] connect to 515 from inside [**] MY.NET.153.114:3987 -> MY.NET.150.198:515
01/27-19:47:17.219869 [**] connect to 515 from inside [**] MY.NET.153.114:3987 -> MY.NET.150.198:515

Defensive recommendation:
Examine the internal host MY.NET.150.198 and the need for lpd service running. If it is needed, make sure the software is latest and all the security patches have been applied.

SNMP public access

Description:
The Simple Network Management Protocol (SNMP) is an application layer protocol used to manage network devices. This protocol assist network/system administrators to manage network performance, locate any networking issue and resolve this problem. It can also be used as a tool to baseline networks and plan upgrades, etc.

Internal host who have SNMP agent installed triggers the "SNMP public access" alert. When the agent is installed on a host, it has a default community string of "public". To keep it secured from any intruder it is recommended to change this community string to something else. If intercepted, SNMP can provide extensive information about the network segment of any given device.

Here is a FAQ's on SNMP as a reference.
http://www.faqs.org/faqs/by-newsgroup/comp/comp.protocols.snmp.html

Statistics/Analysis:

Top 10 sources triggering this attack signature

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
MY.NET.70.177	11203	11229	25	25
MY.NET.88.240	6613	6613	1	1
MY.NET.150.198	2954	2954	102	102
MY.NET.150.41	2320	2320	1	1
MY.NET.153.220	2014	2014	1	1
MY.NET.150.245	1260	1260	1	1
MY.NET.186.10	625	625	1	1
MY.NET.84.155	241	241	16	16
MY.NET.150.49	106	1582	6	10

MY.NET.183.11	46	46	5	5

Top 10 Destinations receiving this attack signature

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
MY.NET.150.195	6637	6640	6	8
MY.NET.152.109	5613	5613	4	4
MY.NET.5.96	1830	2243	1	32
MY.NET.5.37	1796	1796	1	1
MY.NET.5.127	1789	1789	1	1
MY.NET.5.128	1789	1789	1	1
MY.NET.151.114	1524	1527	3	5
MY.NET.5.249	1399	1454	2	5
MY.NET.5.141	987	1008	1	6
MY.NET.5.92	675	683	1	5

There are all together 15 different hosts trying to connect to SNMP port to 140 different internal hosts residing on MY.NET.x.x

It can be assumed that these systems may be misconfigured. The ratio of 15 hosts to 140 hosts tends to be a lot.

Correlation:
Although I could not find any of the above source IP addresses to correlate but MY.NET.101.192 has appeared in the past analysis with SNMP public access, "public" as the community string.
http://www.giac.org/practical/Roland_Gerlach_GCIA.html

As quoted from Roland Gerlach practicals

"A significant number of **SNMP public access** Alerts relate to Internal traffic which confirms the use of "public" Community String on: MY.NET.14.1, MY.NET.50.154, and MY.NET.101.192.
It is recommended that all SNMP capable systems should be configured with a more secure *Read* and *Write* SNMP Community strings."

Scan logs also provides the below trace which has one of the source host which is listed as one of the top ten talkers of this alert. There were several attempts made by this source to various destination on port 161, either doing an "snmpget" or snmpset" as an example.

```
Jan 23 00:42:59 MY.NET.150.198:3326 -> MY.NET.106.199:161 UDP
Jan 23 00:43:02 MY.NET.150.198:3328 -> MY.NET.162.203:161 UDP
Jan 23 00:43:02 MY.NET.150.198:3331 -> MY.NET.153.219:161 UDP
Jan 23 00:43:02 MY.NET.150.198:3334 -> MY.NET.85.25:161 UDP
Jan 23 00:43:02 MY.NET.150.198:3343 -> MY.NET.160.148:161 UDP
Jan 23 00:43:05 MY.NET.150.198:3344 -> MY.NET.190.13:161 UDP
Jan 23 00:43:07 MY.NET.150.198:3348 -> MY.NET.71.24:161 UDP
Jan 23 00:43:43 MY.NET.150.198:3420 -> MY.NET.163.108:161 UDP
Jan 23 00:43:43 MY.NET.150.198:3423 -> MY.NET.85.40:161 UDP
Jan 23 00:43:44 MY.NET.150.198:3428 -> MY.NET.163.11:161 UDP
Jan 23 00:43:44 MY.NET.150.198:3431 -> MY.NET.87.218:161 UDP
Jan 23 00:43:44 MY.NET.150.198:3432 -> MY.NET.162.109:161 UDP
Jan 23 00:43:46 MY.NET.150.198:3435 -> MY.NET.162.242:161 UDP
Jan 23 00:43:46 MY.NET.150.198:3438 -> MY.NET.70.170:161 UDP
Jan 23 00:43:46 MY.NET.150.198:3444 -> MY.NET.86.39:161 UDP
Jan 23 00:43:46 MY.NET.150.198:3449 -> MY.NET.115.12:161 UDP
Jan 23 00:43:47 MY.NET.150.198:3451 -> MY.NET.138.205:161 UDP
Jan 23 00:43:49 MY.NET.150.198:3452 -> MY.NET.138.230:161 UDP
```

Defensive recommendation:
It should be noted and should be a practice not to use the default "public" as the community strings. Also make sure you are using something secure as a community strings, instead of being in a practice of using company's stock ticker, private, etc.

Below are URL to consult for securing SNMP:
http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/snmp.htm
http://www.sans.org/topten.htm

Comment:

As can be seen from both of the above alerts the source as well as the destination host were within the internal network. This can be categorized as legitimate internal traffic. The configuration file for snort i.e. snort.conf can be configured in this case to properly define the variable for HOME_NET to reduce these false positives alerts. This can drastically cut down the alerts being logged, in our case by 51%.

Top Talkers Analysis:

"Top Ten Talkers" list in terms of alert logs.

Rank	Total # Alerts	**Source IP**	# Signatures triggered	Destinations involved
rank #1	11229 alerts	**MY.NET.70.177**	2 signatures	(25 destination IPs)
rank #2	6821 alerts	**MY.NET.153.114**	5 signatures	(21 destination IPs)
rank #3	6613 alerts	**MY.NET.88.240**	1 signatures	MY.NET.150.195
rank #4	5008 alerts	**63.210.47.81**	2 signatures	MY.NET.153.45
rank #5	4970 alerts	**63.250.208.34**	1 signatures	MY.NET.151.63
rank #6	4316 alerts	**MY.NET.153.111**	2 signatures	224.0.0.2, MY.NET.150.198
rank #7	3719 alerts	**MY.NET.153.119**	3 signatures	(9 destination IPs)
rank #8	2985 alerts	**MY.NET.150.121**	2 signatures	224.0.0.2, 216.241.219.14
rank #9	2954 alerts	**MY.NET.150.198**	1 signatures	(102 destination IPs)
rank #10	2685 alerts	**MY.NET.153.118**	3 signatures	(3 destination IPs)

As seen in the above Top talkers list most of the traffic is generated from internal host, with an exception of 2 external sources.

Below is the registration information for the two external source IP addresses from alert logs "top talkers" list:

whois -h whois.arin.net 63.210.47.81

```
Level 3 Communications, Inc. (NETBLK-LEVEL4-CIDR)
   1450 Infinite Drive
   Louisville, CO 80027
   US

   Netname: LEVEL4-CIDR
   Netblock: 63.208.0.0 - 63.215.255.255
   Maintainer: LVLT
```

```
      Coordinator:
         level Communications  (LC-ORG-ARIN)   ipaddressing@level3.com
         +1 (877) 453-8353

      Domain System inverse mapping provided by:

      NS1.LEVEL3.NET  209.244.0.1
      NS2.LEVEL3.NET  209.244.0.2

      ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

      Record last updated on 30-May-2001.
      Database last updated on  10-Mar-2002 19:57:29 EDT.
```

whois -h whois.arin.net 63.250.208.34

```
   Yahoo! Broadcast Services, Inc. (NETBLK-NETBLK2-YAHOOBS)
      2914 Taylor st
      Dallas, TX 75226
      US

      Netname: NETBLK2-YAHOOBS
      Netblock: 63.250.192.0 - 63.250.223.255
      Maintainer: YAHO

      Coordinator:
         Bonin, Troy  (TB501-ARIN)   netops@broadcast.com
         214.782.4278 ext. 2278

      Domain System inverse mapping provided by:

      NS.BROADCAST.COM  206.190.32.2
      NS2.BROADCAST.COM  206.190.32.3

      ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

      Record last updated on 29-Jun-2001.
      Database last updated on  10-Mar-2002 19:57:29 EDT.
```

The above two source have generated 9977 occurrences of "MISC Large UDP Packet"

MISC Large UDP Packet

Description:
This can be a sign of UDP flood. If several large UDP packets are sent to a given host it can cause a Denial of Service attack. There can possibly also be a session of UDP covert channel used by a cracker to communicate with a compromised system. Or this packet can also be crafted as can be seen by source/destination port being 0.

This also can be a fragmented packet without any TCP header as in fragmented packet; the first packet is the only packet that contains the psuedo headers of any protocol.

The Snort rule from the current rule set that triggered this alert is defined below:

```
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;)
```

This activity may also constitute any gaming activity as seen most of the gaming activity uses both TCP as well as UDP ports. TCP is used to initiate a session and then TCP and or UDP is used for any subsequent packets after the session is being established.

There is a great white paper, which discusses the Internet gaming activity as well as talks about the various port used for gaming activities.
http://www.incidents.org/detect/gaming.php
Matt Scarborough. "What are some of the signs of Internet Gaming"

A point to also note is that Microsoft requires their Proxy server to open port 0 as a destination port for subsequent UDP packets; for their DirectPlay games on MSN zone. See below
http://support.microsoft.com/support/Games/Zone/FAQ/connect.asp

Additional analysis shows various other hosts sending MISC large UDP packet to the internal network. As mentioned this should be sign of caution. Payload of these packets needs to be examined to determine what kind of activity is going on. Below is a table that lists the top ten talkers for this category of alert apart from the overall top ten talkers list table mentioned above. Our 2-source host in reference to overall top ten categories (above) appears as the 2 top listed source hosts for this type of activity.

Statistics/Analysis:
Here is list of top 10-source host that triggered this alert,

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
63.210.47.81	5007	5008	1	1
63.250.208.34	4970	4970	1	1
211.202.0.47	1012	1012	1	1
210.181.96.14	905	905	1	1
216.106.166.212	767	767	1	1
211.233.70.163	615	615	1	1
217.15.64.179	313	313	1	1
211.174.63.108	241	241	1	1
211.233.70.162	219	219	1	1
211.233.70.165	217	217	1	1

Destinations receiving this attack

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
MY.NET.153.45	5916	5918	3	4
MY.NET.151.63	4971	4971	2	2

MY.NET.153.171	2262	2293	4	12
MY.NET.153.196	53	61	1	5
MY.NET.150.143	1	2	1	2
MY.NET.153.185	1390	1398	5	10

MISC Large UDP Packet	16 sources	6 destinations

Below is a traces captured from the alert logs which shows source and destination port 0 used in this case;

```
01/23-13:53:39.024929  [**] MISC Large UDP Packet [**] 63.210.47.81:0 ->
MY.NET.153.45:0
01/23-13:53:39.225247  [**] MISC Large UDP Packet [**] 63.210.47.81:0 ->
MY.NET.153.45:0
01/23-13:53:39.522443  [**] MISC Large UDP Packet [**] 63.210.47.81:0 ->
MY.NET.153.45:0
01/23-13:53:39.929036  [**] MISC Large UDP Packet [**] 63.210.47.81:0 ->
MY.NET.153.45:0
01/23-13:53:40.022685  [**] MISC Large UDP Packet [**] 63.210.47.81:0 ->
MY.NET.153.45:0
01/23-13:53:40.320288  [**] MISC Large UDP Packet [**] 63.210.47.81:0 ->
MY.NET.153.45:0
01/23-13:53:40.522329  [**] MISC Large UDP Packet [**] 63.210.47.81:0 ->
MY.NET.153.45:0
01/23-13:53:40.617223  [**] MISC Large UDP Packet [**] 63.210.47.81:0 ->
MY.NET.153.45:0
```

There were altogether 16 hosts probing 6 listeners for this alert. Scan logs also show various other probes to specially to port 1221 which is used by sweetWARE Apps port 1221, some sort of Warez Applications.

Below is a trace as it appears in the scan logs;

```
Jan 23 14:22:35 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:22:38 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:22:43 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:22:46 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:22:51 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:22:55 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:22:59 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:23:02 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:23:04 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:23:11 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:23:14 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:23:19 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:23:22 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:23:24 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:23:31 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
Jan 23 14:23:35 63.210.47.81:44230 -> MY.NET.153.45:1221 UDP
```

Correlation:

In Steve Lukacs's GCIA Practical, he documents the same source IP address: port as the one above; 63.210.47.81:0 which is listed as one of the Top ten source IP address that has generated 58 "Incomplete Packet Fragments Discarded" alert.
Refer the below URL:
http://www.giac.org/practical/Steve_Lukacs_GCIA.doc

Gregory Lajon in his GIAC practical categorized this traffic as suspicious.
Quoted from his practical, he mention…………..." We see a large amount of traffic coming from port 0 to port 0. It is interesting to notice that 61 different IPs have sent large UDP Packet with source port = 0 and dest port=0. This is usualy a bad sign, it means that the packet is crafted."

Below is a scan of traces from his practical…

```
09/05-19:03:21.044684  [**] MISC Large UDP Packet [**] 61.153.17.244:0 ->
MY.NET.111.142:0
09/05-19:03:21.179799  [**] MISC Large UDP Packet [**] 61.153.17.24:0 ->
MY.NET.111.221:0
09/05-19:03:21.276811  [**] MISC Large UDP Packet [**] 61.153.17.24:0 ->
MY.NET.111.221:0
09/05-19:03:21.952701  [**] MISC Large UDP Packet [**] 61.153.17.244:0 ->
MY.NET.111.142:0
```

Both the address at the time of the practical belonged to the same owner. The IP address may be spoofed in this situation. It may be a DOS attempt as claimed by Gregory Lajon.
http://www.giac.org/practical/Gregory_Lajon_GCIA.doc

In our scenario above, both the addresses belong to different owners as seen by the registration information above.

```
01/23-13:52:47.116069 [**] MISC Large UDP Packet [**] 63.210.47.81:0 -> MY.NET.153.45:0
01/23-13:52:49.928087 [**] MISC Large UDP Packet [**] 63.210.47.81:0 -> MY.NET.153.45:0
01/23-13:52:50.724740 [**] MISC Large UDP Packet [**] 63.210.47.81:0 -> MY.NET.153.45:0
01/23-13:52:51.118667 [**] MISC Large UDP Packet [**] 63.210.47.81:0 -> MY.NET.153.45:0
01/23-13:52:54.728018 [**] MISC Large UDP Packet [**] 63.210.47.81:0 -> MY.NET.153.45:0
```

Defensive Recommendation:

Where it is possible implement perimeter security at the border router. Apply ACL and deny any inbound UDP access on the edge router/Firewall. Allow incoming traffic only to specified internal resources that need access from any external source.

For more information on blocking inbound traffic, see…
http://rr.sans.org/firewall/blocking_cisco.php
http://rr.sans.org/firewall/blocking_ipchains.php

Now let's take a look at the traffic & alerts generated by the other internal sources from the "top ten talkers" list, in term of alert logs:

Top source talker analysis: (based on the top ten talkers list above)

Source host MY.NET. 70.177 is ranked the number 1 talker and has generated 11229 alerts, 26 instances of SMB Name Wildcard and 11203 instances of SNMP public access (already discussed earlier). As can be seen 99 % of the traffic is SNMP public access. This is classified as legitimate traffic as explained earlier. SMB Name Wildcard will be addressed later on in the report.

MY.NET.153.114 generated 6821 alerts with the following type of traffic.
1 instances of SMB Name Wildcard
3 instances of INFO MSN IM Chat data
38 instances of ICMP Router Selection
333 instances of spp_http_decode: IIS Unicode attack detected
6446 instances of connect to 515 from inside
As you can see 94.50 % of the traffic can be classified as legitimate going to a lpd printer, destined for internal host MY.NET.150.198

MY.NET.88.240, which is ranked, third has triggered 6613 instances of SNMP public access alert. This per earlier discussion can also be classified as legitimate internal traffic.

The two external hosts were discussed earlier generating 9977 instances of MISC Large UDP packets. One of the host 63.210.47.81 also generated 1 instances of High port 65535 udp - possible Red Worm – traffic; this will be addressed later on in the report.

MY.NET.153.111 has triggered 4316 alerts, 8 instances of ICMP Router Selection and 4308 instances of connect to 515 from inside. 99.81 % of the traffic generated by this classified as legitimate. The ICMP router selection was sent to router multicast address of 224.0.0.2, router that supports RFC 1256 will respond with a router advertisement which enables the host to choose its default gateway.

MY.NET.153.119 generated 3719 alerts of which 98.90% is legitimate traffic. The following is the category of the alerts.
6 instances of ICMP Router Selection
35 instances of spp_http_decode: IIS Unicode attack detected
3678 instances of connect to 515 from inside.

MY.NET.150.121 ranked as 8th triggered 2985 alerts, 3 instances of ICMP Router Selection and 2982 instances of spp_http_decode: CGI Null Byte attack detected, this will be discussed later on in the report.

MY.NET.150.198 generated 100% of SNMP public access alerts.

MY.NET.153.118 triggered 3 different type of alert, 4 instances of spp_http_decode: IIS Unicode attack detected, 12 instances of ICMP Router Selection and 2669 instances of connect to 515 from inside.

Alerts Analysis:

spp_http_decode: IIS Unicode attack detected:

Description:
http://www.cve.mitre.org, which standardizes names for security problems, has this attack on their Common Vulnerabilities and Exposures list. CVE-2000-0884 description is as follows

"IIS 4.0 and 5.0 allows remote attackers to read documents outside of the web root, and possibly execute arbitrary commands, via malformed URLs that contain UNICODE encoded characters, aka the "Web Server Folder Traversal" vulnerability."

This access will potentially enable a user who visited the web site to gain additional privileges on the host machine. On acquiring these privileges it can enable the intruder to manipulate the data, execute residing code on the host, or upload new code to the host to run it.

Below are some rules that can be applied to snort IDS system to generate alerts.

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 80: (msg:"spp_http_decode: IIS Unicode attack detected"; flags: A+; content: "%c0";)
alert tcp $EXTERNAL_NET any -> $HOME_NET 80: (msg:"spp_http_decode: IIS Unicode attack detected"; flags: A+; content: "%af";)
```

Statistics/Analysis:
SnortSnarf found 94 source hosts and 370 destination hosts involved in this spp_http_decode: IIS Unicode Attack. Of these 94 sources 87 hosts were from MY.NET.x.x network generating this traffic destined to external IP addresses. The list below shows the seven sources that were triggering these alerts from the outside the MY.NET.x.x network.

External sources:

1. 209.88.103.73
2. 209.88.103.74
3. 209.88.103.75
4. 217.88.128.228
5. 217.88.135.202

6. 203.227.74.100
7. 130.205.92.178

These hosts along that generated "spp_http_decode:IIS Unicode attack" also generated "WEB-MISC Attempt to execute cmd" alerts as can be seen from the traces from alert logs below. 92.55% of the traffic that generated these alerts was from the internal host. Thus this is a good indicative of internal systems that may have been compromised as this malicious traffic is seen originating from the internal network.

```
01/27-23:51:22.926260 [**] spp_http_decode: IIS Unicode attack detected [**]
209.88.103.75:1199 -> MY.NET.5.95:80
01/27-23:51:23.271307 [**] spp_http_decode: IIS Unicode attack detected [**]
209.88.103.74:3084 -> MY.NET.5.95:80

01/27-23:51:22.926260 [**] spp_http_decode: IIS Unicode attack detected [**]
209.88.103.75:1199 -> MY.NET.5.95:80
01/27-23:51:22.926260 [**] WEB-MISC Attempt to execute cmd [**] 209.88.103.75:1199 ->
MY.NET.5.95:80
01/27-23:51:23.451276 [**] WEB-MISC Attempt to execute cmd [**] 209.88.103.75:2594 ->
MY.NET.5.95:80
01/23-20:32:04.204515 [**] WEB-MISC Attempt to execute cmd [**] 203.227.74.100:17796 ->
MY.NET.5.249:80
01/23-20:32:04.733228 [**] WEB-MISC Attempt to execute cmd [**] 203.227.74.100:53408 ->
MY.NET.5.249:80
01/23-20:32:04.735261 [**] WEB-MISC Attempt to execute cmd [**] 203.227.74.100:53408 ->
MY.NET.5.249:80
01/23-20:32:05.209215 [**] spp_http_decode: IIS Unicode attack detected [**]
203.227.74.100:13938 -> MY.NET.5.249:80
01/23-20:32:05.209215 [**] spp_http_decode: IIS Unicode attack detected [**]
203.227.74.100:13938 -> MY.NET.5.249:80
01/23-20:32:05.209215 [**] spp_http_decode: IIS Unicode attack detected [**]
203.227.74.100:13938 -> MY.NET.5.249:80
```

Correlation:
A great correlation I found was in Ryan Quan's paper, below is the trace from his practical.

```
[**] 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                                      ....
```

As you can see in the below trace, in Ryan Quan words………… "The attacker tried to constructs URL commands to move within the machine. In this particular case, the attacker first did a "dir" command, to view the contents of the hard drive. This is accomplished by placing "../" within the url.
For example: http://<any server name>/scripts/..%c1%pc../winnt/system32/cmd.exe?/c+dir+c:\
You can see in the Snort trace that the attacker did the same Unicode replacements in the above detect.

```
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
```

Defensive Recommendation:
Make sure you apply all the service pack for Windows NT along with all the security patches. Also make sure that you update Internet Explorer with its latest service packs too. Update you anti-virus software to its latest definition files. There is actually an "IIS Lockdown Tool" now available from Microsoft that you can download for free and use. Also make sure that you do not allow traffic to initiate from your web server, just make sure it is set up to accept incoming http/https traffic.

WEB-MISC Attempt to execute cmd

Description:
This is Code Red and it's variant, i.e. Code Red II, Nimda, etc. Another worm exploiting buffer overflow.

As shown in the table below there 7 listeners and 9 talkers of this attack signature. This

WEB-MISC Attempt to execute cmd	9 sources	7 destinations

Top 5 hosts generating these alerts

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)

203.227.74.100	30	47	2	2
217.88.135.202	12	18	6	6
217.88.128.228	10	15	5	5
209.88.103.73	8	13	1	1
209.88.103.74	4	5	1	1

Top 5 Destination receiving this attack signature

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
MY.NET.5.249	25	1454	3	5
MY.NET.5.95	18	97	5	10
MY.NET.5.141	13	1008	3	6
MY.NET.5.96	7	2243	3	32
MY.NET.150.83	5	15	2	8

As you can see the source IP addresses are from the external network. This would indicate that if any one of the hosts is vulnerable they could be compromised easily.

```
01/23-20:32:02.603709 [**] WEB-MISC Attempt to execute cmd [**] 203.227.74.100:29542 ->
MY.NET.5.249:80
01/23-20:32:03.112911 [**] WEB-MISC Attempt to execute cmd [**] 203.227.74.100:19812 ->
MY.NET.5.249:80
01/23-20:32:03.653166 [**] WEB-MISC Attempt to execute cmd [**] 203.227.74.100:61809 ->
MY.NET.5.249:80
01/23-20:32:04.203467 [**] WEB-MISC Attempt to execute cmd [**] 203.227.74.100:17796 ->
MY.NET.5.249:80
01/23-20:32:04.204515 [**] WEB-MISC Attempt to execute cmd [**] 203.227.74.100:17796 ->
MY.NET.5.249:80
01/23-20:32:04.733228 [**] WEB-MISC Attempt to execute cmd [**] 203.227.74.100:53408 ->
MY.NET.5.249:80
```

Here is the registration information of the top talker of this alert in our analysis

```
Trying whois -h whois.apnic.net 203.227.74.100

% Rights restricted by copyright. See http://www.apnic.net/db/dbcopyright.html
% (whois7.apnic.net)

inetnum:     203.226.0.0 - 203.231.255.255
netname:     KRNIC-KR
descr:     KRNIC
descr:     Korea Network Information Center
```

```
country:    KR
admin-c:    HM127-AP
tech-c:     HM127-AP
remarks:    ****************************************
remarks:    KRNIC is the National Internet Registry
remarks:    in Korea under APNIC. If you would like to
remarks:    find assignment information in detail
remarks:    please refer to the KRNIC Whois DB
remarks:    http://whois.nic.or.kr/english/index.html
remarks:    ****************************************
mnt-by:     APNIC-HM
mnt-lower:  MNT-KRNIC-AP
changed:    hostmaster@apnic.net 19981001
changed:    hostmaster@apnic.net 20010606
source:     APNIC

person:     Host Master
address:    Korea Network Information Center
address:    Narajongkeum B/D 14F, 1328-3, Seocho-dong, Seocho-ku, Seoul, 137-
070, Republic of Korea
country:    KR
phone:      +82-2-2186-4500
fax-no:     +82-2-2186-4496
e-mail:     hostmaster@nic.or.kr
nic-hdl:    HM127-AP
mnt-by:     MNT-KRNIC-AP
changed:    hostmaster@nic.or.kr 20010514
source:     APNIC

inetnum:    203.227.64.0 - 203.227.79.255
netname:    DAESUNGNET-KR
descr:      Daesung Group
descr:      155-2 Kwanhoon-Dong Chongro-Ku seoul
descr:      SEOUL
descr:      110-300
country:    KR
admin-c:    EK31-KR
tech-c:     EK32-KR
remarks:    This IP address space has been allocated to KRNIC.
remarks:    For more information, using KRNIC Whois Database
remarks:    whois -h whois.nic.or.kr
remarks:    This information has been partially mirrored by APNIC from
remarks:    KRNIC. To obtain more specific information, please use the
remarks:    KRNIC whois server at whois.krnic.net.
mnt-by:     MNT-KRNIC-AP
changed:    hostmaster@nic.or.kr 20020204
```

```
source:     KRNIC

person:     Euikeun Kim
country:    KR
phone:      02 735 5671
fax-no:     02 733 6168
e-mail:     master@hawk.daesung.co.kr
nic-hdl:    EK31-KR
remarks:    This information has been partially mirrored by APNIC from
remarks:    KRNIC. To obtain more specific information, please use the
remarks:    KRNIC whois server at whois.krnic.net.
mnt-by:     MNT-KRNIC-AP
changed:    hostmaster@nic.or.kr 20020204
source:     KRNIC
```

Correlations:
CVE-1999-0233 describes the following…
IIS allows users to execute arbitrary commands using .bat or .cmd files.
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0233
http://www.cert.org/incident_notes/IN-2001-09.html
http://www.incidents.org/react/code_redII.php
http://www.incidents.org/react/code_red.php

Defensive Recommendation:
Make sure you are up to date on your patches. Also there are numerous tool available to remove Nimda/Code Red but your best bet is to backup your data, reload the OS and all related applications. Make sure you virus definitions files are also up to date.

High port 65535 udp – possible Red Worm - traffic

Description:
There were 70 sources found probing 115 destination hosts on the internal network. As mentioned these probe were from the outside. This could be a case of Red worm also known as Adore worm, it scans the Internet looking for vulnerable Linux host. It usually looks for vulnerable services such as rpc-statd, wu-ftpd, LPRng, and DNS BIND. Once a system is compromised it move its self to /usr/bin/adore. It also send information out such as the /etc/shadow and or ifconfig information. A detailed analysis can be found at the url below…
http://rr.sans.org/threats/mutation.php

Also note that there is a RC1 (Remote Control 1) Trojan listens which also listens on UDP port 65535. Here is some more information on RC1 Trojan.
http://www.glocksoft.com/trojan_list/RC1_trojan.htm
http://www.sans.org/newlook/resources/IDFAQ/oddports.htm

Statistics/Analysis:

Name:	RC1 trojan
Aliases:	Remote Control, Rc,
Ports:	65535 (port can be changed)
Files:	Rc1.zip - 2,005,874 bytes Service.zip - Setup.exe - 59,904 bytes .... and many more
Created:	Aug 1997
Requires:	N/A
Actions:	Remote Access
Versions:	1.4,
Registers:	Does not register.
Notes:	Works on Windows 95, 98 and NT.
Country:	N/A
Program:	Written in Visual Basic.

The connection involves both the source as well as the destination port to be 65535
These traces could be either of the two types of activity.

```
01/24-14:31:00.796037 [**] High port 65535 udp - possible Red Worm - traffic [**]
66.38.185.141:65535 -> MY.NET.88.163:65535
01/24-14:31:45.372070 [**] High port 65535 udp - possible Red Worm - traffic [**]
66.38.185.141:65535 -> MY.NET.88.163:65535
01/24-14:38:40.152364 [**] High port 65535 udp - possible Red Worm - traffic [**]
66.38.185.141:65535 -> MY.NET.88.163:65535
01/24-14:40:02.247761 [**] High port 65535 udp - possible Red Worm - traffic [**]
66.38.185.141:65535 -> MY.NET.88.163:65535
01/24-14:40:48.879564 [**] High port 65535 udp - possible Red Worm - traffic [**]
66.38.185.141:65535 -> MY.NET.88.163:65535
01/24-14:40:49.051477 [**] High port 65535 udp - possible Red Worm - traffic [**]
66.38.185.141:65535 -> MY.NET.88.163:65535
```

Source host seen in this alerts have also generated traffic that triggered the "Exploit NTPDX buffer overflow" as well as "TFTP – External UDP connection to internal tftp server" See below traces…………

```
01/24-12:21:11.924281 [**] High port 65535 udp - possible Red Worm - traffic [**]
66.38.185.141:20735 -> MY.NET.88.163:65535
01/24-12:21:15.832389 [**] High port 65535 udp - possible Red Worm - traffic [**]
66.38.185.141:65535 -> MY.NET.88.163:65535
01/24-12:21:19.725313 [**] High port 65535 udp - possible Red Worm - traffic [**]
66.38.185.141:65535 -> MY.NET.88.163:65535
```

```
01/24-12:21:28.457751 [**] High port 65535 udp - possible Red Worm - traffic [**]
66.38.185.141:65535 -> MY.NET.88.163:65535
01/24-12:23:13.385062 [**] EXPLOIT NTPDX buffer overflow [**] 66.38.185.141:1065 ->
MY.NET.88.163:123
01/24-12:23:13.759511 [**] EXPLOIT NTPDX buffer overflow [**] 66.38.185.141:1065 ->
MY.NET.88.163:123
01/24-12:23:13.953802 [**] EXPLOIT NTPDX buffer overflow [**] 66.38.185.141:1065 ->
MY.NET.88.163:123
01/24-12:23:14.109737 [**] EXPLOIT NTPDX buffer overflow [**] 66.38.185.141:1065 ->
MY.NET.88.163:123
01/24-12:23:14.283312 [**] EXPLOIT NTPDX buffer overflow [**] 66.38.185.141:1065 ->
MY.NET.88.163:123
01/24-12:23:38.647808 [**] High port 65535 udp - possible Red Worm - traffic [**]
66.38.185.141:55551 -> MY.NET.88.163:65535
```

Correlation:
There was a correlation found in Lloyd Webb practical but as quoted from his practical its destination is port 28800 which seems to be for MS gaming.
"This is probably a false positive generated by port 65535 but also with port 28800 which seems to be a port for MS gaming.
Example from log:

```
08/25-22:25:50.486083 [**] High port 65535 udp - possible Red Worm - traffic [**]
216.166.204.167:65535 -> MY.NET.234.66:28800
```

Below is registration information of one of the hosts that is generating the most traffic that is triggering this alerts.

66.38.185.141 has valid reverse DNS of 141.185.38.66.gt-est.net

```
IP Address: 66.38.185.141
HostName: 141.185.38.66.gt-est.net
DShield Profile: Country: CA
Contact E-mail: hostmaster@gt.ca
Total Records against IP:  23
Number of targets:  10
Date Range: 2002-02-09 to 2002-02-19
Ports Attacked (up to 10): Port Attacks

Whois: GT Group Telecom Services Corp. (NETBLK-GROUPTELECOM-BLK-3)
   20 BAY STREET SUITE 700
   TORONTO, ON M5J 2N8
   CA

   Netname: GROUPTELECOM-BLK-3
   Netblock: 66.38.128.0 - 66.38.255.255
   Maintainer: GTGR
```

```
  Coordinator:
     GT Group Telecom Services Corp.  (ZG40-ARIN)  hostmaster@gt.ca
     416-848-2000

  Domain System inverse mapping provided by:

  NS1.CLGRAB.GROUPTELECOM.NET   139.142.2.3
  NS2.TOROON.GROUPTELECOM.NET   209.135.99.3
```

Defensive Recommendation:
First of all there is a tool available to remove any Adore Worm found at Dartmouth University, below is the url
http://www.ists.dartmouth.edu/IRIA/knowledge_base/tools/adorefind.htm
There are also references of fixes available for RedHat flavor of Linux.
http://www.redhat.com/support/alerts/Adore_worm.html
Now in reference to RC-1 Trojan, you see it in listening mode when you would do a netstat –a Any Virus scan tool should be able to scan and delete the Trojan. And also as mentioned earlier make sure the firewall policies have proper rules to block any un-wanted inbound traffic.

For more information on blocking inbound traffic, see……
http://rr.sans.org/firewall/blocking_cisco.php
http://rr.sans.org/firewall/blocking_ipchains.php

EXPLOIT NTPDX buffer overflow:

Description:
NTP, which is time synchronization service on NetBSD as well as many other systems, is vulnerable to a buffer-overflow attack. This Vulnerability can be exploited by executing arbitrary code to the NTP daemon, which usually runs as root.
http://www.linuxsecurity.com/advisories/netbsd_advisory-1255.html

Spp_http_decode: CGI Null Byte attack detected

Description:
This is a part of the http preprocessor that comes with snort. Say if http decoding finds a %00 in any request, it will generate an alert. This may be sign of false positive if MY.NET is using cookies with url encoded binary data. It will also generate alert if the scan traffic is SSL encrypted traffic. More data is needed to analyze this. We may have to log the alerted packet and actually check the packet strings that caused this alert to generate.

Statistics/Analysis:
There were 10 talkers and 21 listeners that generated this alert.

spp_http_decode: CGI Null Byte attack detected	10 sources	21 destinations

All of the source addresses were from the internal network of MY.NET.x.x . Also all of the destination addresses were external addresses. This is again sign for caution as these hosts may have been compromised. We also need to look at packet payload from the logged information as to what traffic is being generated here.

As can also be seen from the traces most of the host that generated this type of alerts also generated "INFO MSN IM Chat data" as well as "ICMP router selection". This is addressed later on in this report.

```
01/23-15:59:36.643039 [**] INFO MSN IM Chat data [**] MY.NET.150.165:2119 ->
64.4.12.164:1863
01/23-15:59:57.785055 [**] INFO MSN IM Chat data [**] MY.NET.150.165:2119 ->
64.4.12.164:1863
01/23-16:00:12.236743 [**] INFO MSN IM Chat data [**] MY.NET.150.165:2119 ->
64.4.12.164:1863
01/23-16:03:36.369340 [**] ICMP Router Selection [**] MY.NET.150.165 -> 224.0.0.2
01/23-16:03:39.454815 [**] ICMP Router Selection [**] MY.NET.150.165 -> 224.0.0.2
01/23-16:03:42.514436 [**] ICMP Router Selection [**] MY.NET.150.165 -> 224.0.0.2
01/23-17:33:23.306852 [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.150.165:1934 -> 205.226.241.199:80
01/23-17:33:23.306852 [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.150.165:1934 -> 205.226.241.199:80
01/23-17:33:23.306852 [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.150.165:1934 -> 205.226.241.199:80
01/23-17:33:23.306852 [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.150.165:1934 -> 205.226.241.199:80
01/23-17:33:23.306852 [**] spp_http_decode: CGI Null Byte attack detected [**]
MY.NET.150.165:1934 -> 205.226.241.199:80
```

Defensive recommendation:
See the defense recommendation discussed for "spp_http_decode: IIS Unicode attack detected" This should also be applicable for the above alert. You can also implement filtering at your border router specially if its Cisco you can implement Network Based Application Recognition (NBAR) access list.

INFO MSN IM Chat data

Description:
This alert basically captures MSN Instant messenger traffic. Below is the rule that can be used with snort to alert on these activities on your internal network. The traffic usually initiates from the internal segment connecting to a Chat server on port 1863

```
alert tcp $HOME_NET any - $EXTERNAL_NET 1863 (msg:"INFO MSN IM Chat data";flags:
A+; content:"|746578742F706C61696E|"; )
```

Correlation:
There was no correlation found for this alert.

Statistics/Analysis:
Most of the traffic was being generated from the internal MY.NET.x.x to a block of 64.4.0.0 network. There were 78 talkers and 75 listeners of this traffic alert per SnortSnarf.

Traces below show several host on MY.NET initiating a TCP session with Hotmail host.

```
Jan 24 09:42:00 MY.NET.150.242:1198 -> 64.4.12.163:1863 SYN ******S*
Jan 24 09:59:38 MY.NET.150.242:1278 -> 64.4.12.161:1863 SYN ******S*
Jan 24 10:11:05 MY.NET.152.158:1151 -> 64.4.13.17:1863 SYN ******S*
Jan 24 10:11:05 MY.NET.152.158:1152 -> 64.4.13.78:1863 SYN ******S*
Jan 24 10:11:45 MY.NET.152.158:1192 -> 64.4.13.78:1863 SYN ******S*
Jan 24 10:12:09 MY.NET.150.242:1323 -> 64.4.12.174:1863 SYN ******S*
Jan 24 10:31:14 MY.NET.152.176:4114 -> 64.4.13.50:1863 SYN ******S*
Jan 24 11:04:20 MY.NET.153.194:1973 -> 64.4.13.32:1863 SYN ******S*
Jan 24 11:04:21 MY.NET.153.194:1980 -> 64.4.13.93:1863 SYN ******S*
Jan 24 11:58:45 MY.NET.152.214:2628 -> 64.4.13.17:1863 SYN ******S*
Jan 24 11:58:45 MY.NET.152.214:2629 -> 64.4.13.84:1863 SYN ******S*
Jan 24 11:58:54 MY.NET.152.214:2660 -> 64.4.12.160:1863 SYN ******S*
Jan 24 14:32:17 MY.NET.153.154:1190 -> 64.4.13.81:1863 SYN ******S*
Jan 24 14:32:28 MY.NET.153.154:1208 -> 64.4.12.155:1863 SYN ******S**
Jan 24 15:05:16 MY.NET.153.168:2550 -> 64.4.13.17:1863 SYN ******S*
Jan 24 15:05:16 MY.NET.153.168:2551 -> 64.4.13.85:1863 SYN ******S*
Jan 24 16:40:15 MY.NET.88.181:2395 -> 64.4.12.184:1863 SYN ******S*
Jan 24 17:03:42 MY.NET.152.247:2457 -> 64.4.13.17:1863 SYN ******S*
Jan 24 17:03:53 MY.NET.152.247:2494 -> 64.4.12.191:1863 SYN ******S*
Jan 24 17:03:56 MY.NET.152.247:2494 -> 64.4.12.191:1863 SYN ******S*
Jan 24 17:07:12 MY.NET.152.168:2473 -> 64.4.12.130:1863 SYN ******S*
Jan 24 17:11:32 MY.NET.153.194:1414 -> 64.4.13.93:1863 SYN ******S*
Jan 24 20:41:19 MY.NET.153.177:1863 -> 66.28.48.201:80 SYN ******S*
Jan 24 21:38:07 MY.NET.150.165:1489 -> 64.4.13.59:1863 SYN ******S*
Jan 24 21:38:15 MY.NET.150.165:1497 -> 64.4.12.182:1863 SYN ******S*
Jan 24 22:49:24 MY.NET.88.181:4436 -> 64.4.12.124:1863 SYN ******S*
Jan 24 22:49:24 MY.NET.88.181:4437 -> 64.4.13.32:1863 SYN ******S*
```

Traces of Hotmail host responding.

01/23-12:14:17.900061 [**] INFO MSN IM Chat data [**] 64.4.12.172:1863 -> MY.NET.152.19:1156
01/23-12:14:28.973829 [**] INFO MSN IM Chat data [**] 64.4.12.172:1863 ->

MY.NET.152.19:1156
01/23-12:18:28.639534 [**] INFO MSN IM Chat data [**] 64.4.12.172:1863 -> MY.NET.152.19:1156
01/23-12:18:39.631871 [**] INFO MSN IM Chat data [**] 64.4.12.172:1863 -> MY.NET.152.19:1156
01/23-12:18:51.658915 [**] INFO MSN IM Chat data [**] 64.4.12.172:1863 -> MY.NET.152.19:1156
01/23-12:19:01.901015 [**] INFO MSN IM Chat data [**] 64.4.12.172:1863 -> MY.NET.152.19:1156

Looking at the registration below you can see that the whole block of 64.4.0.0 belong to Hotmail services.

```
Trying whois -h whois.arin.net 64.4.12.172
MS Hotmail (NETBLK-HOTMAIL)
  1065 La Avenida
  Mountain View, CA 94043
  US

  Netname: HOTMAIL
  Netblock: 64.4.0.0 - 64.4.63.255


Coordinator:
    Myers, Michael  (MM520-ARIN)  icon@HOTMAIL.COM
    650-693-7072

  Domain System inverse mapping provided by:

  NS1.HOTMAIL.COM 216.200.206.140
  NS3.HOTMAIL.COM 209.185.130.68

  Record last updated on 09-Jan-2001.
  Database last updated on  10-Feb-2002 19:55:25 EDT.
```

Defensive Recommendation:
Depending on security policy of the site this is either accepted usage of the Internet connection and therefore the above rule defined can be removed from Snort IDS, Thus eliminating false positives. Or if the policy does not permit this activity deploying outbound filters on the router/Firewall can block this access.

For more information on blocking outbound traffic, see…
http://rr.sans.org/firewall/blocking_cisco.php
http://rr.sans.org/firewall/blocking_ipchains.php

ICMP Router Selection

Description:
There were 130 talkers who generated these 1474 alerts to destination of 224.0.0.2

ICMP Router Selection	130 sources	1 destinations

This would be ICMP Router Discovery, as described in the RFC 1256
http://www.ietf.org/rfc/rfc1256.txt
Router Discovery enables hosts to be attached to a broadcast network and discover IP addresses of neighboring routers. A host that's RFC 1256 compliant needs to be either configured with a default gateway on boot or if the default gateway is down, it actually sends out a Router discovery request using an ICMP message (Type 10, Code 0). This request is sent to the all-routers IP multicast address of 224.0.0.2. Routers that support RFC 1256 residing on the host's network immediately respond with a Router Advertisement, this enables the host to choose the router as its default gateway. This article below goes in depth on ICMP discovery and how Windows machines react.
http://www.microsoft.com/windows2000/techinfo/reskit/en/intwork/inae_ips_oliq.htm

Statistics/Analysis:
Thus the host below are sending router discovery request on multicast address of 224.0.0.2

01/26-19:39:24.801721 [**] ICMP Router Selection [**] MY.NET.153.71 -> 224.0.0.2
01/26-19:39:27.806239 [**] ICMP Router Selection [**] MY.NET.153.71 -> 224.0.0.2
01/26-19:42:49.822551 [**] ICMP Router Selection [**] MY.NET.153.71 -> 224.0.0.2
01/26-19:42:53.878634 [**] ICMP Router Selection [**] MY.NET.153.71 -> 224.0.0.2
01/26-19:42:56.878204 [**] ICMP Router Selection [**] MY.NET.153.71 -> 224.0.0.2

Correlation:
There were no correlations found, as this signature may be new one. This traffic can be classified as inoffensive and can be fine tuned to eliminate false positive.

Defensive Recommendation:
Make sure the source addresses are from your internal network, i.e. MY.NET in our case here. Customize your rule to make sure no external addresses are seen traversing through your default route, specially in case of dual home BGP set up to the Internet. Below is a sample rule to reflect the customization.

alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP Router Selection"; itype: 10; icode: 0;)

SMB Name Wildcard

Description:
Here the host is requesting the NETBIOS name of the destination host. This is a part of Windows file sharing protocol to obtain domain name, user name, host name, etc. This traffic mostly occur from port 137 to port 137
All the traffic that generated this alert is from MY.NET to MY.NET, which is normal traffic for NETBIOS Name resolution service on port 137; any attacker can use this reconnaissance information for future attack.

Statistics/Analysis:

Top 5 hosts triggering this alert.

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
MY.NET.150.209	144	246	3	5
MY.NET.150.127	85	152	1	3
MY.NET.150.28	70	115	2	2
MY.NET.5.96	58	61	2	3
MY.NET.150.77	40	66	3	3

Top 5 destination hosts receiving this attack signature packet

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
MY.NET.5.4	448	750	15	15
MY.NET.5.96	58	2243	2	32
MY.NET.5.3	45	187	6	8
MY.NET.98.151	33	33	1	1
MY.NET.5.35	31	50	6	6

```
01/23-13:19:31.204245 [**] SMB Name Wildcard [**] MY.NET.150.209:137 ->
MY.NET.5.35:137
01/23-13:19:32.706225 [**] SMB Name Wildcard [**] MY.NET.150.209:137 ->
MY.NET.5.35:137
01/23-13:19:34.209120 [**] SMB Name Wildcard [**] MY.NET.150.209:137 ->
MY.NET.5.35:137
01/23-13:19:54.540323 [**] SMB Name Wildcard [**] MY.NET.150.209:137 ->
MY.NET.5.35:137
```

Correlation:
A great SANS white paper references this alert that was on the rise around the time when the white paper was drafted.
Reference: http://www.sans.org/newlook/resources/IDFAQ/port_137.htm

Defensive recommendation:
Make sure file sharing is not enabled on all machines. Use of personal firewall can also assist in blocking port 135 through 139. Also filter and block port 135 through 139 on your border router going in both directions (inbound as well as outbound)

There are numerous CERT advisories and also CVE references which talks about related vulnerabilities.
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=2000-0673
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=2000-0347
http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=1999-0288

According to CERT Advisories….
Denial of Service Attack in NetBIOS Services
Overview
The NetBIOS Name Service (NBNS) provides a means for hostname and address mapping on a NetBIOS-aware network. The NetBIOS over TCP/IP protocols (including NBNS) are described in the Internet Engineering Task Force (IETF) Request for Comments RFC1001 and RFC1002. These protocols do not specify a method for authenticating communications, and as such, machines running NetBIOS services are vulnerable to spoofing attacks.

Refer to Vulnerability Note VU#32650
http://www.kb.cert.org/vuls/id/32650

Other notable alerts not in the top ten list:

ICMP Echo Request CyberKit 2.2 Windows

Description:
CyberKit is a network utility for Windows machines. V2.2 is a 32-bit utility. It combines all the following utility into a smart swiss army type solution… Ping, TraceRoute, WhoIs, Finger, Quote of the Day, Name Server LookUp, Time Synchronizer, NetScanner, DBScanner, Check for New Mail and Keep Alive in one easy to use utility.

Statistics/Analysis:

ICMP Echo Request CyberKit 2.2 Windows	3 sources	4 destinations

There were 3 talkers of this tool trying to probe 4 destinations

Source triggering this attack

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
MY.NET.150.49	1463	1582	3	10
MY.NET.150.232	4	316	2	20
MY.NET.150.145	3	7	3	5

Destinations receiving this attack

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
204.71.200.33	736	736	3	3
204.71.200.34	732	732	3	3
216.136.175.132	1	1	1	1
216.136.130.46	1	1	1	1

```
01/23-20:27:49.221718 [**] ICMP Echo Request CyberKit 2.2 Windows [**] MY.NET.150.49 -> 204.71.200.34
01/23-20:27:54.764906 [**] ICMP Echo Request CyberKit 2.2 Windows [**] MY.NET.150.49 -> 204.71.200.33
01/23-20:27:58.264515 [**] ICMP Echo Request CyberKit 2.2 Windows [**] MY.NET.150.49 -> 204.71.200.34
01/23-20:28:03.728222 [**] ICMP Echo Request CyberKit 2.2 Windows [**] MY.NET.150.49 -> 204.71.200.33
01/23-20:28:07.231643 [**] ICMP Echo Request CyberKit 2.2 Windows [**] MY.NET.150.49 -> 204.71.200.34
```

From the traces one can ascertain that this tool uses ICMP echo request data payload for communication. Thus this makes it evasive as the Firewall policy might allow ICMP control message to go in and out.

Per CyberKit web page it requires support of sockets v2 to run this utility. The newer release v3.0 of Cyberkit will include support for ICMP.dll. This will eliminate the need for sockets 2

Reference:
http://www.cyberkit.net/index.html

I could not find any data to analyze this activity in the scan file. We'll have to enable logging of alerted packet to examine the packet content.

Correlation:
I believe this is a brand new alert/signature. I did not find any correlations at the time this paper was written.

Defensive *Recommendation:*
For now to begin with, Log the alerted packet and examine the packet content.

ICMP Fragment Reassembly Time Exceeded

Description:
Occurs when the fragments are not received within the time-out value defined by the Operating System. The host does the reassembly of the datagram. This can be used for DoS on a system, when a system is overwhelmed with many connections which having missing fragments. An intruder to by-pass the router or Intrusion detection system often uses this technique.

Statistics/Analysis:

ICMP Fragment Reassembly Time Exceeded	17 sources	26 destinations

Top 5 sources triggering this attack signature

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
MY.NET.153.159	334	342	1	3
MY.NET.153.171	281	1416	5	60
MY.NET.153.197	164	1162	1	34
MY.NET.153.185	149	1474	5	46
MY.NET.153.45	86	120	3	5

Top 5 Destinations receiving this attack signature

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
211.234.110.20	498	498	2	2
211.174.63.106	119	119	1	1
208.172.128.163	84	84	1	1
211.174.63.108	59	59	1	1
211.171.202.142	58	58	1	1

In our logs we have the internal network MY.NET hosts sending ICMP Fragment Reassembly Time Exceeded error messages to external hosts. Which indicates that some of the fragment of a given stream of data may have been lost in the transit. This is a great way for reconnaissance. Since the first fragment of a datagram contains the protocol header such as TCP, UDP or ICMP. Only this packet will be denied to enter if guarded by a packet-filtering device. On receiving the subsequent fragments of the same datagram, the packet filtering would have no clue what to do, as it does not keep a state of the packet header and it will let the subsequent fragment pass through.

Correlation:
No correlations were found. But there are a number of documentation published, which talks about using fragmentation to evade an IDS system.

As quoted in Robert Graham White paper on IDS

"Fragmentation is the ability to break up a single IP packet into multiple smaller packets. The receiving TCP/IP stack then reassembles the data back again before forwarding the data back up to the application. Most intrusion detection systems do not have the ability to reassemble IP packets."
http://www.robertgraham.com/pubs/network-intrusion-detection.html#9.4

Defensive Recommendation:
A Stateful Firewall must be deployed. This firewall would keep the state of the packet header along with the fragment ID. In some cases (certain vendor specific firewall) all the defragmented packets are resembled and then given to the destination host. If any data offset errors are found, the whole datagram is discarded.

Watchlist 000220 IL-ISDNNET-990517

Top 5 hosts triggering this attack

Source	# Alerts (sig)	# Alerts (total)	# Dsts (sig)	# Dsts (total)
212.179.35.118	449	449	3	3
212.179.27.176	115	115	1	1
212.179.27.6	54	54	3	3
212.179.127.37	19	19	1	1
212.179.76.28	16	16	1	1

Top 5 destination hosts receiving this attack signature

Destinations	# Alerts (sig)	# Alerts (total)	# Srcs (sig)	# Srcs (total)
MY.NET.153.162	575	586	4	8
MY.NET.150.133	72	277	11	86
MY.NET.88.162	39	138	2	37
MY.NET.150.145	22	36	2	5
MY.NET.150.220	14	19	4	8

There were 741 alert logged for this attack.

These scans have been seen coming from the 212.179.0.0 address block. See below for registration information.

```
inetnum:     212.179.0.0 - 212.179.255.255
netname:     IL-ISDNNET-990517
descr:       PROVIDER
country:     IL
admin-c:     NP469-RIPE
tech-c:      TP1233-RIPE
tech-c:      ZV140-RIPE
tech-c:      ES4966-RIPE
status:      ALLOCATED PA
mnt-by:      RIPE-NCC-HM-MNT
changed:     hostmaster@ripe.net 19990517
changed:     hostmaster@ripe.net 20000406
changed:     hostmaster@ripe.net 20010402
source:      RIPE
route:       212.179.0.0/17
descr:       ISDN Net Ltd.
origin:      AS8551
notify:      hostmaster@isdn.net.il
mnt-by:      AS8551-MNT
changed:     hostmaster@isdn.net.il 19990610
source:      RIPE
person:      Nati Pinko
address:     Bezeq International
address:     40 Hashacham St.
address:     Petach Tikvah  Israel
phone:       +972 3 9257761
e-mail:      hostmaster@isdn.net.il
nic-hdl:     NP469-RIPE
changed:     registrar@ns.il 19990902
source:      RIPE
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
person:      Zehavit Vigder
address:     bezeq-international
address:     40 hashacham
address:     petach tikva 49170 Israel
```

```
phone:     +972 52 770145
fax-no:    +972 9 8940763
e-mail:    hostmaster@bezeqint.net
nic-hdl:   ZV140-RIPE
changed:    zehavitv@bezeqint.net 20000528
source:    RIPE
person:    Eran Shchori
address:    BEZEQ INTERNATIONAL
address:    40 Hashacham Street
address:    Petach-Tikva 49170 Israel
phone:     +972 3 9257710
fax-no:    +972 3 9257726
e-mail:    hostmaster@bezeqint.net
nic-hdl:   ES4966-RIPE
changed:    registrar@ns.il 20000309
source:    RIPE
```

Description:
Watchlists are lists of suspected address ranges, that have been known to carry out malicious activities. Watchlist 000220 refers to the list, which is comprised of address block 212.179.0.0

Correlation:
There are several correlations for this alerts, in fact there are several Watch list similar to watchlist 000220.

In Jeff Holland practical the same watchlist 000220-IL—ISDNNET-990517 shows different registration information. See below
http://www.giac.org/practical/Jeff_Holland_GCIA.doc

```
Server used for this query: [ whois.ripe.net ]
inetnum:    212.79.0.0 - 212.79.63.255
netname:    DE-POP-981102
descr:     POP Point of Presence
descr:     PROVIDER
country:    DE
role:     Hostmaster POP
address:    POP Point of Presence GmbH
address:    Wendenstrasse 375-377
address:    D-20537 Hamburg
address:    Germany

inetnum:    212.79.64.0 - 212.79.95.255
netname:    BE-ASSURNET-990202
descr:     PROVIDER
```

```
country:    DE
role:       Assurnet IP OPERATOR
address:    Assurnet
address:    150 chaussee de La Hulpe
address:    1170 Bruxelles
address:    Belgium
```

Note: Current registration shows that, this same address space apparently got reassigned to the ISP in Israel.

In Robert Turner practical he also refers to a watchlist 000222 and its comprises of address block of 159.226.0.0/16
In fact we see some of the alerts coming from the same network block 159.226.0.0/16, i.e. Watchlist 000222 NET-NCFC

Below is the registration info of this network block (159.226.0.0/16). It is not alarming as there were only 2 alerts generated.

```
01/27-21:35:13.749915 [**] NMAP TCP ping! [**] 159.226.208.40:80 ->
MY.NET.150.133:1214
```

```
The Computer Network Center Chinese Academy of Sciences (NET-NCFC)
  P.O. Box 2704-10,
  Institute of Computing Technology Chinese Academy of Sciences
  Beijing 100080, China
  CN

  Netname: NCFC
  Netblock: 159.226.0.0 - 159.226.255.255

  Coordinator:
    Qian, Haulin  (QH3-ARIN)  hlqian@NS.CNC.AC.CN
    +86 1 2569960

  Domain System inverse mapping provided by:

  NS.CNC.AC.CN                  159.226.1.1
  GINGKO.ICT.AC.CN              159.226.40.1

  Record last updated on 25-Jul-1994.
  Database last updated on  12-Feb-2002 19:56:22 EDT.
```

http://www.giac.org/practical/Robert_Turner_GCIA.doc

Now back to Watchlist 000220-IL-ISDNNET-990517

Statistics/Analysis:
Below is a list of all the ports that were being targeted.

1383	1381	1427	2563	4616
1214	1394	1491	2579	6346
1315	1397	1499	2582	
1319	1411	1508	2597	
1320	1423	1514	2598	
1321	1424	1526	2599	

Port Database tool on http://www.snort.org is a great tool to find unknown ports information.

The most heavily utilized port found was 1214 KAZAA service.

The below SANS url also list all odd ports utilized by different Trojans.
http://www.sans.org/newlook/resources/IDFAQ/oddports.htm

```
01/24-18:54:32.575930 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.76.28:3846 -
> MY.NET.88.162:1214
01/24-18:54:36.809247 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.76.28:3846 -
> MY.NET.88.162:1214
01/24-18:54:41.867741 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.76.28:3846 -
> MY.NET.88.162:1214
01/24-18:54:41.886209 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.76.28:3846 -
> MY.NET.88.162:1214
01/24-18:54:44.913432 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.76.28:3846 -
> MY.NET.88.162:1214
```

Defensive recommendation:
Deploy a Stateful Firewall if not already deployed. Block all unnecessary inbound as well as outbound traffic by deploying ACL on the routers. Make sure all systems have anti-virus software installed with its latest signature files. Also determine that all the hosts have the latest security patch deployed.

It is also highly recommended to block this IP address block at your Internet border router by deploying Access List.

For more information on blocking traffic, see…
http://rr.sans.org/firewall/blocking_cisco.php
http://rr.sans.org/firewall/blocking_ipchains.php

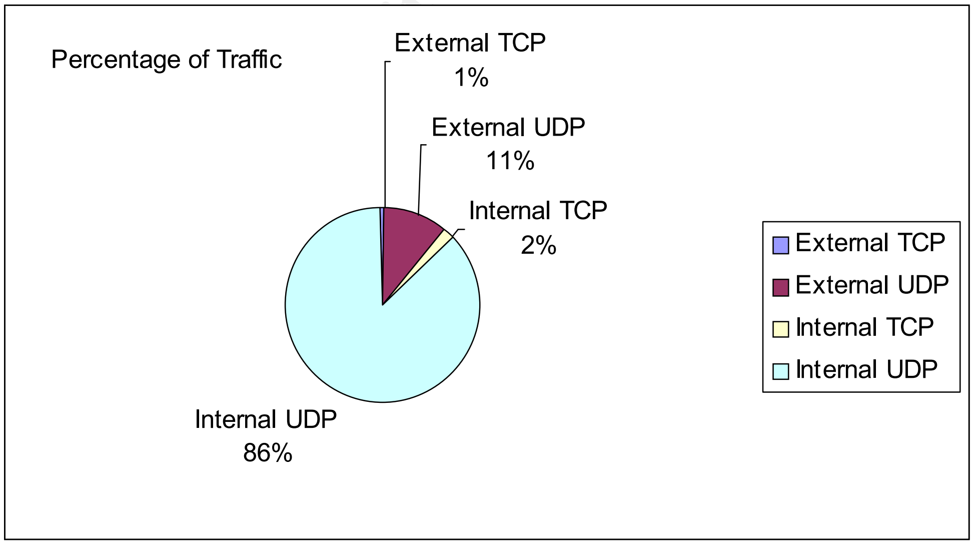
Port Scan Analysis:

"Top ten talkers" list in terms of scan logs.

Source IP	Total # alerts
MY.NET.60.43	390866
MY.NET.6.49	68098
MY.NET.6.45	49592
MY.NET.6.50	45581
MY.NET.6.48	40692
MY.NET.6.52	36043
MY.NET.6.60	26932
66.38.185.141	23798
MY.NET.153.171	23493
MY.NET.6.53	21556

As you can see from the above list majority of the port scan activity is being triggered from the internal network, with the exception of one source 66.38.185.141

The below graphs shows the traffic analysis based on traffic directions, As you can see the Internal host have generated 86% of UDP traffic going outbound. The were 998236 port scan entries found in the five days logs of which the top ten host conducted majority of the scans, i.e. 726651 scans approximately 73% of overall scans.



Host: MY.NET.60.43 has been seen generating 390866 UDP packets. Now when you look at the logs there were 348158 entries from source port UDP/123, which is typically used by NTP daemon. This can very well be MY.NET.60.43 host responding to the NTP exploits explained earlier. This itself constitutes 89% of the traffic generated by this host in particular.
There were other 37736 instances of UDP source port 7000 responding which seems to appear like some sort of Gnutella type traffic or AFS system (see below reference)

"An AFS Cell is a collection of servers grouped together administratively and presenting a single, cohesive file system,"

http://www.angelfire.com/hi/plutonic/afs-faq.html#sub1.03

```
Jan 23 00:02:18 MY.NET.60.43:7000 -> MY.NET.149.96:7001 UDP
Jan 23 00:03:20 MY.NET.60.43:7000 -> MY.NET.153.187:7001 UDP
Jan 23 00:03:36 MY.NET.60.43:7000 -> MY.NET.152.139:7001 UDP
Jan 23 00:03:36 MY.NET.152.139:7001 -> MY.NET.60.43:7000 UDP
Jan 23 00:03:47 MY.NET.60.43:7000 -> MY.NET.5.107:7001 UDP
Jan 23 00:03:47 MY.NET.5.107:7001 -> MY.NET.60.43:7000 UDP
```

Of course this host was also a target of either Queso or NMAP OS fingerprinting there was 3885 instances of traffic seen to port 0. But as we see the traffic even to port 0 is all internal; see traces below.

```
Jan 23 09:59:10 MY.NET.60.43:0 -> MY.NET.153.172:0 UDP
Jan 23 09:59:14 MY.NET.60.43:0 -> MY.NET.153.172:0 UDP
Jan 23 10:09:05 MY.NET.60.43:0 -> MY.NET.153.202:0 UDP
Jan 23 10:09:09 MY.NET.60.43:0 -> MY.NET.153.202:0 UDP
Jan 23 10:10:41 MY.NET.60.43:0 -> MY.NET.153.202:0 UDP
Jan 23 10:10:45 MY.NET.60.43:0 -> MY.NET.153.202:0 UDP
Jan 23 10:10:51 MY.NET.60.43:0 -> MY.NET.153.202:0 UDP
Jan 23 10:10:56 MY.NET.60.43:0 -> MY.NET.153.202:0 UDP
Jan 23 10:10:56 MY.NET.60.43:0 -> MY.NET.153.202:1 UDP
Jan 23 10:11:01 MY.NET.60.43:0 -> MY.NET.153.202:0 UDP
```

Host: MY.NET.6.49; again there was a lot of traffic generating on port 0 i.e. Queso/NMAP OS fingerprinting as well as 7935 instances on 7000 and 1801 on 7001. This could well be assumed to be AFS or Gnutella activity. Now this was seen as internal, i.e. from source/destination host, which were within the MY.NET.x.x network.

```
Jan 24 14:46:26 MY.NET.6.49:7000 -> MY.NET.88.148:7001 UDP
Jan 24 14:46:26 MY.NET.6.49:7000 -> MY.NET.153.169:7001 UDP
Jan 24 14:46:26 MY.NET.6.49:7000 -> MY.NET.152.22:7001 UDP
Jan 24 14:46:26 MY.NET.6.49:7000 -> MY.NET.153.170:7001 UDP
Jan 24 14:46:26 MY.NET.6.49:7000 -> MY.NET.153.172:7001 UDP
Jan 24 14:46:26 MY.NET.6.49:7000 -> MY.NET.153.153:7001 UDP
Jan 24 14:46:26 MY.NET.6.49:7000 -> MY.NET.153.195:7001 UDP
Jan 24 14:46:30 MY.NET.6.49:7000 -> MY.NET.152.16:7001 UDP
Jan 24 14:46:35 MY.NET.6.49:7000 -> MY.NET.152.16:7001 UDP
```

Host: MY.NET.6.45; here we see extensive traffic on port 7001 as destination port coming from internal host on MY.NET.x.x. And also traffic sourcing on the same 7001 UDP port. Altogether I saw 35063 inbound/outbound UDP packets and also some port 0 traffic inbound 4336 instances and 4439 instances of outbound traffic. There were 8415 instances of traffic initiating from UDP port 123 to internal network, as seen from the traces below

```
Jan 27 05:38:06 MY.NET.6.45:123 -> MY.NET.153.144:4982 UDP
Jan 27 06:34:07 MY.NET.6.45:123 -> MY.NET.153.144:1063 UDP
Jan 27 07:05:07 MY.NET.6.45:123 -> MY.NET.153.144:1094 UDP
Jan 27 07:10:07 MY.NET.6.45:123 -> MY.NET.153.144:1099 UDP
Jan 27 07:15:07 MY.NET.6.45:123 -> MY.NET.153.144:1104 UDP
Jan 27 07:20:07 MY.NET.6.45:123 -> MY.NET.153.144:1109 UDP
Jan 27 07:25:07 MY.NET.6.45:123 -> MY.NET.153.144:1114 UDP
Jan 27 07:30:07 MY.NET.6.45:123 -> MY.NET.153.144:1119 UDP
Jan 27 07:35:07 MY.NET.6.45:123 -> MY.NET.153.144:1124 UDP
Jan 27 07:40:07 MY.NET.6.45:123 -> MY.NET.153.144:1129 UDP
Jan 27 08:16:07 MY.NET.6.45:123 -> MY.NET.153.144:1165 UDP
Jan 27 12:23:30 MY.NET.6.45:123 -> MY.NET.153.180:2445 UDP
Jan 27 12:55:10 MY.NET.6.45:123 -> MY.NET.153.144:1444 UDP
Jan 27 12:59:10 MY.NET.6.45:123 -> MY.NET.153.144:1448 UDP
Jan 27 13:04:30 MY.NET.6.45:123 -> MY.NET.153.180:2486 UDP
Jan 27 13:05:30 MY.NET.6.45:123 -> MY.NET.153.180:2487 UDP
Jan 27 13:21:10 MY.NET.6.45:123 -> MY.NET.153.144:1470 UDP
Jan 27 13:36:10 MY.NET.6.45:123 -> MY.NET.153.144:1485 UDP
Jan 27 13:46:10 MY.NET.6.45:123 -> MY.NET.153.144:1495 UDP
Jan 27 13:52:10 MY.NET.6.45:123 -> MY.NET.153.144:1501 UDP
Jan 27 13:52:30 MY.NET.6.45:123 -> MY.NET.153.180:2534 UDP
Jan 27 13:53:10 MY.NET.6.45:123 -> MY.NET.153.144:1502 UDP
Jan 27 13:57:10 MY.NET.6.45:123 -> MY.NET.153.144:1506 UDP
Jan 27 14:09:10 MY.NET.6.45:123 -> MY.NET.153.144:1518 UDP
```

Host: MY.NET.6.50; once again a lot of activity is seen on UDP port 0, UDP port 7000 and UDP port 0, as source and destination going to the same internal network.

Host: MY.NET.6.48; Similar activity as above has been seen on this host too, i.e. UDP port 7000, port 7001, & port 0.

Host: MY.NET.6.52; Similar activity has been found as above.

Host: MY.NET.6.60; Same as above.

Host: 66.38.185.141; we see over here similar activity but its been sourced from an external device targeted to MY.NET.x.x network, i.e. UDP port 7000/7001 & 0 and It's coming inbound to our internal network.

Host: MY.NET.153.171; This host is seen receiving a lot UDP packets to port 6112 about 16686 which is about 71% of the total traffic generated for this host. Now this port is alarming as CDE sub process control dtspcd is usually seen using this port. There are a number of exploits for this service. In fact there a several advisories on it see below references.

http://online.securityfocus.com/advisories/3651
http://project.honeynet.org/scans/dtspcd/dtspcd.txt
http://www.iss.net/security_center/static/7396.php
http://www.kb.cert.org/vuls/id/172583

Below description is from cert.'s advisories web page (referenced above)
<snip>
Internet Security Systems (ISS) X-Force has reported a remotely exploitable buffer overflow in the Common Desktop Environment (CDE) Subprocess Control Service (`dtspcd`). CDE is an integrated graphical user interface that runs on Unix and Linux operating systems. `dtspcd` is a network daemon that accepts requests from clients to execute commands and launch applications remotely. On systems running CDE, `dtspcd` is spawned by the Internet services daemon (typically `inetd` or `xinetd`) in response to a CDE client request. `dtspcd` is typically configured to run on port 6112/tcp with root privileges. `dtspcd` makes a function call to a shared library, `libDTSvc.so.1`, that contains a buffer overflow condition in the client connection routine. The buffer overflow can be exploited by a specially crafted CDE client request. Although the buffer overflow occurs in a shared library, the CERT/CC is not aware of any other CDE applications that use the vulnerable function.
<snip>

Host: MY.NET.6.53; Once again it's the same port as seen above on all internal systems i.e. UDP port 0, 7000 & 7001 used as source/destination within MY.NET.x.x network.

Comments:
A lot of the traffic seen from the scan logs appears to be internal with an exception of source host 66.38.185.141, which is triggering traffic to MY.Net.x.x. We also see the CDE traffic directed towards host MY.NET.153.171 from the external sources. This host also appears in the top destinations list for targeted MISC Large UDP packet alert, in Snortsnarf analysis. As we see that most of the traffic is within the internal network snort.conf $HOME_NET variable needs to be configured correctly to eliminate at least 75% of all the false positives. A logging of payload should be done to examine the packet to see if any of the above CDE buffer overflow vulnerability is exploited.

OOS Data:

These packet were Out-of-Specification with the RFC compliancy. Snort detected these packets and as mentioned they do no meet IP specification. The reason could be that they were crafted to cause undesired behavior on the target host. There could be other reason for these packets to be Out-of Spec, such as get corrupted via a router/network device on the way; the source machine corrupted it when the packets were traversing the IP stack, etc. It should also be noted that specifications do change from time to time, such as what used to be reserved bits in the IP and TCP header and they are now being used for ECN – Explicit Congestion Notification.

OOS Analysis:

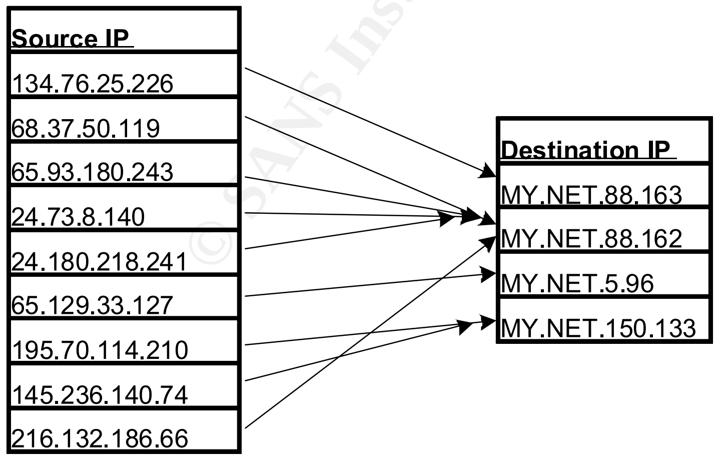
96% of the packets captured had both the reserved bit flagged with an exception of one packet that appeared with normal TCP flag, but as you look at the packet it appears to be crafted as you see the hex value of IP Version and the IP header length is set to 00 (see below)

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=
+
01/23-13:22:13.297592 24.180.218.241:0 -> MY.NET.88.162:2173
TCP TTL:112 TOS:0x0 ID:21326  DF
*1SF**A* Seq: 0x4BE08BB   Ack: 0x91611D1B   Win: 0x5010
00 00 08 7D 04 BE 08 BB 91 61 1D 1B 0B 93 50 10  ...}.....a....P.
22 38 EE F9 00 00 00 00 00 00                    "8........

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

The scan logs also shows 71% of the traffic going to one particular host MY.NET.88.162 probing port 1214 that is typically used for Kazaa. There were few other destination ports probed which I could not find any information as they may not be well known ports. But one of the source host (same src & dest as the above trace) that appeared constantly probing the above destination host was also trying to connect to destination port 1796 which is used by Vocaltec Server Administration program.

There were altogether 9-source host found in the analysis of OOS logs generating traffic towards 4 internal hosts, which are depicted below.



Below is all the traffic that was collected by the OOS logs in a tabular format;

Source IP	Port	TCP Flag	Dest IP	Port
68.37.50.119	1371	21*FR***	MY.NET.88.162	1214
65.93.180.243	3476	**SF**A*	MY.NET.88.162	1214
65.129.33.127	18245	2*SFRP*U	MY.NET.5.96	21536
65.129.33.127	18245	2*SFRP*U	MY.NET.5.96	21536
24.73.8.140	3210	2*SFR*AU	MY.NET.88.162	1214
24.73.8.140	0	2*SFR**U	MY.NET.88.162	3210
24.180.218.241	1449	21SFR***	MY.NET.88.162	1214
24.180.218.241	1522	21S*****	MY.NET.88.162	1214
24.180.218.241	1796	21**RPAU	MY.NET.88.162	1214
24.180.218.241	1796	21**RP**	MY.NET.88.162	1214
24.180.218.241	2173	21SFRP*U	MY.NET.88.162	1214
24.180.218.241	2342	21**R*AU	MY.NET.88.162	1214
24.180.218.241	2342	21*FR**U	MY.NET.88.162	1214
24.180.218.241	13	21**RP*U	MY.NET.88.162	1661
24.180.218.241	166	21**RPAU	MY.NET.88.162	1796
24.180.218.241	0	*1SF**A*	MY.NET.88.162	2173
24.180.218.241	0	21*FPAU	MY.NET.88.162	2259
216.132.186.66	2828	21S*****	MY.NET.88.162	1214
195.70.114.210	2406	21S*****	MY.NET.150.133	1214
145.236.140.74	1214	21S*****	MY.NET.150.133	1214
145.236.140.74	1251	21S*****	MY.NET.150.133	1214
145.236.140.74	1361	21S*****	MY.NET.150.133	1214
145.236.140.74	1417	21S*****	MY.NET.150.133	1214
134.76.25.226	3099	21**RP*U	MY.NET.88.163	41004

Top Talkers Analysis:

Source IP	Occurrences
24.180.218.241	11
145.236.140.74	4

Besides the post scans, there were also 11 alerts generated from the alert logs from 24.180.218.241 source probing MY.NET.88.162. They were “Null Scan” and “Queso fingerprinting”

Queso is a program used to probe any remote system with certain type of TCP sequence. The response packet from the system is typically used to determine the OS that’s running on this remote system. Another tool used for OS Fingerprinting is NMAP, which must have been used in an attempt to determine the host Operating system. Both of these activities may have caused these alerts.

```
01/23-11:18:40.978255  [**] Null scan! [**] 24.180.218.241:1214 -
> MY.NET.88.162:4242
01/23-11:18:40.978255  [**] Null scan! [**] 24.180.218.241:1214 -
> MY.NET.88.162:4242
01/23-11:32:04.400460  [**] spp_portscan: portscan status from
24.180.218.241: 1 connections across 1 hosts: TCP(1), UDP(0)
STEALTH [**]
01/23-11:32:05.748749  [**] spp_portscan: End of portscan from
24.180.218.241: TOTAL time(0s) hosts(1) TCP(1) UDP(0) STEALTH
[**]
01/23-11:32:32.445863  [**] spp_portscan: PORTSCAN DETECTED from
24.180.218.241 (STEALTH) [**]
01/23-11:32:33.438389  [**] spp_portscan: portscan status from
24.180.218.241: 1 connections across 1 hosts: TCP(1), UDP(0)
STEALTH [**]
01/23-11:32:34.679000  [**] spp_portscan: End of portscan from
24.180.218.241: TOTAL time(0s) hosts(1) TCP(1) UDP(0) STEALTH
[**]
01/23-11:33:03.676546  [**] spp_portscan: PORTSCAN DETECTED from
24.180.218.241 (STEALTH) [**]
01/23-11:21:33.582178  [**] Queso fingerprint [**]
24.180.218.241:1522 -> MY.NET.88.162:1214
01/23-11:33:04.607911  [**] spp_portscan: portscan status from
24.180.218.241: 1 connections across 1 hosts: TCP(1), UDP(0)
STEALTH [**]
01/23-11:33:05.874825  [**] spp_portscan: End of portscan from
24.180.218.241: TOTAL time(0s) hosts(1) TCP(1) UDP(0) STEALTH
[**]
01/23-11:33:44.581512  [**] spp_portscan: PORTSCAN DETECTED from
24.180.218.241 (STEALTH) [**]
```

The same scans activities from the same source was also seen captured in the scan logs; see traces below.

```
Jan 23 11:03:15 24.180.218.241:1436 -> MY.NET.88.162:1214 NOACK 1*U*PR**
RESERVEDBITS
Jan 23 11:04:33 24.180.218.241:1449 -> MY.NET.88.162:1214 NOACK 12***RSF
RESERVEDBITS
Jan 23 11:18:40 24.180.218.241:1214 -> MY.NET.88.162:4242 NULL ********
Jan 23 11:20:07 24.180.218.241:1522 -> MY.NET.88.162:1214 NOACK *2U*P*S*
RESERVEDBITS
Jan 23 11:21:33 24.180.218.241:1522 -> MY.NET.88.162:1214 SYN 12****S*
RESERVEDBITS
Jan 23 11:23:35 24.180.218.241:1522 -> MY.NET.88.162:1214 NOACK **U*P*S*
Jan 23 11:27:05 24.180.218.241:1522 -> MY.NET.88.162:1214 NULL 12******
RESERVEDBITS
Jan 23 11:36:50 24.180.218.241:13 -> MY.NET.88.162:1661 NOACK 12U*PR**
RESERVEDBITS
Jan 23 11:40:57 24.180.218.241:1772 -> MY.NET.88.162:1214 NULL ********
Jan 23 11:44:00 24.180.218.241:1796 -> MY.NET.88.162:1214 UNKNOWN *2*A****
RESERVEDBITS
Jan 23 11:44:16 24.180.218.241:1796 -> MY.NET.88.162:1214 INVALIDACK **UAPR**
Jan 23 11:45:03 24.180.218.241:166 -> MY.NET.88.162:1796 INVALIDACK 12UAPR**
RESERVEDBITS
Jan 23 11:45:10 24.180.218.241:1796 -> MY.NET.88.162:1214 INVALIDACK 12UAPR**
RESERVEDBITS
Jan 23 11:45:17 24.180.218.241:1796 -> MY.NET.88.162:1214 NOACK 12**PR**
RESERVEDBITS
Jan 23 11:47:45 24.180.218.241:17 -> MY.NET.88.162:1796 INVALIDACK *2UA*R**
RESERVEDBITS
Jan 23 11:49:03 24.180.218.241:1796 -> MY.NET.88.162:1214 INVALIDACK 1*UA*R**
RESERVEDBITS
Jan 23 11:49:34 24.180.218.241:1796 -> MY.NET.88.162:1214 VECNA **U*****
Jan 23 11:51:32 24.180.218.241:1876 -> MY.NET.88.162:1214 UNKNOWN 12UAP***
RESERVEDBITS
Jan 23 11:52:45 24.180.218.241:239 -> MY.NET.88.162:1876 UNKNOWN 12UAP***
RESERVEDBITS
```

Once again 4 alerts were seen from the second source 145.236.140.74 from the above table.
This is seen probing host MY.NET.150.133
It generated 4 occurrences of Queso fingerprinting (explained earlier) along with spp_portscan activity.

```
01/26-12:46:04.996605  [**] spp_portscan: PORTSCAN DETECTED from
145.236.140.74 (STEALTH) [**]
01/26-12:31:22.573004  [**] Queso fingerprint [**]
145.236.140.74:1251 -> MY.NET.150.133:1214
01/26-12:46:05.233047  [**] spp_portscan: portscan status from
145.236.140.74: 1 connections across 1 hosts: TCP(1), UDP(0)
STEALTH [**]
01/26-12:46:05.485979  [**] spp_portscan: End of portscan from
145.236.140.74: TOTAL time(0s) hosts(1) TCP(1) UDP(0) STEALTH
[**]
01/26-12:46:26.884584  [**] spp_portscan: PORTSCAN DETECTED from
145.236.140.74 (STEALTH) [**]
01/26-12:39:37.342370  [**] Queso fingerprint [**]
145.236.140.74:1361 -> MY.NET.150.133:1214
01/26-12:46:27.024417  [**] spp_portscan: portscan status from
145.236.140.74: 1 connections across 1 hosts: TCP(1), UDP(0)
STEALTH [**]
```

```
01/26-12:46:27.183379  [**] spp_portscan: End of portscan from
145.236.140.74: TOTAL time(0s) hosts(1) TCP(1) UDP(0) STEALTH
[**]
01/26-12:46:37.066373  [**] spp_portscan: PORTSCAN DETECTED from
145.236.140.74 (STEALTH) [**]
01/26-12:43:34.640037  [**] Queso fingerprint [**]
145.236.140.74:1417 -> MY.NET.150.133:1214
```

Below is the trace from scans logs correlating the above alert traces and activity found in OOS logs.

```
Jan 26 12:28:24 145.236.140.74:1214 -> MY.NET.150.133:1214 SYN 12****S*
RESERVEDBITS
Jan 26 12:31:22 145.236.140.74:1251 -> MY.NET.150.133:1214 SYN 12****S*
RESERVEDBITS
Jan 26 12:39:37 145.236.140.74:1361 -> MY.NET.150.133:1214 SYN 12****S*
RESERVEDBITS
Jan 26 12:43:34 145.236.140.74:1417 -> MY.NET.150.133:1214 SYN 12****S*
RESERVEDBITS
```

Defensive Recommendation/Summary:

It is alarming that most all of the top ten source IP addresses are MY.NET addresses. This Alert logs analysis as mentioned earlier contained about 51% of the initial LPD service and SNMP traffic. This was classified as false positive. Also from the scans logs it appears that there is some case of Gnutella and or AFS file systems in use internally within MY.NET.x.x network. AFS file systems uses UDP port in the range of 7000 to 7009. Of course this should be verified by examining the payload data. To eliminate all this false positive make sure snort is configured correctly, i.e. HOME_NET variable need to be properly defined in snort.conf file. Also particularly make sure the host that we saw being probed on UDP port 6112 i.e. CDE dtspcd is not compromised by the CDE buffer overflow exploit. Make you have the necessary patches deployed on this system. We also found instances of traffic generated causing IIS Unicode attack alerts, which were targeted to internal MY.NET.x.x network. Make sure these hosts are not compromised. Apply the proper security patches available to alleviate any vulnerability issues on those hosts. There were also instances of internal host sending "spp_http_decode: CGI Null byte attack" traffic to external hosts and generating this alert. This is sure sign of these hosts being compromised. Make sure these hosts are properly scrutinized. Also appropriate security patches are applied.

Please address these issues ASAP. All system should be checked against their OS security checklist to make sure in the end none of the systems are left vulnerable. There are several checklists available on Cert.'s website as well as some great information on checklist for various systems can be found at the below references.

http://www.labmice.net/articles/securingwin2000.htm
http://www.cert.org/tech_tips/usc20_full.html
http://www.cymru.com/~robt/Docs/Articles/secure-ios-template.html

http://online.securityfocus.com/library/3516

In short:
Block Gnutella and or AFS and other File sharing system to get access from the external network.
Block any printing access from outside, i.e Port 515.
Block any SNMP access to Internal network.
Block NetBIOS at the border router, both in/out bound.
Ensure all systems that need to run ftp, http, telnet, ssh, rpc portmapper, lpd services are properly patched, as these tend to become the focus of all probe coming from external source.

Use of a firewall is imminent with proper policies/rules set. The University already have a Snort based IDS, that's great. They should look into tools or writes scripts to correlate the Firewall logs to snort logs. This will help them lock down the environment from being compromised. Also work on reducing false positives and customize rules to needs based on security policy of the site. Review other networking devises access list such as routers.

For more information on blocking/filtering traffic, see…
http://rr.sans.org/firewall/blocking_cisco.php
http://rr.sans.org/firewall/blocking_ipchains.php

Appendix A - Data Analysis process:

I used various tools to analyze the data set, which was quite extensive.
The Alert logs as well as the scan logs were concatenated into a master alert logs and scan logs.

I ran Snortsnarf perl scripts against this data. I came across a memory issues on the machine I was running it on. I resolved it by adding memory as well as extending swap space. It took over 18 hours to process this data.

Tools used were:

1. Snortsnarf.pl
2. Data separation perl scripts (written by Lenny Zelster
   See the URL below for information
   http://www.zelster.com/sans/practical/correlate.tgz
3. Other custom shell scripts
4. Perl v5.06
5. BASH Shell
6. GNU Awk
7. GNU Sed
8. Grep
8. Egrep
9. Microsoft Excel 2000
10. Microsoft Word 2000
11. Microsoft Visio 2000

I analyzed the alerts logs using Snortsnarf.pl perl script. I had to change all first two-octet references of MY.NET in the alert logs to 192.256, as Snortsnarf cannot interpret "MY.NET". This actually causes the source/destination addresses to appear as no IP in Snortsnarf Signature page.

To analyze scan logs I used Lenny Zelster perl scripts. Use of Excel was great to create chart/Link graphs. Visio was used to create the OOS logs table.

Data Analysis:
Scan logs data analysis was done by writing perl scripts, which are attached below;

```
#!/usr/bin/perl

$scanFile="/home/moloo/correlate/lib2/scans.txt";

$internalUDP="/home/moloo/correlate/data/internalUDP.txt";
$internalTCP="/home/moloo/correlate/data/internalTCP.txt";
$externalUDP="/home/moloo/correlate/data/externalUDP.txt";
$externalTCP="/home/moloo/correlate/data/externalTCP.txt";
```

```
    # Open notesLdifFile
    open(scanfile, $scanFile) || die " Can't open $scanFile $!";

    open(IUDP, ">$internalUDP") || die " Can't open $scanFile $!";
    open(ITCP, ">$internalTCP") || die " Can't open $scanFile $!";
    open(EUDP, ">$externalUDP") || die " Can't open $scanFile $!";
    open(ETCP, ">$externalTCP") || die " Can't open $scanFile $!";

    my ($iudp,$eudp,$itcp,$etcp);

    while ( <scanfile> ) {

        next if !(/^Jan/);

        ($d1,$d2,$d3,$src,$arrow,$dst,$protocol) = split(/ /,$_);


        # If the src and the dst start with mynet and protocol is a UDP
        # packet then dump in internalUdp
        if (($src=~ /MY.NET/) && ($dst=~ /MY.NET/) && ($protocol=~ /UDP/)){
            print IUDP $_;
            $iudp++;
        }

        if (($src=~ /MY.NET/) && ($dst=~ /MY.NET/) && !($protocol =~ /UDP/)){
            print ITCP $_;
            $itcp++;
        }

        if (!($src=~ /MY.NET/) && ($dst=~ /MY.NET/) && !($protocol =~ /UDP/)){
            print ETCP $_;
            $etcp++;
        }

        if (!($src=~ /MY.NET/) && ($dst=~ /MY.NET/) && ($protocol =~ /UDP/)){
            print EUDP $_;
            $eudp++;
        }
    }

    print "Total internal UDP scans: $iudp\n";
    print "Total internal TCP scans: $itcp\n";
    print "Total external UDP scans: $eudp\n";
    print "Total external TCP scans: $etcp\n";
```

This was used to generate information to create the scan logs link graph.

References:

SANS Web site
http://www.sans.org

Incidents/Dshield Web site
http://www.incidents.org
http://www.dshield.org

Carnegie Mellon's CERT Advisories
http://www.cert.org

CVE Database
http://cve.mitre.org

CyberKit Web site
http://www.cyberkit.net/index.html

IETF RFC References
http://www.ietf.org

Microsoft Web site
http://www.microsoft.com

Cisco's Web site – Security on networking gear
http://www.cisco.com

Internet Security Systems & its X-Force database
http://www.iss.net
http://www.xforce.net

Wingate Proxy Server/Firewall
http://www.wingate.net

Dartmouth University
http://www.ists.dartmouth.edu/IRIA/knowledge_base/tools/adorefind.htm

Redhat Linux reference
http://www.redhat.com/support/alerts/Adore_worm.html

Trojan Reference site
http://www.glocksoft.com/trojan_list/RC1_trojan.htm

SNMP Reference
http://www.faqs.org/faqs/by-newsgroup/comp/comp.protocols.snmp.html

Robert Graham's IDS White paper
http://www.robertgraham.com/pubs/network-intrusion-detection.html

Fyodor's NMAP & other great white paper on security.
http://www.insecure.org
http://www.insecure.org/reading.html

Snort's Web site
http://www.snort.org

Reading material used:
All SANS curriculum books (IDS Track)
Stephen Northcutt, Mark Cooper, Matt Fearnow & Karen Frederick. Intrusion Signatures and Analysis (SANS – GIAC)
Stevens, W. Richard. TCP/IP Illustrated, Volume 1. Reading: Addison-Wesley Professional Computing Series.

Other security Websites used:
http://www.securityfocus.com
http://www.ciac.org/ciac/
http://project.honeynet.org

For Registration information, I used
http://www.arin.net
http://www.apnic.net
http://www.ripe.net
http://www.samspade.org

For TCP/IP port references, I used
http://www.isi.edu/in-notes/iana/assignments/port-numbers
http://www.snort.org/Database/portsearch.asp
http://advice.networkice.com/advice/Exploits/Ports/
http://www.simovits.com/nyheter9902.html
http://www.pointman.org/Docs/port-list
http://www.sans.org/newlook/resources/IDFAQ/oddports.htm
http://www.good-stuff.co.uk/useful/portfull.html
http://www.clic.net/~hello/puppet/nnports.html
http://www.tlsecurity.com/trojanh.htm
http://www.ec11.dial.pipex.com/port-num3.htm
http://www.iana.org/assignments/port-numbers

Search Engine used:
http://www.google.com