Global Information Assurance Certification Paper

Dwayne Samson
GIAC GCIAPractical (version 3.5)
Submitted September 12, 2004

Table of Contents

Abstract

This paper is for GIAC CGIA version 3.5 certification and will consist of three assignments. In the first assignment I will design and IDS architecture for an enterprise environment. This paper will discuss some of the challenges one may encounter with deployment hardware and placement of IDS sensors.

In the second assignment of this paper I will analyze three separate network detects. The three detects are as follows; Scan Proxy 8080 attempt, Short UDP Packet Length and NETBIOS SMB-DS IPC$, NETBIOS SMB-DS. With each analyze detect this paper will provide a description of the attack, the mechanisms used for the attack, along with any correlations.

Finally, in the third assignment of this paper we will be tasked to perform a security audit for a higher learning institution. We will be provided with five days of contiguous log, alert and OOS files. We will provide recommendations along with any correlations from the data we will get from these files.

1 Assignment #1 Design an Enterprise IDS Architecture

1.1 Summary of Enterprise Network

The network in which this paper will be designing an IDS architecture around will consist of approximately 40,000 computers. The majority of these devices will be desktops. These desktops will be running *Microsoft*[1] operating systems. The operating systems are Windows XP, Windows 2000 Pro, and Windows NT. The remaining computers will be configured in a data center environment. The data center devices will be running Windows 2000, NT and 2003 Server along with *Sun's Solaris*[2], *HP-UX* [3], and some open source *BSD*[4] and *Linux*[5] operating systems.

Part of the enterprise's physical layout will comprise of two office locations. These two locations will be geographically situated 20 miles apart. Within the two locations a combined DMZ infrastructure environment will be built. This combined environment will allow for a physically redundant Internet across the entire enterprise with two egress and ingress points. The pipe size at both locations will be a DS3. These locations will be referred to as location **A** and location **B**. Both locations will be home to a Web server farm infrastructure. Both will be offering HTTP, HTTPS, FTP, SMTP, DNS services publicly. Some of these services will be positioned behind content services devices from the *F5 Corporation* [6]. These content devices will provide a NAT (network address translation) and PAT (port address translation) service. This allows for the obfuscating of the server's real IP address and port services from users who are requesting them. Location **A** will house approximately 19,000 users and location **B** will house a user population of approximately 7000.

Most of the remaining user population will be disbursed to satellite locations situated around the Continental United States. Many of these satellite offices will house a small number of users, somewhere in the neighborhood of 3 to 25.A few of the larger offices will employ up to 400 users. All sites will be using a frame Circuit connection or an office to office VPN connection back to either location **A** or **B**. Users residing at these locations will be accessing location **A** or **B** for egress traffic to the internet.

Located in the Pacific Rim region of the world is another satellite location. This location will have a dedicated leased circuit back to location **B**. This office will have its own connections to the internet. There will be two *T1* circuits used for internet access. Local users will use these connections for egress traffic. This location will use a SSL VPN connection for remote users to access the network. User population in this location is approximately 500.

Positioned at the **A** and **B** locations will be choke routers which will facilitate external connections to third party or business partner companies. Access will be controlled to and from these locations with access-lists which are applied to the interfaces on the choke routers.

Remote users will need to access the network in locations **A** and **B**. To accomplish this, the remote user's local PC will be configured to run VPN client software. This software in conjunction with the user credentials will allow remote access connections to a VPN device located at the either of the locations.

1.2 More Detailed background on the enterprise network

The interior routing and switching is done using the *Cisco Systems* [7] family of routers and switches. This is important to note, they will be used in the SPANS for the deployment of IDS sensors. The VPN technology being used is the *Nortel Networks Contivity* [8] products. These products will consist of head end switches and remote office switches. Contivity software will be used for remote user VPN connections. IPSEC technology will be used for the remote user VPN connections. IPSEC[9] is an encryption technology used in the transfer of data. We will be looking at the VPN traffic after data decryption has occurred. We will be deploying *Check Point* [10] *NG* Firewall products. They will be deployed on the perimeter at locations **A** and **B** and in the Pacific Rim location as well. Check Point firewalls will be used as to protect the user space into the DMZ egress (NAT) traffic out to the Internet. Network taps will be used in the configuration and deployment of the IDS sensors.

The design and placement of the IDS sensors will need to factor for the following. Location **A** and **B** ingress and egress traffic, DMZ traffic and traffic to and from the web server farms. The design will also have to address remote user connections after decryption along with the Internet connection being used in the Pacific location. Finally, an IDS will be needed to monitor traffic from the satellite offices as well as the business partner and third party connections into the network.

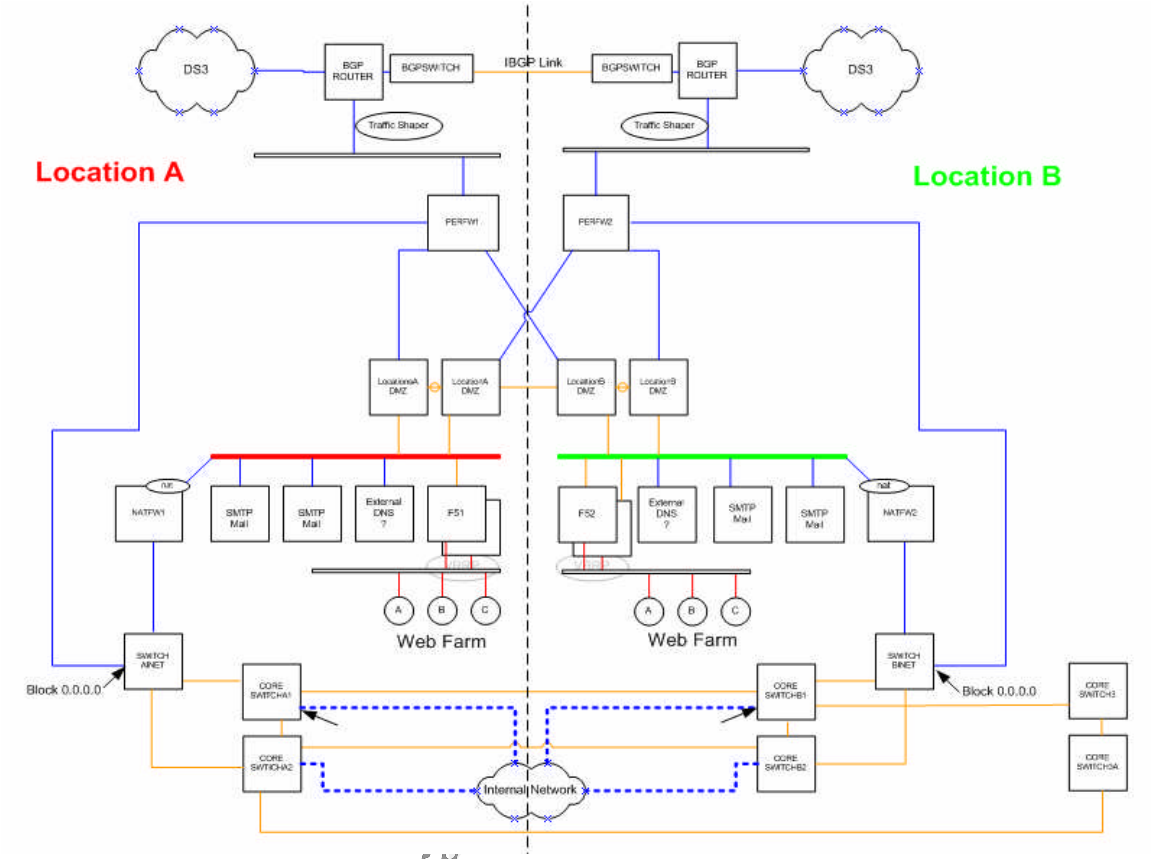
The following diagrams show pre network locations with no IDS deployments.

Diagram 1.2a Locations A and B with no IDS

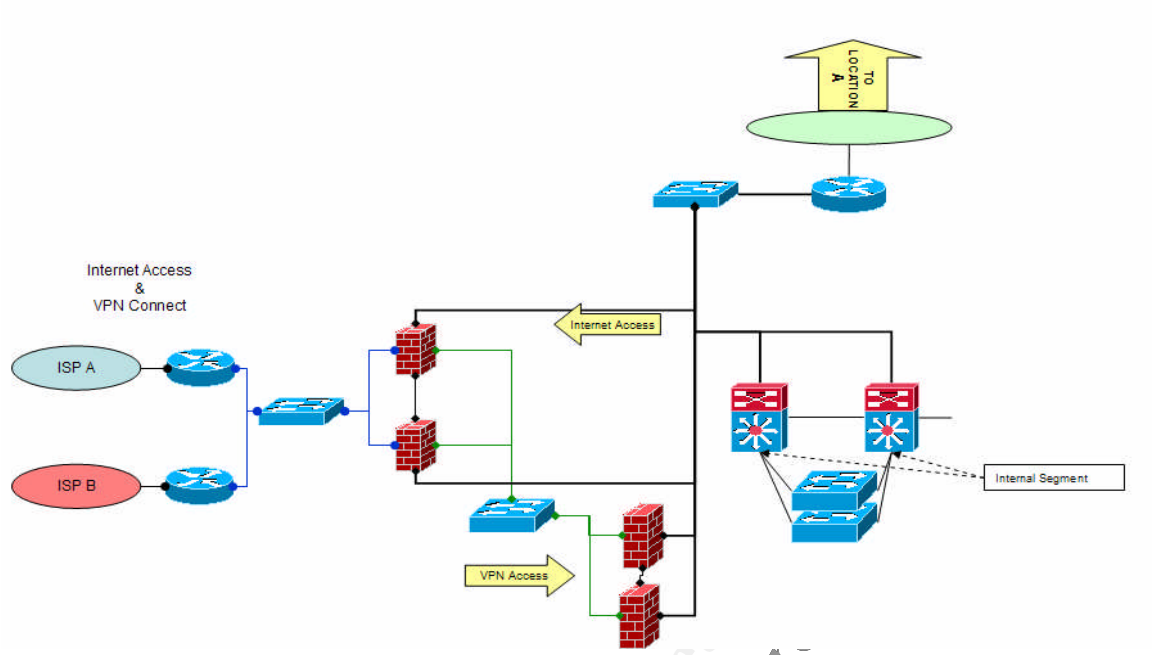
Diagram 1.2b Location of Pacific Rim

Diagram 1.2c Business Partner and Satellite Connections

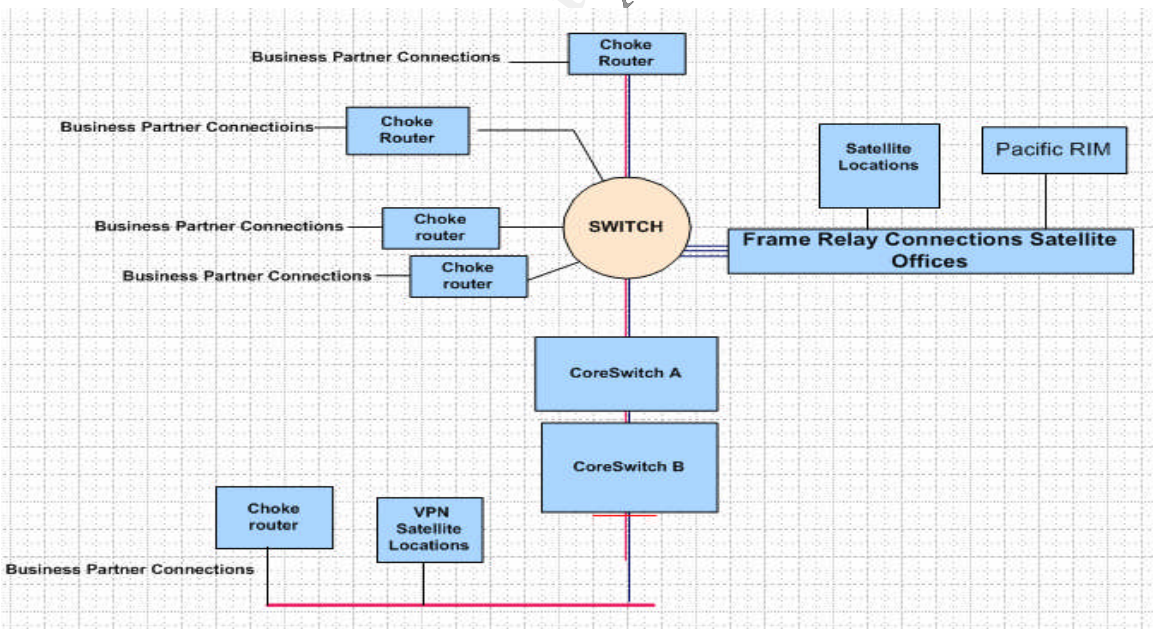
Diagram 1.2d VPN Head End Connections



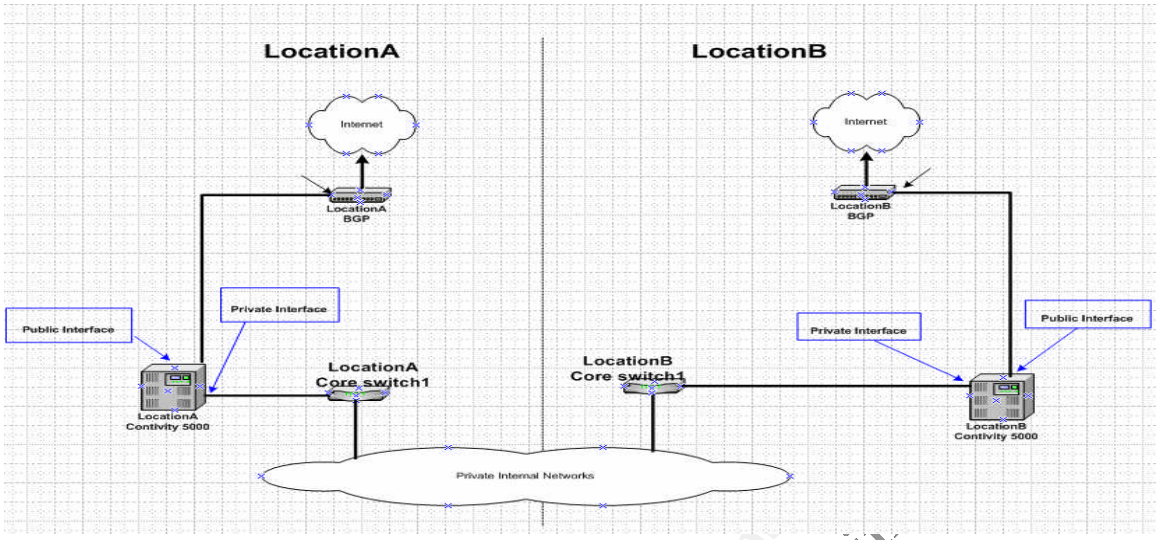
1.2a Locations A and B



1.2b Pacific Rim



1.2c Satellite and Business Partner Connections



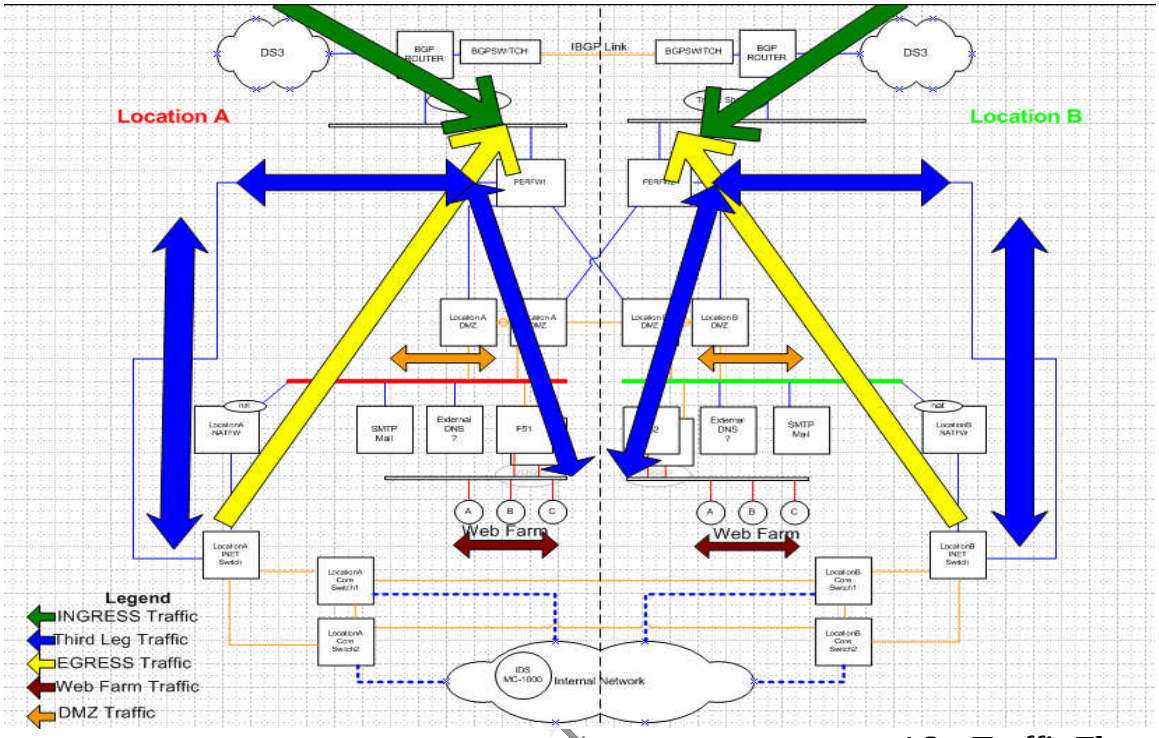
1.2d VPN head end Connections

1.3 Proposed IDS deployment and configurations

Now that we have a general idea of what the enterprise network will physically look like, we will turn our focus to the placement of our IDS sensors. The IDS hardware to be installed at all locations will be from *SourceFire Network Security*[11]. We will be using two particular SourceFire devices. The first device and its related hardware and software are as follows: The NS-1000 network sensor. This unit is a 1U rack mount unit. They will be running Linux *Red Hat 8* [12]for the *Intel* [13]architecture. They will have a 2.8GHZ CPU with 512M of RAM installed. The hard drive will store 36 gigabytes with a 10k UltraSCSI disk controller. The box will have two 10/100/1000 network interface cards. The NS-1000 uses the *SNORT*[14] rule base detection engine and runs at 45MPS. The second box being deployed is the SourceFire MC-1000 master console. This is a 1U rack mount unit as well. The hardware and software specifications for this unit are Dual 2.8GHZ CPU with 2GB of RAM installed and running Linux Red Hat 8 for Intel architecture. There will be a 66GB hard drive installed with a Ultra 160/wide channel controller. The MC-1000 will be used as our management console; from here we can aggregate event information from each of the NS-1000 sensors. The master console will allow central management to all distributed SourceFire sensors. Policies, alert responses, and user privileges will be configured from here also. Current revision of code on the Sourcefire sensors will be Version 3.0.

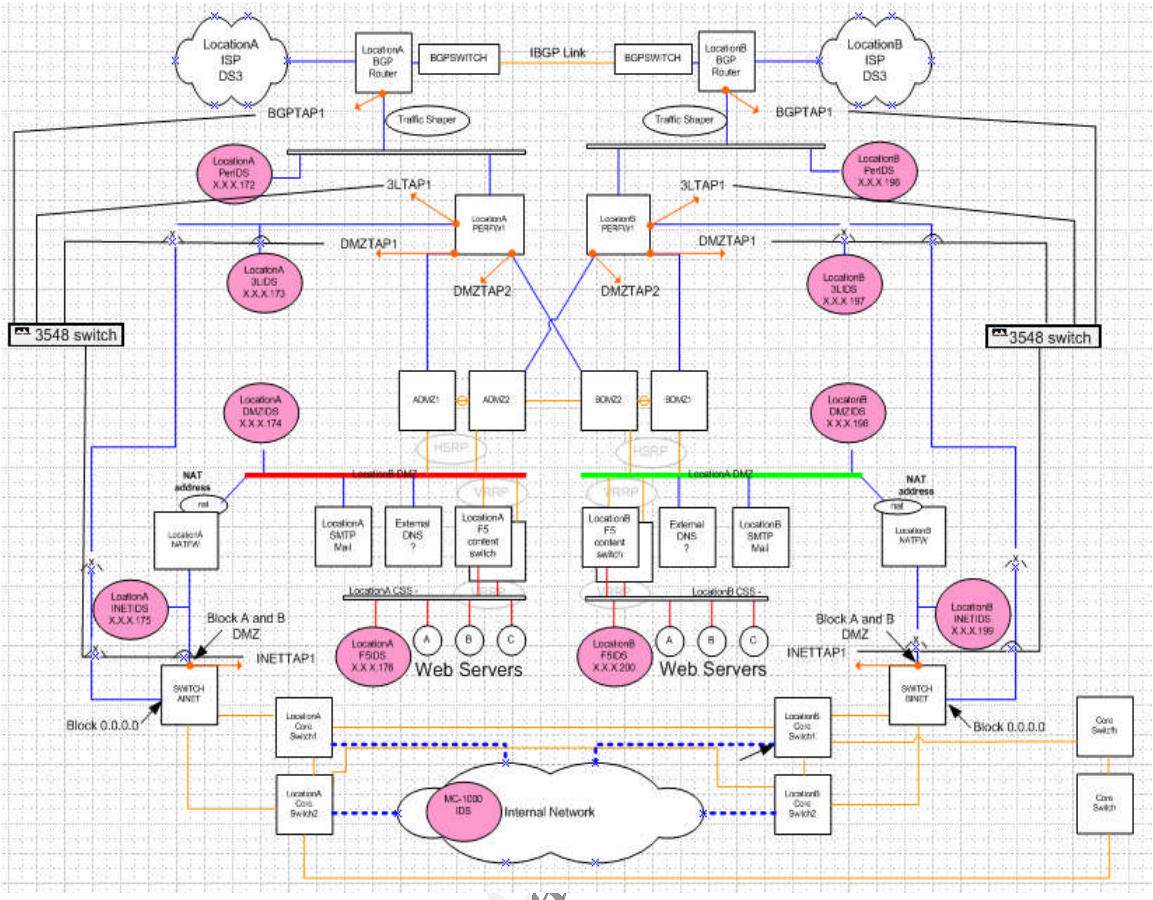
In sites **A** and **B,** we will deploy sensors that will parallel each other in function in both locations. We will need to observe traffic ingress from the perimeter at the **A and B** locations. We will need to observe traffic to and from each of the web server farms. Please note that traffic to the web farm originating from the enterprise user segment does not pass through the 'NATTED' firewall connection. This traffic will flow thru the third leg of the 'INETSWITCHES'. The reverse rule applies that traffic originating from the DMZ destined for the enterprise user segment will flow out of the third leg of the

perimeter firewall. With that said we will need to monitor the third leg connection in both locations. We will monitor egress traffic from the user segment along with traffic within the DMZ at both locations. Diagram 1.3a shows the data flows.



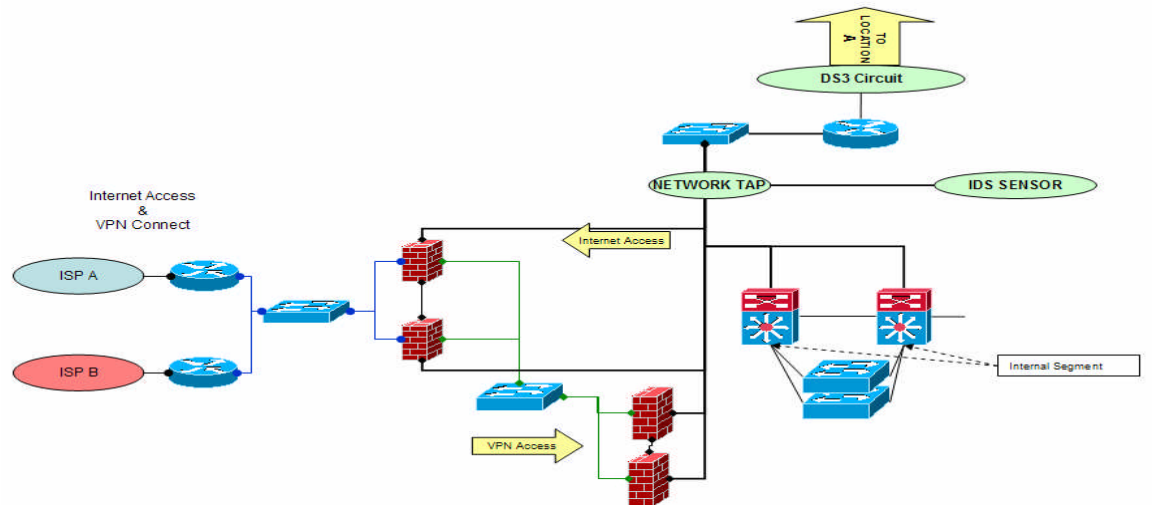
1.3a Traffic Flow

Diagram 1.3b shows that we have placed NC-1000 IDS sensors at the ingress perimeter points for both locations. We have placed NC-1000's in the third leg segment and inside the web server farms and Egress points at both locations. One of the network interfaces on all the IDS sensors in location **A** will be addressed on a local area network space of X.X.7.0 /24. At location **B,** one of the interfaces on all the NC-1000 sensors will be addressed on a local area network space of X.X.9.0 /24. These interfaces will communicate with MC-1000 sensor located inside the user/data center segment and addressed as X.X.X.177. Communication between the master console and network sensors will be over a secure socket SSL connection. Firewall rules will be applied to allow for communication between the NC-1000 devices in and out the DMZ to the MC-1000 master console located in Location **A**. Diagram 1.3b show the placement of the IDS sensors and taps for the **A** and **B** location along with the network IP addressing.



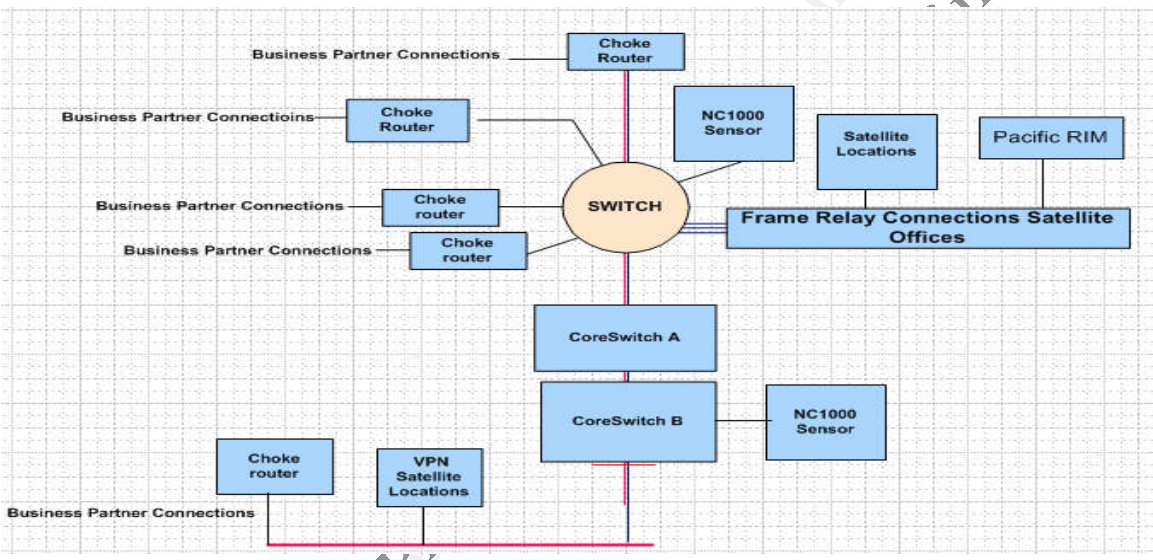
1.3b IDS placement and addressing

For the Pacific Rim location, one NC-1000 sensor will be used to monitor the user egress traffic and post SSL VPN decryption. The sensor will be addressed on X.X.4.100 /24 and will communicate back to the MC-1000 housed at location **A**. Diagram 1.3c shows the placement of the sensor.



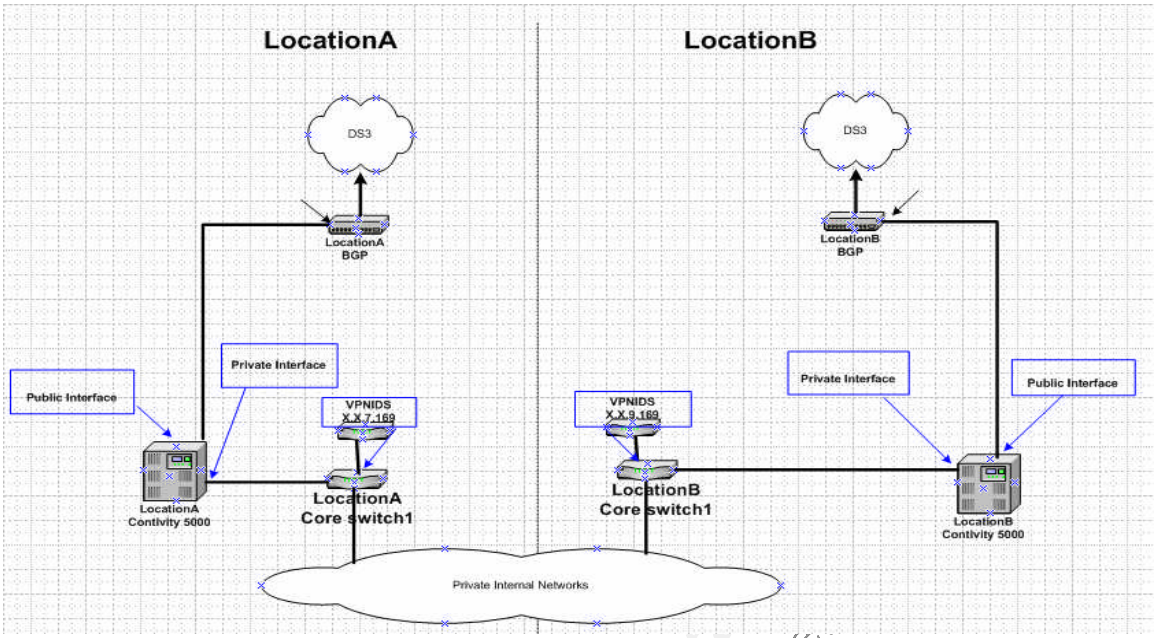
1.3c Pacific Rim with IDS deployment

We will use two NC-1000 sensors for the business partner third party connections, along with frame relay and VPN site to site connections. The current circuit aggregation design is considered a spoke and wheel. This design has the majority of the connections aggregating into a splat type environment. We will need to place one of the IDS sensors where we can see traffic traveling between business partners or frame connected remote offices. To accomplish this we will position the IDS sensor on the splat switch. This sensor will be configured with an IP address of X.X.7.170 and will communicate with the MC-1000 master console. The second IDS sensor will be configured with an X.X.7.168 IP address and place into the Core **B** Switch. This will allow us to capture the VPN offices connections post decryption along with the one off business partner connections. This will also communicate with the MC-1000. Diagram 1.3d show us this.



1.3d Business Partner Satellite location IDS Deployment

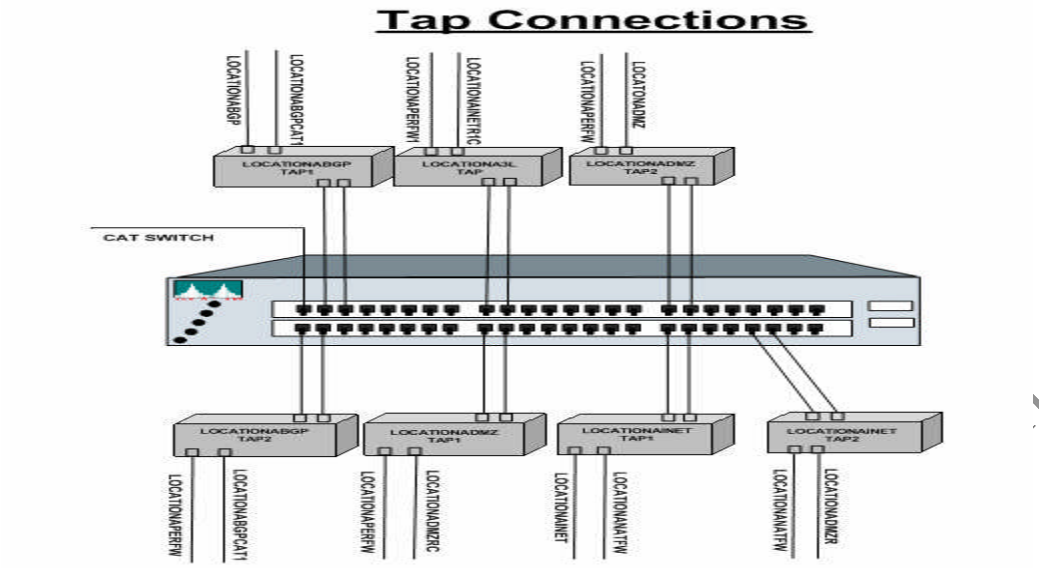
Placement of the NC-1000 sensors for remote VPN connections to the **A** and **B** locations has them installed on the Private side of Nortel Contivity switches. The placement of the sensors at these positions will allow us to look at the traffic post data decryption. These boxes will be sending their data collection back to the MC-1000 master console. The sensors will be addressed as X.X.7.169 and X.X.9.169 respectively. Diagram 1.3e shows us that.



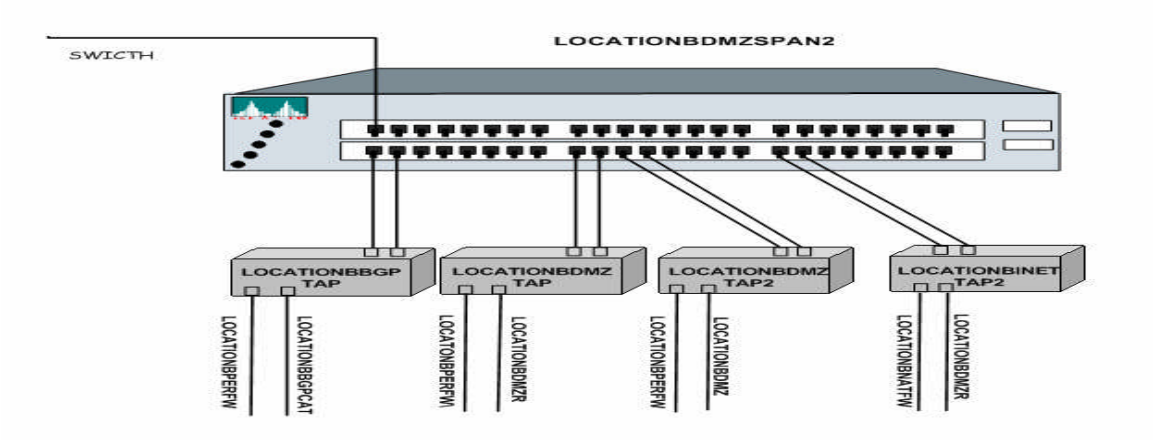
1.3e Location A and B remote VPN IDS Placement

1.4 Network Taps

This section of this paper will discuss the use of network taps in the enterprise. Taps are used to permit permanent access ports allowing for passive monitoring. They can be configured between two network devices and allow access to them from a monitoring device. A monitoring device connected to a tap will receive traffic as if it was connected directly on the wire. Taps are passive devices; they do not act on the network traffic other than possibly regenerating or splitting the signal. If a tap were to fail, the traffic will continue to flow through it. The tap will not allow the sensor to inject traffic into the data stream. The use of the tap addressed the problem of span limitations that were inherent on the devices that spans were configured on such as switches. These limitations include the size of the back plane of the device as well as packet loss and span limitations. The use of taps will make the NIDS sensor more secure by preventing attackers to directly attack the NIDS. Please note, although this configuration utilizes taps some of spans are aggregating into a Cisco 3548. These switches were lab tested with the result of no packet loss or saturation of the switches back plane. They were also deployed due to the amount of fiber spans that were configurable on the switch. Diagram 1.3b, 1.4a, and 1.4b shows how taps were deployed in this enterprise environment.



1.4a Location A Taps



1.4b Location B Taps

1.5 Stealth interfaces

The other NC-1000 interface will be configured to run in stealth mode. This mode is configured on the sensors to allow for passive monitoring. There will be no IP address assigned to this interface. The configuration of the stealth interface on the SourceFire sensors is labeled as follows

```
root@X.X.X. 9:~# ifconfig
      bond0     Link encap:Ethernet  HWaddr 00:03:47:32:7F:18
      UP BROADCAST RUNNING NOARP PROMISC MASTER MULTICAST
MTU:1500  Metric:1
      RX packets:988345862 errors:5 dropped:0 overruns:2078 frame:5
      TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
```

RX bytes:151010746 (144.0 Mb) TX bytes:0 (0.0 b)

The interface labeled as **bond0** was placed into stealth mode by applying the ***promisc*** command to that interface. This interface should not generate any network traffic.

1.6 Monitoring

The monitoring of the IDS sensors will be conducted by a team of security specialists. These specialists will rotate on a scheduled basis. The scheduled period is defined as every two weeks starting on Tuesday. The IDS sensors will be broken up into three groups. The first group will be comprised of the remote user, satellite offices, third party and Pacific Rim IDS sensors. The second group will be comprised of all the IDS sensors at location **A**. The final group will encompass all the IDS sensors at location **B**. The specialists will rotate between the defined IDS sensor groups. During this scheduled period it will be the responsibility of the assigned specialist to tune out any noisy rules and update to the latest current rule sets using the open source tool oinkmaster[15]. All changes performed as such will be documented to a repository. This will include type of change, sensor name and timestamp. The specialists will be logging on to the master console MC-1000. From here they will be able to perform their monitoring duties on their assigned IDS sensors. They will then analyze the data. Any information found to be abnormal in nature will be analyzed for severity and criticality and then acted upon. This may require the specialist to contact the owner or system administrator of the device being targeted as well as the contact or system admin from the source network. The specialist will monitor over weekends and holidays. This however is not a 24/7 shop, the amount of time spent analyzing and frequency spent looking at the alerts is left to the discretion of the specialist. Current guide line will be checking first thing in the morning and periodically for the remainder of the day and evening. However any known new vulnerabilities and exposures, along with security warnings can and will affect the frequency of monitoring. Currently there is no plan for any out of bandwidth management of the sensors. Specialist will analyze the data from their workstations connecting into the master console using a username and password. Remote access to the IDS sensors will be thru a VPN connection back into the network. The inherent email tool found with in the SourceFire application will be configured to create customized email alerts. These alerts will be mailed to all member of the security team. These alerts will be threshold barriers such as hard disk and memory usage. There are also custom rule alerts for specific signatures such as viruses.

1.7 Backups

The backup of the data will be by SFTP (Secure File Transfer Protocol) to a *NAS* (network attached storage device).The current size of the *NAS* share is 200 gigabytes. These backups will be initiated by a CRON job. This job will tar up the data and will be run on all IDS sensors on a bi-monthly basis. The backup of this data can be used in the future for disaster recovery and upgrades. Data will not be encrypted for storage. Due to the nature of the business, data will be kept for an indefinite period of time. Currently

compliance regulations and legal statues are being reviewed to determine what would be an appropriate time period for the retention of the data.

1.8 Conclusion

The preceding illustrates at a minimum, where the deployment of IDS sensors will need to be positioned with in the enterprise network. Additional devices could have been installed. For example, redundant equipment could have been deployed in the Pacific location as well as for the third party connections. And we could have had a redundant master console. However, one factor that I have not spoke to as it pertains to the construction of the enterprise, is cost. Cost will determine how many sensors can be purchased along with network taps and SPAN switches. That said, this configuration has met the requirements defined, Ingress and Egress traffic along with the third leg, DMZ, and Web farm traffic being monitored at both the **A** and **B** locations. Third Party Business connections as well as remote user connections are being monitored. The deployment of the IDS sensor at the Pacific Rim location allows the monitor of that remote site as well.

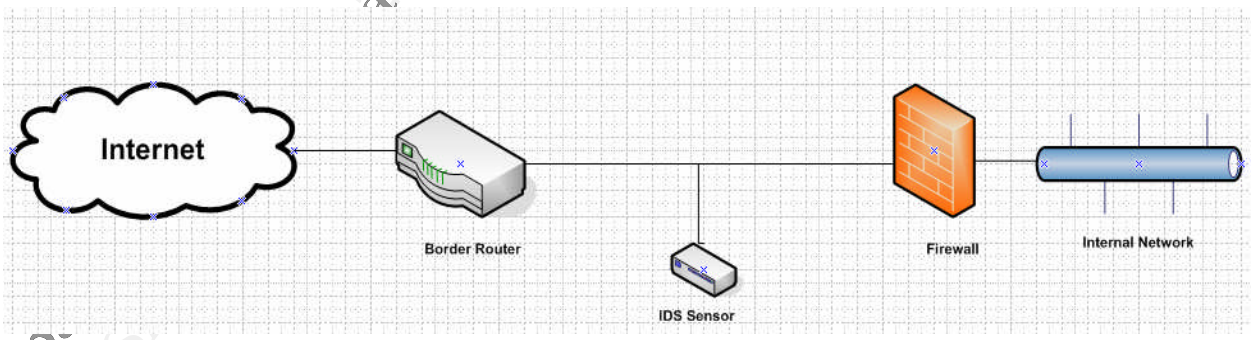
2 Assignment #2 – Network Detects:

2.1 Detect #1 Scan Proxy 8080 attempt

2.1a Source of the trace:

This detect was extracted from a company network, The IP addresses have been sanitized.

2.1a1 Network Diagram



2.1b Detect generated by

This detect was generated by Snort version 2.0.5 with a custom rules set configuration. The following Snort syntax was used.

Snort –b –A fast -c <file locations>/snort.conf -i eth1 –D

-b log packet in tcpdump format (used for speed, much faster)

-A set alert mode, fast, full, console, or none

-c use rules file

-i listen on interface eth1

-D run snort in background (daemon mode)

2.1c Alerts generated from trace

05/12-03:42:29.206867 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:3129 -> x.x.x.197:8080
05/12-03:42:32.200004 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:3129 -> xx.x.x.197:8080
05/12-03:42:35.400088 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:3129 -> x.x.x.197:8080
05/12-03:42:44.219102 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:4265 -> x.x.x.197:8080
05/12-03:42:47.216054 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:4265 -> x.x.x.197:8080
05/12-03:42:50.415876 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:4265 -> x.x.x.197:8080
05/12-03:42:52.723813 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:1386 -> x.x.x.197:8080
05/12-03:42:55.715287 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:1386 -> x.x.x.197:8080
05/12-03:42:58.916120 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:1386 -> x.x.x.197:8080
05/12-03:43:02.114964 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:1386 -> x.x.x.197:8080
05/12-03:43:05.313619 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:1386 -> x.x.x.197:8080
05/12-03:43:08.546533 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:1386 -> x.x.x.197:8080
05/12-03:47:05.038038 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classifica
tion: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:2355 -> x.

x.x.197:8080
05/12-03:47:08.032379 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:2355 -> x.x.x.197:8080
05/12-03:47:11.233398 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:2355 -> x.x.x.197:8080
05/12-03:47:14.431162 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:2355 -> x.x.x.197:8080
05/12-03:47:14.717008 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:2662 -> x.x.x.197:8080
05/12-03:47:17.711918 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:2662 -> x.x.x.197:8080
05/12-03:47:20.909567 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:2662 -> x.x.x.197:8080
05/12-03:47:22.436430 [**] [1:620:2] SCAN Proxy (8080) attempt [**] [Classification: Attempted Information Leak] [Priority: 2] {TCP} 209.98.98.116:2160 -> x.x.x.197:8080

2.1d Rule which was triggered from trace

alert tcp $EXTERNAL_NET any -> $HOME_NET 8080 (msg:"SCAN Proxy Port 8080 attempt"; stateless; flags:S,12; classtype:attempted-recon; sid:620; rev:6;)

Syntax for rule
2.1e Rule Header

<**Aler**t – *output format*> <**tcp**- *proctocol being used*> **<$EXTERNAL_NET** *–variable for External networks*> <**any** *–source port*> <**->** *-conversation dircection*> **<$HOME_NET** *–variable for defined the internal network*> <**8080** *–destination port*>

2.1f Rule Options

<(**msg:"SCAN Proxy Port 8080 attempt"** *–message displayed by alert*> <**stateless** *– flow control option activating on packets regardless of state*> <**flags:S,12** *–determine which tcp flag is set,* **S** *– syn flag,***12** *–determines if reserved bits 1 and2 are set*> <**classtype: attempted-recon** *– classification identifier*> <**sid:620** *–snort rule unique identifier*> <**rev:6** *–version number of the rule*>

2.1g Probability the source address was spoofed

The probability of the source address being spoofed is very low but not out of the realm of possibility that it is being spoofed. There are scanning techniques such as idlescan[16] which can scan a network with out sending a single packet to the target from its own IP address. With this in mind I used the program P0f[17] a passive OS finger printing tool and ran it with the snort tcpdump files for the May 12th date with a filter for the source address of 209.98.98.116, following are samples generated from this program.

```
209.98.98.116:3250 - FreeBSD 4.6-4.8 (up: 4731 hrs)(distance 9, link: ethernet/modem)
209.98.98.116:3447 - FreeBSD 4.6-4.8 (up: 4731 hrs) (distance 9, link: ethernet/modem)
209.98.98.116:3250 - FreeBSD 4.6-4.8 (up: 4731 hrs) (distance 9, link: ethernet/modem)
209.98.98.116:3949 - FreeBSD 4.6-4.8 (up: 4731 hrs) (distance 9, link: ethernet/modem)
209.98.98.116:3949 - FreeBSD 4.6-4.8 (up: 4731 hrs) (distance 9, link: ethernet/modem)
209.98.98.116:3949 - FreeBSD 4.6-4.8 (up: 4731 hrs) (distance 9, link: ethernet/modem)
209.98.98.116:2355 - FreeBSD 4.6-4.8 (up: 4731 hrs) (distance 9, link: ethernet/modem)
209.98.98.116:2355 - FreeBSD 4.6-4.8 (up: 4731 hrs) (distance 9, link: ethernet/modem)
209.98.98.116:2355 - FreeBSD 4.6-4.8 (up: 4731 hrs) (distance 9, link: ethernet/modem)
209.98.98.116:2355 - FreeBSD 4.6-4.8 (no RFC1323) (distance 9, link: ethernet/modem)
209.98.98.116:2662 - FreeBSD 4.6-4.8 (up: 4731 hrs) (distance 9, link: ethernet/modem)
209.98.98.116:2662 - FreeBSD 4.6-4.8 (up: 4731 hrs) (distance 9, link: ethernet/modem)
209.98.98.116:2662 - FreeBSD 4.6-4.8 (up: 4731 hrs) (distance 9, link: ethernet/modem)
```

The results show the likely hood of an Open Source FreeBSD box being used with distance hop count of around 9. I then ran a traceroute command to this box from the network border router and was able to obtain a hop count of 8.

```
border router>traceroute 209.98.98.116
Type escape sequence to abort.
Tracing the route to hectate.visi.com (209.98.98.116)
  1 x.x.x.x 4 msec 4 msec 4 msec
  2 x.x.net (x.x.x.x) [AS x] 4 msec 4 msec 8 msec
  3 x.x.xt.x (x.2) [AS x 24 msec 24 msec 24 msec
  4 x.x.x.x.x (x221) [AS x] 32 msec 32 msec 32 msec
  5 x.x.x.x [AS x] 32 msec 32 msec 32 msec
  6 pos1-1-0.core-2.mpls.visi.com (209.42.3.245) [AS 8015] 36 msec 36 msec 40 msec
  7 ge6-0-0.core-1.mpls.visi.com (209.98.3.222) [AS 8015] 32 msec 40 msec 40 msec
  8 hectate.visi.com (209.98.98.116) [AS 8015] 36 msec 32 msec 40 msec
```

The correlation between the two is why I believe the probability that the source address is being spoofed is low. The hop count from the attacker to my network and the hop count from my network to the attacker is very similar, along with the attacking host answering my traceroute.

2.1h Description of the attack

The attacker is scanning multiple ports that are normally related to known proxy services. The attacker is looking for that port to be open. The ports are 8000, 8080, 31278, 4480, 6588, 1075, 1182, 8085, and 7033 all documented proxy service ports. Where the informational scan stops more deviant behavior may take over. If the attacker finds the open proxy he can then test it to see if it is a vulnerable proxy. There are numerous vulnerabilities associated with misconfigured proxy servers.

Scan log taken from the May 12th date show the source address sending these SYN packet to know proxy ports.

May 12 03:42:29 209.98.98.116:3129 -> x.x.x.197:8080 SYN ******S*
May 12 03:42:29 209.98.98.116:3339 -> x.x.x.197:8081 SYN ******S*
May 12 03:42:29 209.98.98.116:3549 -> x.x.x.197:8090 SYN ******S*
May 12 03:42:29 209.98.98.116:3757 -> x.x.x.197:5490 SYN ******S*
May 12 03:42:29 209.98.98.116:3974 -> x.x.x.197:7033 SYN ******S*
May 12 03:42:29 209.98.98.116:4182 -> x.x.x.197:8085 SYN ******S*
May 12 03:42:29 209.98.98.116:4365 -> x.x.x.197:8095 SYN ******S*
May 12 03:42:29 209.98.98.116:4603 -> x.x.x.197:8100 SYN ******S*
May 12 03:42:35 209.98.98.116:2413 -> x.x.x.197:4480 SYN ******S*
May 12 03:42:32 209.98.98.116:2660 -> x.x.x.197:6588 SYN ******S*
May 12 03:42:32 209.98.98.116:2915 -> x.x.x.197:8000 SYN ******S*
May 12 03:42:32 209.98.98.116:3129 -> x.x.x.197:8080 SYN ******S*
May 12 03:42:32 209.98.98.116:3339 -> x.x.x.197:8081 SYN ******S*
May 12 03:42:32 209.98.98.116:3549 -> x.x.x.197:8090 SYN ******S*
May 12 03:42:32 209.98.98.116:4182 -> x.x.x.197:8085 SYN ******S*
May 12 03:42:32 209.98.98.116:4365 -> x.x.x.197:8095 SYN ******S*
May 12 03:42:32 209.98.98.116:4603 -> x.x.x.197:8100 SYN ******S*
May 12 03:42:37 209.98.98.116:3426 -> x.x.x.197:6588 SYN ******S*
May 12 03:42:37 209.98.98.116:3679 -> x.x.x.197:8000 SYN ******S*
May 12 03:42:35 209.98.98.116:3129 -> x.x.x.197:8080 SYN ******S*
May 12 03:42:35 209.98.98.116:3339 -> x.x.x.197:8081 SYN ******S*

Correlation done with the Port scan log files for the day also has the attacker sending single TCP SYN packets to ports which are not proxy related services such as port 80,81 HTTP ports, port 21 the FTP port, port 23 the Telnet port as well as port 1182 Jaunt port, which is used for web based remote control.

May 12 03:47:14 209.98.98.116:3713 -> x.x.x.197:1181 SYN ******S*
May 12 03:47:14 209.98.98.116:4629 -> x.x.x.197:23 SYN ******S*
May 12 03:47:03 209.98.98.116:2794 -> x.x.x.197:80 SYN ******S*
May 12 03:47:03 209.98.98.116:3045 -> x.x.x.197:81 SYN ******S*
May 12 03:47:32 209.98.98.116:1074 -> x.x.x.197:21 SYN ******S*

It appears the attacker this day was looking for more than just open Proxy services. However the majority of his scans were being targeted at known Proxy services.

2.1i Attack mechanism

In this detect, the traffic would be considered a stimulus and would be very noisy. The attacker is sending a single TCP SYN packet looking to illicit a single TCP SYN ACK response from the target host on a number of known proxy ports. If the attacker is able to determine if an open proxy port is available, he can then launch attacks from the proxy device to other hosts. This will help in obfuscating his real source IP address from the host he is attacking. The attacker can also run some known vulnerabilities against the open proxy service in hopes of compromising that vulnerability.

2.1j Correlations

Firewall logs for that day show connection attempts from 209.98.98.116 were being dropped to targeted host. Eric Montcalm in his CGIA practical states multiple CVE's for ISA, Squid, and Cisco Proxy services. Below are the ones listed.

Name	CVE-2002-0068
Description	Squid 2.4 STABLE3 and earlier allows remote attackers to cause a denial of service (core dump) and possibly execute arbitrary code with an ftp:// URL with a larger number of special characters, which exceed the buffer when Squid URL-escapes the characters.

Name	CVE-2002-0916
Description	Format string vulnerability in the allowuser code for the Stellar-X msntauth authentication module, as distributed in Squid 2.4.STABLE6 and earlier, allows remote attackers to execute arbitrary code via format strings in the user name, which are not properly handled in a syslog call.

Name	CAN-2002-0735 (under review)
Description	Format string vulnerability in the logging() function in C-Note Squid LDAP authentication module (squid_auth_LDAP) 2.0.2 and earlier allows

	remote attackers to cause a denial of service and possibly execute arbitrary code by triggering log messages.
References	• VULN-DEV:20020506 ldap vulnerabilities • URL:http://marc.theaimsgroup.com/?l=vuln-dev&m=102070267500932&w=2 • VULNWATCH:20020506 [VulnWatch] ldap vulnerabilities • URL:http://archives.neohapsis.com/archives/vulnwatch/2002-q2/0053.html • BUGTRAQ:20020506 ldap vulnerabilities • URL:http://online.securityfocus.com/archive/1/271173 • BID:4679 • URL:http://www.securityfocus.com/bid/4679 • XF:squidauthldap-logging-format-string(9019) • URL:http://www.iss.net/security_center/static/9019.php
Phase	Proposed (20020726)
Votes	ACCEPT(2) Cole, Armstrong NOOP(3) Cox, Wall, Foat
Comments	

Micosoft ISA

Name	CVE-2001-0239
Description	Microsoft Internet Security and Acceleration (ISA) Server 2000 Web Proxy allows remote attackers to cause a denial of service via a long web request with a specific type.

Name	CVE-2001-0658
Description	Cross-site scripting (CSS) vulnerability in Microsoft Internet Security and Acceleration (ISA) Server 2000 allows remote attackers to cause other clients to execute certain script or read cookies via malicious script in an invalid URL that is not properly quoted in an error message.

www.ARIN.NET was used to determine who owned the address space the attack was originating from.

```
OrgName:    Vector Internet Services, Inc.
OrgID:      VECT
Address:    12 S 6th St
Address:    Suite 630
City:       Minneapolis
StateProv:  MN
PostalCode: 55402
Country:    US

NetRange:   209.98.0.0 - 209.98.255.255
CIDR:       209.98.0.0/16
NetName:    VECTOR-BLK1
```

Mynetwatchman.com[18] provided the follow on activity from the attacker address.

Incident ID: 53870184	Source IP: 209.98.98.116
Provider Domain: geeks.org	
DNS Name:	
Total Event Count : 35	Total Distinct Agent: 1/0
Response : False Positive	
Status Description: Closed	
Exclusion Reason :	
Orig Autonomous Sys (AS)	**AS Responsible Party**
8015	visi.com
Network Name/NextNIC	**Start IP - End IP**
VECTOR-BLK1/DUMMY	209.98.0.0 - 209.98.255.255
NextNIC:99999 Whois provider:	

```
  Vector Internet Services, Inc.
OrgID:      VECT
Address:    12 S 6th St
Address:    Suite 630
City:       Minneapolis
StateProv:  MN
PostalCode: 55402
Country:    US
```

Most Recent Event									
Date/Time (UTC)	Agent Alias	Agent Type	Log Type	Target IP	# of IPs Targeted	Protocol/ Port	Port/ Issue Description	Source Port	Event Count
8 Jul 2004 22:17:02	pronetmcdralb	win32	SonicWall	68.157.x.x	1	255/65535	Unspecified Unknown	65535	49

2.1k Evidence of active targeting

This is a targeted reconnaissance scan to a specific host. I catted the scan logs for May 12th and used grep looking for a source address of 209.98.98.116, they showed TCP SYN packets from the fore mentioned attacking host to only one targeted host on my network. Furthermore I ran tcpdump against the snort.log file for the 12th with the filter of host 209.98.98.116, this showed all traffic from attacking host was a TCP packet with the SYN flag set directed against the single host on my network. The following was taken from the alert file for and was generated by the Snort Preprocessor portscan.

05/12-03:43:09.538321 [**] [100:2:1] spp_portscan: portscan status from 209.98.98.116: 17 connections across 1 hosts: TCP(17), UDP(0) [**]

2.1l Severity

(Criticality + lethality) – (system countermeasures + network countermeasures) = severity
(3 + 2) – (2 + 2) = 0

2.1m Criticality

The Proxy service is being targeted and it may or may not be a misconfigured Proxy..

2.1n Lethality

This is mostly and informational gathering scan. No evidence that attacker found an open proxy. Host being targeted is not configured and is not offering proxy services

2.1o System countermeasures

A well patched system that is not offering any Proxy services.

2.1p Network countermeasures

A firewall is in place and is blocking scanned ports that the attacker is scanning.

2.1q Defensive recommendations

Secure all public facing interfaces and apply latest security patches and service packs. Do not offer any services on the boxes that are not needed, (in this case the Proxy service). Deploy and ingress filtering firewall and in this case deny traffic to known Proxy ports.

2.1r Multiple choice question

Which of the following ports are documented WWW Proxy ports?

A) 4480

B) 8000

C) 8080

D) All of the above

Answer is D, 4480,8000 and 8080 are all documented WWW proxy ports.

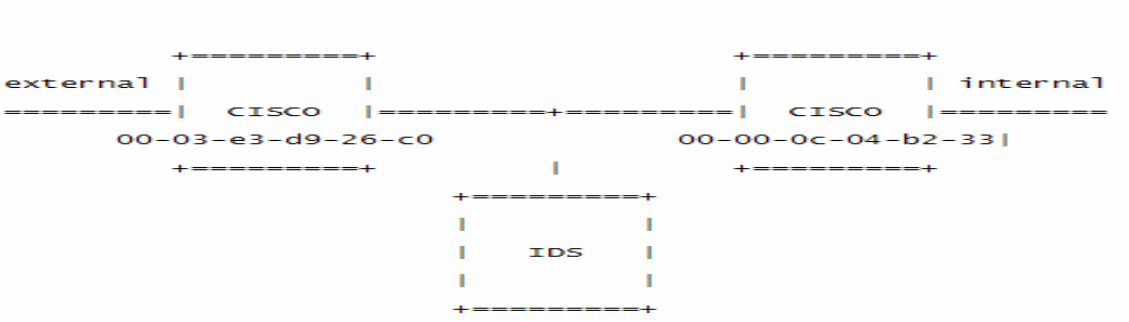
2.1s References

1. Snort 2.0 Intrusion Detection — Brian Caswell Technical Editor
2. Snort FAQ — http://www.snort.org/docs/faq.html
3. Snort — http://www.snort.org/

2.2 Detect #2 Short UDP Packet Length

2.2a Source of trace

This trace was downloaded from http://www.incidents.org/logs/Raw/2002.10.4
The network layout as I see it from the hardware addresses ascertained.



2.2b Detect generated by

This trace was generated by Snort version 2.1.2 with the default rule set configuration.

The following syntax was used.

Snort –b –A fast -c <file locations>/snort.conf -r <file location>2002.10.4

-b log packet in tcpdump format (used for speed, much faster)

-A set alert mode, fast, full, console, or none

-c use rules file

-r read file from the following location

2.2c Alerts generated by trace

[**] [116:97:1] (snort_decoder): Short UDP packet, length field > payload length [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
11/04-09:53:48.086507 151.196.186.220:0 -> 207.166.191.242:0
UDP TTL:112 TOS:0x0 ID:60425 IpLen:20 DgmLen:78
Len: 129

[**] [116:97:1] (snort_decoder): Short UDP packet, length field > payload length [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
11/04-09:53:51.146507 151.196.186.220:0 -> 207.166.191.7:0
UDP TTL:112 TOS:0x0 ID:19210 IpLen:20 DgmLen:78
Len: 129

2.2d Rule that generated trace

The rule was generated by the snort_decoder, the snort_decoder's function is to decode the raw data link packets off the wire captured using the *libpcap* library. In this example when the *libpcap* library received the packets from the network card driver it ran the *ProcessPacket* function. This function then called upon the *DecodeEthPkt* function. This then called upon the *DecodeIP* function which finally called upon the *DecodeUDPPkt* function. After this processing an abnormality was discovered in the packets generating the alert. The snort_decoder maps its alerts to the *gen-msg.map* file located in the etc directory of snort. This mapping speaks directly to the [116:97:1] (snort_decoder): Short UDP packet, length field > payload length error. Please note an error in the Snort alert which had a source port of 0. While according to the packet dump it had a source port of 1026. That said if the length field of the packet had been correct these packets would have triggered the BAD TRAFFIC UDP Port 0 rule from the snort rule set.

2.2.e Possibility the source address was spoofed

The possibility that this source address was spoofed is low. It is a UDP packet which requires no connection method and is not concerned with delivery. UDP packets

addressed for destination port 0 could be a DOS attack. This source address could not be pinged and I was not able to traceroute to it. This could lead me to believe that the source address was spoofed. However further inspection of the packets shows a payload of (CKAAAAAAAAA) which is the used in a NetBIOS querrie. The packet payload contains a hex value of 0x41 padded. The wild card character "*" is two character hex represented by 2A hex when added to the 0x41 value would give us the C and K value. The remaing 0x41 hex is A. The generated alert gives the appearance that the packet is a malformed NetBIOS query. If that were to be a true NetBIOS querrie the sender would require a response, and would not spoof the source address. My conclusion is this address is probably not spoofed. A lookup of the IP address from www.arin.net[19] provides the following results.

```
          Verizon Internet Services VIS-151-196 (NET-151-196-0-0-1)
                                    151.196.0.0 - 151.205.255.255
Verizon Internet Services VZ-DSLDIAL-CYVLMD-6 (NET-151-196-181-0-1)
                                    151.196.181.0 - 151.196.189.255
```

2.2f Description of the attack

I believe that it is a malformed NetBIOS wild card scan looking for a NetBIOS name status request. In this example the scanning host sends a name request to the target host. If the target host accepts the request it would respond with its NetBIOS hostname, Windows workgroup or domain name, and users currently logged on. Traffic as such is usually seen on an internal network with Microsoft clients. This traffic however was generated from an external source and would more accurately be characterized as a reconnaissance scan.

2.2g Attack mechanism

Attacking host sends two UDP packets to a host on the network with a destination port 0. The length field in the UPD header is 137 bytes which is much larger than the total length of the IP datagram 78 bytes. Further inspection of the packets using tcpdump with the following syntax gave up more insight.

tcpdump -nXr 2002.10.4 src host 151.196.186.220 > filename

-n don't convert host address and port numbers to names

-X when printing hex print ASCII to

-r read from following file

src host filter on specific source host

> send output to filename

This produced the following tcpdump output.

09:53:48.086507 151.196.186.220.1026 > 207.166.191.242.0: [bad udp cksum b6b5!] udp 129 (ttl 112, id 60425, len 78, bad cksum c6a5!)

```
0x0000   4500 004e ec09 0000 7011 c6a5 97c4 badc        E..N....p.......
0x0010   cfa6 bff2 0402 0000 0089 003a 23b6 0100         ...........:#...
0x0020   0010 0001 0000 0000 0000 2043 4b41 4141         ...........CKAAA
0x0030   4141 4141 4141 4141 4141 4141 4141 4141        AAAAAAAAAAAAAAAA
0x0040   4141 4141 4141 4141 4141 4100 0021             AAAAAAAAAAA..!
```

09:53:51.146507 151.196.186.220.1026 > 207.166.191.7.0: [bad udp cksum b4b6!] udp 129 (ttl 112, id 19210, len 78, bad cksum 6792!)

```
0x0000   4500 004e 4b0a 0000 7011 6792 97c4 badc        E..NK...p.g.....
0x0010   cfa6 bf07 0402 0000 0089 003a 23a3 0100         ...........:#...
0x0020   0010 0001 0000 0000 0000 2043 4b41 4141         ...........CKAAA
0x0030   4141 4141 4141 4141 4141 4141 4141 4141        AAAAAAAAAAAAAAAA
0x0040   4141 4141 4141 4141 4141 4100 0021             AAAAAAAAAAA..!
```

Byte 3 offset from zero 0x4e has a value of 78 bytes decimal. While bytes 5 and 6 into the UDP header 0x00, 0x89 shows us a decimal value of 137, which coincidently is the NetBIOS name service port. Looking at the packet there appears to be two possibilities. One is that the packet was crafted and was being used for recognizance. I come to this conclusion by looking at the 3 and 4 byte into the UDP header. If these bytes were not injected and all other bytes following were to move two places to the left. This would give us a destination port of 137 and a total UPD header length of 58. If we were to take the IP header length from the zero offset byte which is a decimal 20 and add it to this new UDP header length the sum is correct with the total length of the datagram 78 bytes. My second theory is that this packet was mangled by a router or such device. Routers do not validate UDP check sums. UPD validation is done from end node to end node.

2.2h Correlations

The SMB wild card attacks have been reported in the wild dating back to almost four years. The following are links provide some insight.

http://www.sans.org/resources/idfaq/port_137.php

http://www.finchhaven.com/pages/incidents/030102_udp_137.html

http://www.digitaltrust.it/arachnids/IDS177/research.html

2.2i Evidence of targeting

The two Snort rules which this traffic tripped have the same destination address from the same source address.

2.2j Severity

(Criticality + lethality) – (system countermeasures + network countermeasures) =severity
(2 + 4) – (3 + 2) = 1

2.2k Criticality

The attack in this example was designed to do slow probe. However the packet was malformed. If the packet was normal it could have elicited a response from the targeted host. If that were to be true, criticality would have been a 2.

2.2l Lethality

The attack if successful would provide some information on mapping of the host along with domain name and user id. This could be used for a future attack.

2.2m System countermeasures

There were no responses to these packets.

2.2n Network countermeasures

I would deploy an ingress firewall and block inbound NetBIOS traffic along with possibly blocking the source host network address.

2.2o Defensive recommendations

I would ensure that all PC's on the internal network are hardened with all current patches and updates. I would deploy an ingress firewall and block inbound NetBIOS traffic. If that is not possible I would apply an access list to the Cisco router interface and block inbound NetBIOS that would break state.

2.2p Multiple choice question

The Snort_decoder *decodeUDPPkt* function is used to decode?

A) ICMP Packets

B) TCP Packets

C) UDP Packets

D) all of the above

Answer: is "C" The Snort_decoder *decodeUPDPkt* function decodes UDP packets.

2.2q References

1. Snort 2.0 Intrusion Detection Brian Caswell Technical Editor
2. Snort FAQ http://www.snort.org/docs/faq.html

3. Snort http://www.snort.org/

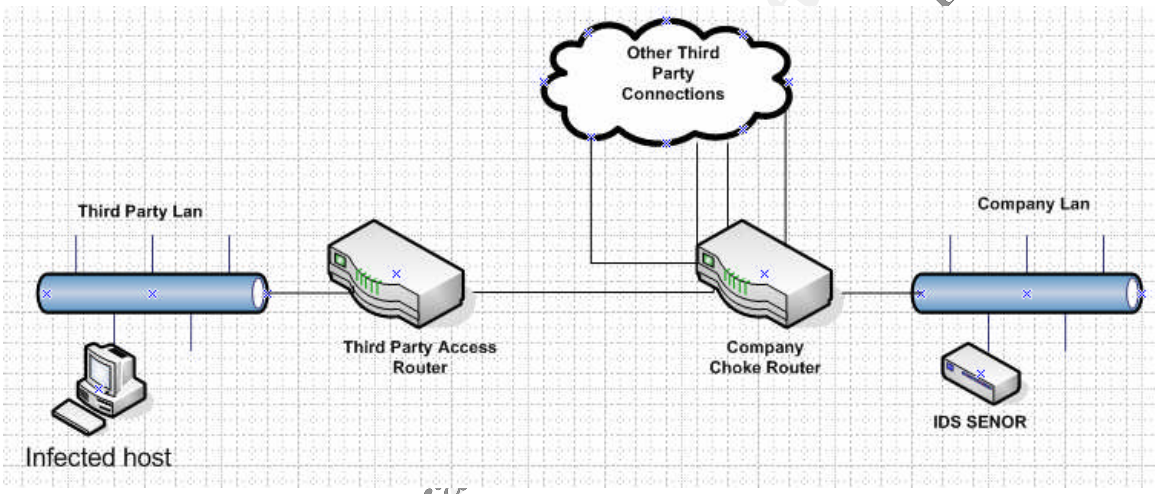
.

2.3 Detect #3 NETBIOS SMB-DS IPC$ share Unicode access NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt

2.3a Source of trace

This trace was captured on my company network. IP addressing will be obfuscated to hide the real IP addresses.

2.3a1 Network Diagram



2.3b Detect Generated by

This detect was generated by Snort version 2.1.2with a custom rules set Configuration. The following Snort syntax was used.

Snort –b –A fast -c <file locations>/snort.conf -i eth1 –D

-b log packet in tcpdump format (used for speed, much faster)

-A set alert mode, fast, full, console, or none

-c use rules file

-i listen on interface eth1

-D run snort in background (daemon mode)

2.3c Alerts generated by trace

06/12-22:13:16.394451 [**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} x.x.9.150.4:52430 -> x.x.209.248.209:445
06/12-22:13:16.755124 [**] [1:2514:2] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} x.x.9.150.4:52430 -> x.x.209.248.209:445
06/13-00:57:54.490384 [**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} x.x.9.150.4:53428 -> x.x.209.250.63:445
06/13-00:57:54.856683 [**] [1:2514:2] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} x.x.9.150.4:53428 -> x.x.209.250.63:445
06/13-01:589.693417 [**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} x.x.9.150.4:53955 -> x.x.209.248.209:445
06/13-01:58:19.105241 [**] [1:2514:2] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} x.x.9.150.4:53955 -> x.x.209.248.209:445
06/13-01:58:51.003511 [**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} x.x.9.150.4:53961 -> x.x.209.248.209:445
06/13-01:599.945204 [**] [1:2514:2] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} x.x.9.150.4:53961 -> x.x.209.248.209:445
06/13-02:23:37.524649 [**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} x.x.9.150.4:54097 -> x.x.209.249.149:445
06/13-02:23:37.884708 [**] [1:2514:2] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} x.x.9.150.4:54097 -> x.x.209.249.149:445
06/13-04:45:37.698094 [**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} x.x.9.150.4:56001 -> x.x.209.248.32:445
06/13-04:45:38.226271 [**] [1:2514:2] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} x.x.9.150.4:56001 -> x.x.209.248.32:445
06/13-055:47.693252 [**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} x.x.9.150.4:56110 -> x.x.209.250.30:445

06/13-055:48.053336 [**] [1:2514:2] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} x.x.9.150.4:56110 -> x.x.209.250.30:445
06/13-07:266.131528 [**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} x.x.9.150.4:57773 -> x.x.209.250.84:445
06/13-07:266.560255 [**] [1:2514:2] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} x.x.9.150.4:57773 -> x.x.209.250.84:445
06/13-09:53:39.606534 [**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} x.x.9.150.4:59225 -> x.x.209.249.149:445
06/13-09:53:40.000749 [**] [1:2514:2] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} x.x.9.150.4:59225 -> x.x.209.249.149:445
06/13-10:10:51.826070 [**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} x.x.9.150.4:59342 -> x.x.209.248.161:445
06/13-10:10:52.240953 [**] [1:2514:2] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} x.x.9.150.4:59342 -> x.x.209.248.161:445
06/13-10:53:17.838742 [**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} x.x.9.150.4:59905 -> x.x.209.249.125:445
06/13-10:53:18.282154 [**] [1:2514:2] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} x.x.9.150.4:59905 -> x.x.209.249.125:445
06/13-10:58:21.614576 [**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} x.x..150.4:59946 -> x.x.209.249.210:445
06/13-10:58:22.003544 [**] [1:2514:2] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} x.x.9.150.4:59946 -> x.x.209.249.210:445
06/13-17:10:41.634451 [**] [1:2466:1] NETBIOS SMB-DS IPC$ share unicode access [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} x.x.9.150.4:64557 -> x.x.209.249.196:445
06/13-17:10:42.072199 [**] [1:2514:2] NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} x.x.9.150.4:64557 -> x.x.209.249.196:445

2.3d Rules that generated trace

alert tcp $EXTERNAL_NET any -> $HOME_NET 445 (msg:"NETBIOS SMB-DS IPC$ share unicode access"; flow:to_server,established; content:"|00|"; depth:1; content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,>,127,6,relative; content:"I|00|P|00|C|00 24 00 00|"; distance:32; classtype:protocol-command-decode; nocase; sid:2466; rev:)

alert tcp $EXTERNAL_NET any -> $HOME_NET 445 (msg:"NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt"; flow:to_server,established; content:"|FF|SMB|2F|"; nocase; offset:4; depth:5; content:"|05|"; content:"|00|"; distance:1; within:1; content:"|09 00|"; distance:19; within:2; flowbits:isset,netbios.lsass.bind.attempt; reference:cve,CAN-2003-0533; reference:url,www.microsoft.com/technet/security/bulletin/MS04-011.mspx; classtype:attempted-admin; sid:2514; rev:2;)

Syntax for rule
2.3e Rule1 Header

Alert – *output format*> <**tcp**- *proctocol being used*> <**$EXTERNAL_NET** –*variable for External networks*> <**any** –*source port*> <**->** -*conversation dircection*> <**$HOME_NET** –*variable for defined the internal network*> <**445** –*destination port*>

2.3e1 Rule2 Header

Alert – *output format*> <**tcp**- *proctocol being used*> <**$EXTERNAL_NET** –*variable for External networks*> <**any** –*source port*> <**->** -*conversation dircection*> <**$HOME_NET** –*variable for defined the internal network*> <**445** –*destination port*>

2.3f Rule Options

<(**msg:"NETBIOS SMB_DS IPC$ share unicode access"** –*message displayed by alert*> <**flow: to server established** –*flow control option activating on packets that are part of the established tpc session*> < **content:"|00|"; depth:1; content:"|FF|SMB|75|"; offset:4; depth:5; byte_test:1,>,127,6,relative; content:"I|00|P|00|C|00 24 00 00|"; distance:32**; *if packet is matched against the rule tree node, Snort will take the following content and will try to match it against the packet using the Boyer-Moore search algorithm*> <**classtype**:**protocol-command-decode** classification of the attack>; nocase; <**sid:2466** snort rule unique identifier,> < **rev**:version number for the rule>)

2.3f1 Rule Options

<(**msg:"NETBIOS SMB-DS DCERPC LSASS DsRolerUpgradeDownlevelServer exploit attempt**"; *message displayed by alert*> <**flow:to_server,established**; *flow control option activating on packets that are already part of a established tcp session*>

<**content:"|FF|SMB|2F|"; nocase; offset:4; depth:5; content:"|05|"; content:"|00|"; distance:1; within:1; content:"|09 00|"; distance:19; within:2;** *if packet is matched against the rule tree node, Snort will take the following content and will try to match it against the packet using the Boyer-Moore search algorithm*>
<**flowbits:isset,netbios.lsass.bind.attempt**; **reference:cve,CAN-2003-0533**; *CVE reference number*>reference:url,www.microsoft.com/technet/security/bulletin/MS04-011.mspx; classtype:attempted-admin; <**sid:2514; rev:2**;*snort rule identifier and rev rule number*>)

2.3g Possibility the source address was spoofed

The probability of the source address being spoofed is very low. This connection is to a third party business connection sitting behind a choke router. The source address is from a valid address allowed to traverse the router and access the network. The address was sent ICMP echo request packets from the internal network, it in returned sent back echo-reply packets. System admin for the third party network was called and infected host was identified and verified.

2.3h Description of the attack

This attack is coming from a sasser virus infected host. This worm exploits the Windows LSASS vulnerability defined in MS04-11, which is a buffer overrun that allows remote code execution and enables an attacker to gain full control of the affected system. To propagate, it scans the network for vulnerable systems. When it finds a vulnerable system, this malware sends a specially crafted packet to produce a buffer overflow on LSASS.EXE**.** It sends the specially crafted packet to TCP port 445, a valid port used by Windows 2000 to transport SMB (Server Message Block) over TCP and UDP.

2.32i Attack mechanism

Let us first take a look at the some of the tcpdumps extracted from the date of the 13th. tcpdump syntax used was tcpdump -nnvvX -s 1514 -r <f*ilname*>

-nn

-vv very verbose

-s 1514 capture snap length

-r <filename> read from file

10:10:51.826070 x.x.150.4.59342 > x.x.248.161.445: P [tcp sum ok]
2076516558:2076516658(100) ack 4260870410 win 16
049 (DF) (ttl 117, id 30187, len 140)
0x0040 0000 0000 0000 fffe 0008 3000 04ff 005c .....0....\
0x0050 0008 0001 0035 0000 5c00 5c00 3xxx xxxx .....5..\.\.x.x.

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

```
0x0060   3700 2e00 3200 3000 3900 2e00 3200 3400        x...2.0.9...2.4.
0x0070   3800 2e00 3100 3600 3100 5c00 6900 7000        8...1.6.1.\.i.p.
0x0080   6300 2400 0000 3f3f 3f3f 3f00                  c.$...?????.
10:10:52.240953 x.x.150.4.59342 > x.x.248.161.445: . [tcp sum ok] 364:1744(1380) ack
328 win 15722 (DF) (ttl 117, id 30232, len 1420)
0x0000   4500 058c 7618 4000 7506 cdd2 xxxx 9604        E...v.@.u.....
0x0010   xxxx f8a1 e7ce 01bd 7bc5 223a fdf7 ba51        ........{.":...Q
0x0020   5010 3d6a fc40 0000 0000 10f8 ff53 4d42        P.=j.@.......SMB
0x0030   2f00 0000 0018 07c8 0000 0000 0000 0000        /...............
0x0040   0000 0000 0008 fffe 0008 6000 0eff 00de        ..........`.....
0x0050   de00 4000 0000 00ff ffff ff08 00b8 1000        ..@.............
0x0060   00b8 1040 0000 0000 00b9 10ee 0500 0001        ...@............
0x0070   1000 0000 b810 0000 0100 0000 0c20 0000        ................
0x0080   0000 0900 ad0d 0000 0000 0000 ad0d 0000        ................
0x0090   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x00a0   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x00b0   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x00c0   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x00d0   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x00e0   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x00f0   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x0100   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x0110   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x0120   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x0130   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x0140   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x0150   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x0160   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x0170   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x0180   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x0190   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x01a0   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x01b0   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x01c0   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x01d0   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x01e0   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x01f0   9000 9000 9000 9000 9000 9000 9000 9000        ................
0x0200   9000 9000 9000 9000 9000 9000 9000 9000        ................
22:13:16.394451 x.x.150.4.52430 > x.x.248.209.445: P [tcp sum ok]
2422330240:2422330340(100) ack 319318857 win 65024 (DF
) (ttl 116, id 40302, len 140)
0x0000   4500 008c 9d6e 4000 7406 ac4c xxxx 9604        E....n@.t..L....
0x0010   xxxx f8d1 ccce 01bd 9061 d380 1308 6b49        .........a....kI
0x0020   5018 fe00 17fc 0000 0000 0060 ff53 4d42        P.........`.SMB
0x0030   7500 0000 0018 07c8 0000 0000 0000 0000        u...............
0x0040   0000 0000 0000 fffe 0008 3000 04ff 005c        ..........0....\
```

```
0x0050   0008 0001 0035 0000 5c00 5c00 xxxx xxxx      .....5..\.\.x.x.
0x0060   3700 2e00 3200 3000 3900 2e00 3200 3400      x...2.0.9...2.4.
0x0070   3800 2e00 3200 3000 3900 5c00 6900 7000      8...2.0.9.\.i.p.
0x0080   6300 2400 0000 3f3f 3f3f 3f00                c.$...?????.
22:13:16.755124 x.x.150.4.52430 > x.x.248.209.445: . [tcp sum ok] 364:1744(1380) ack
328 win 64697 (DF) (ttl 116, id 403
42, len 1420)
0x0000   4500 058c 9d96 4000 7406 a724 xxxx 9604      E.....@.t..$....
0x0010   xxxx f8d1 ccce 01bd 9061 d4ec 1308 6c90      .........a....l.
0x0020   5010 fcb9 c923 0000 0000 10f8 ff53 4d42      P....#.......SMB
0x0030   2f00 0000 0018 07c8 0000 0000 0000 0000      /...............
0x0040   0000 0000 0008 fffe 0008 6000 0eff 00de      ..........`.....
0x0050   de00 4000 0000 00ff ffff ff08 00b8 1000      ..@.............
0x0060   00b8 1040 0000 0000 00b9 10ee 0500 0001      ...@............
0x0070   1000 0000 b810 0000 0100 0000 0c20 0000      .................
0x0080   0000 0900 ad0d 0000 0000 0000 ad0d 0000      .................
0x0090   9000 9000 9000 9000 9000 9000 9000 9000      .................
0x00a0   9000 9000 9000 9000 9000 9000 9000 9000      .................
0x00b0   9000 9000 9000 9000 9000 9000 9000 9000      .................
0x00c0   9000 9000 9000 9000 9000 9000 9000 9000      .................
0x00d0   9000 9000 9000 9000 9000 9000 9000 9000      .................
0x00e0   9000 9000 9000 9000 9000 9000 9000 9000      .................
0x00f0   9000 9000 9000 9000 9000 9000 9000 9000      .................
0x0100   9000 9000 9000 9000 9000 9000 9000 9000      .................
0x0110   9000 9000 9000 9000 9000 9000 9000 9000      .................
0x0120   9000 9000 9000 9000 9000 9000 9000 9000      .................
0x0130   9000 9000 9000 9000 9000 9000 9000 9000      .................
0x0140   9000 9000 9000 9000 9000 9000 9000 9000      .................
0x0150   9000 9000 9000 9000 9000 9000 9000 9000      .................
```

The infected host is sending a TCP SYN packet to the Windows Service Messenger block port of 445. The receiving host if listening on this port will send back a TCP SYN ACK packet. The packet dump above show us how this virus will use the noop (no operations) sled to perform the buffer overrun on the LSASS service. The noop sled is represented by the hex values of 0x90 00 in the packet. If the host is vulnerable and unpatched to this attack, a reverse command.exe or reverse VNC could be obtained on this host. The rule is being triggered when the completing ACK is sent from the source host to complete the TCP three way handshake, this packet is the one which contains the vulnerability payload. Content with in the packet is tripping signature rule which is firing off alert. The following content is what is tripping rule: **content:"|09 00|"; here is the noop content along with, content:"|FF|SMB|75**. We also see the destination port of **445**. This port is used by Microsoft Windows W2k, Windows XP, and Windows server 2003 for SMB (service messenger block) communication over TCP. In this example the destination hosts SYN ACK and allowed a TCP session to be established.

2.3j Correlations

CVE for sasser virus

Name	CAN-2003-0533 (under review)
Description	Stack-based buffer overflow in certain Active Directory service functions in LSASRV.DLL of the Local Security Authority Subsystem Service (LSASS) in Microsoft Windows NT 4.0 SP6a, 2000 SP2 through SP4, XP SP1, Server 2003, NetMeeting, Windows 98, and Windows ME, allows remote attackers to execute arbitrary code via a packet that causes the DsRolerUpgradeDownlevelServer function to create long debug entries for the DCPROMO.LOG log file, as exploited by the Sasser worm.
References	• FULLDISC:20040413 EEYE: Windows Local Security Authority Service Remote Buffer Overflow • URL:http://lists.netsys.com/pipermail/full-disclosure/2004-April/020069.html • EEYE:AD20040413C • URL:http://www.eeye.com/html/Research/Advisories/AD20040413C.html • BUGTRAQ:20040429 MS04011 Lsasrv.dll RPC buffer overflow remote exploit (PoC) • URL:http://marc.theaimsgroup.com/?l=bugtraq&m=108325860431471&w=2 • MS:MS04-011 • URL:http://www.microsoft.com/technet/security/bulletin/ms04-011.mspx • CERT:TA04-104A • URL:http://www.us-cert.gov/cas/techalerts/TA04-104A.html • CERT-VN:VU#753212 • URL:http://www.kb.cert.org/vuls/id/753212
Phase	Assigned (20030708)
Votes	
Comments	

The following links provide insight into the sasser virus.

http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_SASSER.A
http://www.f-secure.com/v-descs/sasser.shtml

The next link speaks to the vulnerability MS04-011 with in the operating system.

http://www.microsoft.com/technet/security/bulletin/MS04-011.mspx

2.3k Evidence of targeting

Due to the nature of this virus, and how it creates randomly generated destination addresses. I would not classify this as a targeted attack. The destination algorithm generates random network addresses.

2.3l Severity

Criticality + lethality) – (system countermeasures + network countermeasures) =severity
(2 + 3) – (4 + 2) = -1

2.3m Criticality

The hosts which responded to infected source were workstation PC's patched with the Microsoft patch MS04-11. They were also running current anti virus definitions. However, have they not been patched and updated they could have been subject to the LSASS vulnerability.

2.3n Lethality

This attack if successful would have increased traffic on the local area network. It may allow root access to the box as defined by MS04-11. If the targeted hosts had not been patched they would have generated network traffic on their subnet. The majority of this traffic would have been seen egressing the network.

2.3o System countermeasures

Responding hosts were running MS04-11 patch and latest anti virus definitions.

2.3p Network countermeasures

Port 445 is an allowed service on much of the network. I would look at the connection between the two parties and determine if communication thru this port is necessary.

2.3q Defensive recommendations

Ensure all systems are patched to MS04-11 to protect against this attack. Validate anti virus software is updated to latest definitions. Allow only needed services and communication access thru the router connection. Deploy a firewall device between the two connections. I would also review these happenings with the third party vendor and determine what will be the proper protocol if future instances were to happen, such as shutting the interface between companies.

2.3r Multiple choice question

The Sasser virus was written to exploit unpatched MS04-011 Microsoft Operating systems, the virus was looking to perform a buffer overflow on the _____ service?

A) awhost service

B) svchost service

C) lsass service

D) winlogon service

Answer : C the sasser was looking to exploit a buffer overflow in the LSASS service

References

http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0533
http://www.petri.co.il/what_is_port_445_in_w2kxp.htm

3 Assignment #3 Analyze this

3.1 Executive summary

I have been tasked with providing a security audit and traffic analysis for five days of logs files which belong to GIAC University. I went into this audit with the pretense this is an educational environment and is willing to provide its students and faculty with access to informational resources with very little boundaries. GIAC University has provided these logs in the form of data from a Snort intrusion style detection system. The rule base being applied appears to be fairly standard with the exception of a few custom rules. We were not given any specifics as it pertains to the physical layout or networking equipment that is currently being used at the University. The five days of traffic were broken up into three distinct files, those being Alert, Scans, and OOS (Out Of Spec) files. Due to the volume of traffic generated, only the top ten alerts from a count standpoint will be analyzed. This along with the top ten types of scans will be looked at. We will use the OOS file for any kind of correlation between itself and the fore mentioned Alert and Scan files.
My findings for the University show the use of P2P file sharing and gaming applications are running rampant on the network. Policy for the use of these applications definitely will have to be visited. One of the best defenses will be to educate internal population with the vulnerabilities associated with this type of behavior along with the legal ramifications. Further findings show the network needs to be tightened down from the inside and from the outside. Machines that will offer services publicly should have a security audit performed on them and patched to protect against any current known vulnerability for the service. Workstations on the internal network should all be updated to the latest security patch levels along with the latest anti-virus definitions. Services that are not required should be shutdown on as well. The type of traffic being allowed in and out of the network should be reviewed. Virus protection and updating will need to be visited. There was the discovery of virus infected hosts propagating on the internal network. When this is all accomplished the IDS boxes them selves should then be tuned to reflect this traffic, this will help with cutting down on some of the false positives.

3.2 File selection

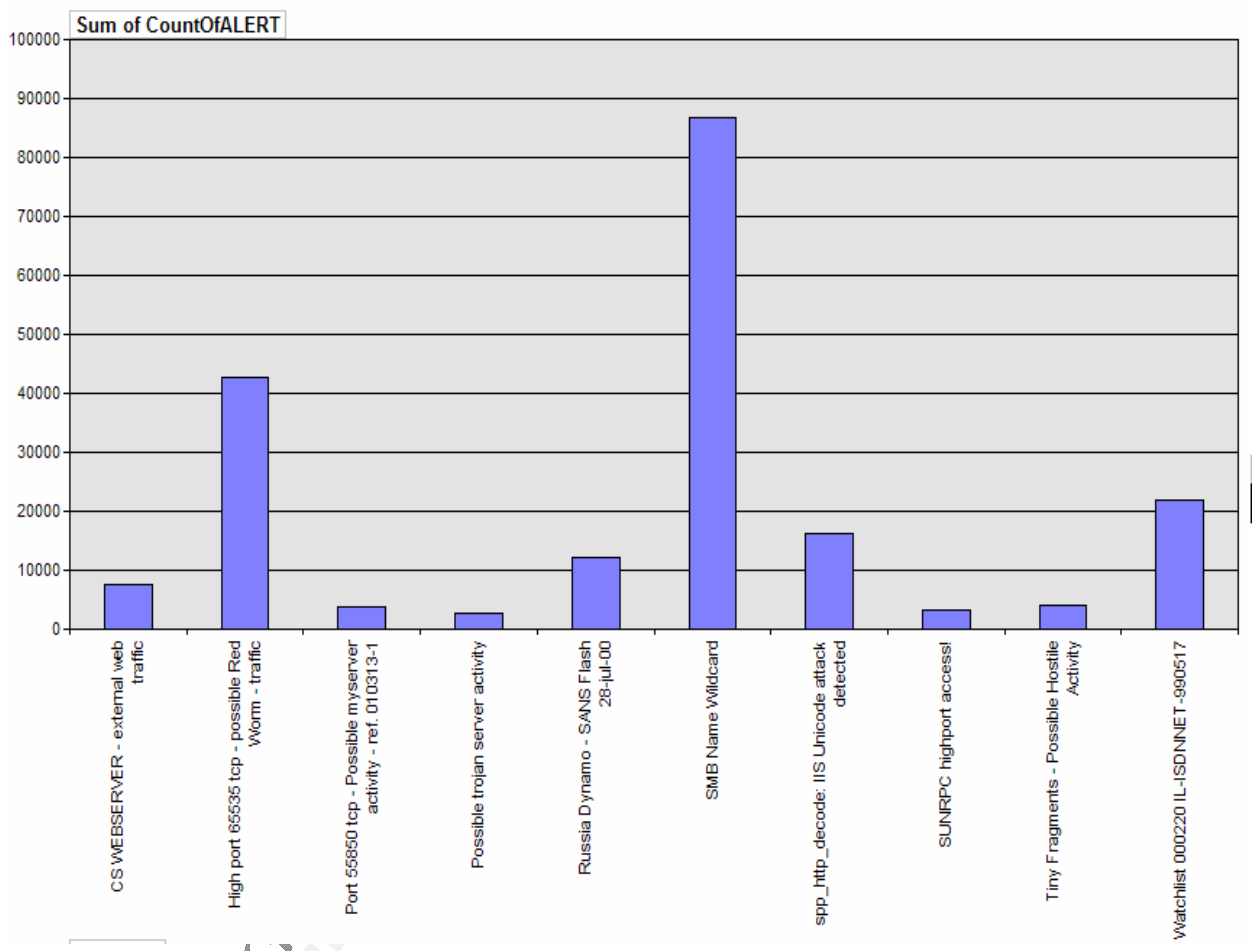
The files which were analyzed were dated from February 25, 2003 to March 1, 2003. These files were downloaded from `http://www.incidents.org/log`

Analysis was performed on the files listed below.

ALERT FILES	SCAN FILES	OOS
alert.030225.gz	scans.030225.gz	OOS_Report_2003_03_25_11706.txt
alert.030226.gz	scans.030226.gz	OOS_Report_2003_03_26_32018.txt
alert.030227.gz	scans.030227.gz	OOS_Report_2003_03_27_17540.txt
alert.030228.gz	scans.030228.gz	OOS_Report_2003_03_28_3648.txt
alert.030301.gz	scans.030301.gz	OOS_Report_2003_03_01_27482.txt

3.4 Alert log files

Following Graph will show the top ten alerts which were tripped during the five day's.



3.4a Top ten alerts

We will analyze the top ten alert attacks from the most frequent on down.

3.5 SMB Name Wild Card

This alert was tripped when hosts were probing on port **137** (NetBIOS). The rule was tripped 86652 times over the five day period. This most likely is an informational gathering probe; they are trying to access the system name table information. Individuals can obtain information which then can be used to launch an attack. Information available includes: The NetBIOS name of the server, Windows NT workgroup domain name and Login names of users who are logged into the server. The name of the administrator account if they are logged into the server. This probe can be automated; one can use scripts or programs written specifically to probe for open shares on a Windows computer, such as NBTscan[20], or Superscan from the Foundstone Corporation. An interesting link

was found which describes how multihomed PC's on the local LAN running Microsoft operating systems could generate spoofed packets.

http://lists.jammed.com/incidents/2001/05/0034.html

Another possible reason for the increase of this activity may be because of a "network.vbs worm". This worm is a visual basic script which infects windows hosts and tries to search for other candidate hosts on which to replicate. It issues these port 137 searches, tryin to enumerate shares and see if any are unprotected

As Matthew Fiddler writes in his practical, this rule should be tuned to allow for SMB name look ups on the local LAN and should be written to alert on just external addresses this will help in cutting down some of the noise. A misconfigured Samba server from an internal Linux system could be generating some false positives with this rule.

02/25-00:31:27.394139 [**] SMB Name Wildcard [**] 218.85.38.79:1030 -> MY.NET.180.138:**137**
02/25-00:19:33.406019 [**] SMB Name Wildcard [**] 210.55.255.34:1026 -> MY.NET.245.49:**137**
02/25-00:19:33.709498 [**] SMB Name Wildcard [**] 210.55.255.34:1026 -> MY.NET.245.51:**137**
02/25-00:19:33.722958 [**] SMB Name Wildcard [**] 24.84.59.206:1025 -> MY.NET.140.205:**137**
02/25-00:31:33.114134 [**] SMB Name Wildcard [**] 218.85.38.79:1030 -> MY.NET.180.252:**137**

3.5a Recommendations

Ensure that external users do not have access to Windows NetBIOS name service. This can be accomplished with a packet filtering device such as a router with an access control list applied to drop traffic destined for internal port 137. A firewall can also be used to block this type of traffic trying to ingress to the local network.

3.5b Correlations

Top three offenders by source address

CountOfSource	Source IP
961	207.6.57.6:137
236	67.83.29.116:137
134	24.202.194.180:137

None of the top offenders showed up in the OOS logs.

http://www.arin.net was used to provide network information from some of the top external source IP addresses

```
TELUS Communications Inc. TELUS-207-6-0-0 (NET-207-6-0-0-1)
                                  207.6.0.0 - 207.6.255.255
TELUS Communications Inc. HSIABC-207-6-32 (NET-207-6-32-0-1)
```

```
                                    207.6.32.0 - 207.6.63.255

Optimum Online (Cablevision Systems) NETBLK-OOL-4BLK (NET-67-80-0-0-1)
                                    67.80.0.0 - 67.87.255.255
Optimum Online (Cablevision Systems) OOL-67HCKNNJ5-0821 (NET-67-83-24-
0-1)
                                    67.83.24.0 - 67.83.31.255

Le Groupe Videotron Ltee VL-2BL (NET-24-200-0-0-1)
                                    24.200.0.0 - 24.203.255.255
Videotron Ltee VL-D-MS-18CA5E00 (NET-24-202-94-0-1)
                                    24.202.94.0 - 24.202.94.255

Jason Thompson make note of this detect in his CGIA practical.
```

3.6 High port scans 65535tcp –possible red worm traffic

This rule was triggered due to the source or destination host enumerating a port number of 65535.It tripped 42788 times during the five day audit period. This worm is also known as the Adore worm[21]. The Red worm is a Trojan that was first discovered in April of 2001 it is looking to make a connection to UPD or TCP port 65535. This worm will try to bind a Trojan back door to UPD port 65535 of the infected host. When activated it scans the Internet checking Linux hosts to determine whether they are vulnerable to any of the following well-known exploits: LPRng, rpc-statd, wu-ftpd and BIND. This alert is also prone to false positives; port 65535 may be selected as an emphemeral port for normal traffic.

02/25-02:30:45.469478 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.204.102:1995 -> 80.202.34.1
95:**65535**
02/25-02:30:45.836771 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.204.102:1995 -> 80.202.34.1
95:**65535**
02/25-02:30:46.221885 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.204.102:1995 -> 80.202.34.1
95:**65535**
02/25-02:30:46.604741 [**] High port 65535 tcp - possible Red Worm - traffic [**] 80.202.34.195:**65535** -> MY.NET.204.
102:1995

3.6a Recommendations

Ensure that all hosts that can be affected by this exposure are patched to protect against this vulnerability. You should also download the latest anti-virus definitions and keep them current. Three systems from the MY.NET network should be checked with a complete security audit immediately to see if they are infected due to the amount of the

traffic they are generating from port 65535. The three devices are: MY.NET83.205, MY.NET.88.193, and MY.NET.247.210

3.6b Correlations

Top five offenders by source address

CountOfSource IP	Source IP
12926	MY.NET.83.205:3186
12168	MY.NET.83.205:3793
7884	12.222.59.215:65535
1018	66.28.249.232:65535
806	MY.NET.88.193:65535
588	MY.NET.247.210:2315

Follow top offenders were found in the OOS logs.

02/28-19:42:26.475706 148.64.10.59:1025 -> **MY.NET.247.210:2315**
03/01-04:32:53.899455 148.64.4.130:1025 -> **MY.NET.247.210:2315**
03/04-20:48:16.127143 24.136.36.66:63442 -> **MY.NET.247.210:2553**
03/04-21:36:20.980887 24.208.247.138:4261 -> **MY.NET.247.210:2315**

http://www.arin.net was used to provide network information from some of the top external source IP addresses

```
CustName:   AT&T Worldnet Services
Address:    200 South Laurel Ave.
City:       Middletown
StateProv:  NJ
PostalCode: 07748
Country:    US
RegDate:    2003-11-26
Updated:    2003-11-26

NetRange:   12.122.0.0 - 12.123.255.255

OrgName:    Cogent Communications
OrgID:      COGC
Address:    1015 31st Street, NW
City:       Washington
StateProv:  DC
PostalCode: 20007
Country:    US

ReferralServer: rwhois://rwhois.cogentco.com:4321/

NetRange:   66.28.0.0 - 66.28.255.255
CIDR:     66.28.0.0/16
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Name	CVE-2000-0666
Description	rpc.statd in the nfs-utils package in various Linux distributions does not properly cleanse untrusted format strings, which allows remote attackers to gain root privileges.

3.7 Watchlist 000220 IL-ISDNNET-990517

The Watchlist 000220 IL-ISDNNET-990517 triggered 21876 alerts in the five day audit period. This alert is being used by the University to keep an eye on specific Network. Interesting is the majoriaty of destination ports into the my.net internal network are known P2P ports such as 3162, 1214.

http://www.arin.net was used to provide network information

```
inetnum:      212.179.13.0 - 212.179.13.255
netname:      CABLES-CONNECTION
descr:        CABLES-CONNECTION
country:      IL
```

3.7a Recommendation

Apparently the current security folks are monitoring this 212.179.0.0/16 network block. I would continue monitoring and add some auditing and tracking capabilities. A firewall can also be used to block access to and from this network if necessary.

Network being watched did not appear in the OOS logs.

3.8 spp_http_decode: IIS Unicode attack detected

The spp_http_decode: IIS Unicode attack Snort pre processor http decode was triggered 16221 times over the five day audit period. This attack is targeting a Microsoft IIS server. The Snort pre processor is triggered by the passing of file representation characters **'../'** or **'..\'** in Unicode. The goal is to traverse outside of the inetpub directory to compromise run remote commands or possibly gain root access to the server. This alert is subject to false positives.

```
02/26-04:49:05.491629  [**] spp_http_decode: IIS Unicode attack detected [**]
211.95.166.194:4467 -> MY.NET.193.206:80
02/26-04:49:05.491629  [**] spp_http_decode: IIS Unicode attack detected [**]
211.95.166.194:4467 -> MY.NET.193.206:80
02/26-04:49:05.491629  [**] spp_http_decode: IIS Unicode attack detected [**]
211.95.166.194:4467 -> MY.NET.193.206:80
```

02/26-04:50:03.070380 [**] spp_http_decode: IIS Unicode attack detected [**] 62.233.208.2:4558 -> MY.NET.218.26:80
02/26-04:51:15.285162 [**] spp_http_decode: IIS Unicode attack detected [**] 61.182.207.64:61245 -> MY.NET.197.193:80
02/26-04:51:33.828757 [**] spp_http_decode: IIS Unicode attack detected [**] 80.48.248.67:64661 -> MY.NET.218.26:80
02/26-05:16:18.540772 [**] spp_http_decode: IIS Unicode attack detected [**] 216.199.132.100:1032 -> MY.NET.252.133:80
02/26-05:16:18.540772 [**] spp_http_decode: IIS Unicode attack detected [**] 216.199.132.100:1032 -> MY.NET.252.133:80
02/26-05:16:18.540772 [**] spp_http_decode: IIS Unicode attack detected [**] 216.199.132.100:1032 -> MY.NET.252.133:80
02/26-05:16:44.232347 [**] spp_http_decode: IIS Unicode attack detected [**] 211.95.166.194:4489 -> MY.NET.195.204:80
02/26-05:18:26.057554 [**] spp_http_decode: IIS Unicode attack detected [**] 61.182.207.64:61722 -> MY.NET.253.7:80

3.8a Recommendations

The Snort pre processor http decode alerted on this signature. This preprocessor does not know the configuration of the IIS server and is independently configured from the web server. This rule may be subject to many false positives. I recommend the all University IIS web servers have a security audit performed on them and they be updated to the latest security patches.

3.8b Correlations

Name	CVE-2000-0884
Description	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.

None of the top source offenders were in the OOS logs.

3.9 Russia Dynamo – SANS Flash 28

The Russia Dynamo – SANS Flash 28 alert was triggered 12267 times over the five day audit period. The rule is tripped due to the source or destination address being from a possible unscrupulous Russian network. Readings on this particular attack has found that data at times has been sent illegitimately to this Russian network. I would look closely at the hosts which are connecting over port 2000 this port has been known to be used by the commercial remote control program "RemotelyAnywhere"[22]

03/01-05:48:41.863720 | Russia Dynamo - SANS Flash 28-jul-00 | **194.87.6.86**:4713 | MY.NET.105.204:**2000**
03/01-05:38:21.927168 | Russia Dynamo - SANS Flash 28-jul-00 | **194.87.6.86**:4713 | MY.NET.105.204:2000
03/01-05:38:28.860663 | Russia Dynamo - SANS Flash 28-jul-00 | **194.87.6.86**:4713 | MY.NET.105.204:**2000**
03/01-05:38:32.275201 | Russia Dynamo - SANS Flash 28-jul-00 | **194.87.6.86**:4713 | MY.NET.105.204:2000
03/01-05:38:40.297698 | Russia Dynamo - SANS Flash 28-jul-00 | MY.NET.105.204:**2000** | **194.87.6.86**:4713
03/01-05:38:40.761168 | Russia Dynamo - SANS Flash 28-jul-00 | **194.87.6.86**:4713 | MY.NET.105.204:2000
03/01-05:38:41.131386 | Russia Dynamo - SANS Flash 28-jul-00 | MY.NET.105.204:**2000** | **194.87.6.86**:4713
03/01-05:38:41.466856 | Russia Dynamo - SANS Flash 28-jul-00 | MY.NET.105.204:2000 | **194.87.6.86**:4713
03/01-05:38:43.567482 | Russia Dynamo - SANS Flash 28-jul-00 | **194.87.6.86**:4713 | MY.NET.105.204:2000
03/01-05:38:43.567631 | Russia Dynamo - SANS Flash 28-jul-00 | MY.NET.105.204:2000 | **194.87.6.86**:4713

3.9a Recommendations

Block all traffic to and from the 194.87.6.0/24 network and perform a security audit on host MY.NET.105.204 for possible Trojan activity, patch and update its anti-virus definitions. In the future I would look for Trojan activity on any host which is sending outbound traffic to this network.

Russian network did not appear in the OOS logs.

3.9b Correlations

```
inetnum:        194.87.6.0 - 194.87.6.255
netname:        DEMOS-DOL-DIALUP
descr:          DEMOS-Online Dialup
descr:          Demos-Internet Co.
descr:          Moscow, Russia
country:        RU
admin-c:        DNOC-ORG
tech-c:         DNOC-ORG
status:         ASSIGNED PA
```

3.10 CS WEBSERVER - external web traffic

The CS WEBSERVER - external web traffic rule was triggered 7581 times over the five day audit period. With out looking at the current Snort rule base from University it is difficult to determine exactly what this rule is being used for. I tried but could not find

any relevant correlations with this rule other than some knowledge that "**CS**" computer science is the universal symbol used in the .edu world by computer science departments. I suspect this is a custom rule used to alert on HTTP traffic from a external source to the web server MY.NET.100.165.

02/25-00:18:24.122638 | CS WEBSERVER - external web traffic | 66.126.94.28:53580 | **MY.NET.100.165**:80
02/25-00:18:28.424104 | CS WEBSERVER - external web traffic | 66.77.73.144:2976 | **MY.NET.100.165**:80
02/25-00:18:51.540371 | CS WEBSERVER - external web traffic | 66.77.73.144:3866 | **MY.NET.100.165**:80
02/25-00:30:59.985110 | CS WEBSERVER - external web traffic | 219.101.183.7:47255 | **MY.NET.100.165**:80
02/25-00:31:39.461938 | CS WEBSERVER - external web traffic | 128.195.180.79:1981 | **MY.NET.100.165**:80
02/25-00:19:46.893829 | CS WEBSERVER - external web traffic | 66.27.203.39:1731 | **MY.NET.100.165**:80
02/25-00:19:54.408563 | CS WEBSERVER - external web traffic | 210.212.215.41:2903 | **MY.NET.100.165**:80
02/25-00:31:59.090381 | CS WEBSERVER - external web traffic | 66.196.72.78:38293 | **MY.NET.100.165**:80
02/25-00:33:21.757831 | CS WEBSERVER - external web traffic | 210.154.148.3:49675 | **MY.NET.100.165**:80
02/25-00:33:27.271126 | CS WEBSERVER - external web traffic | 66.196.72.50:50738 | **MY.NET.100.165**:80
02/25-00:34:51.975382 | CS WEBSERVER - external web traffic | 66.196.72.14:18234 | **MY.NET.100.165**:80
02/25-00:46:51.803712 | CS WEBSERVER - external web traffic | 66.196.72.16:44100 | **MY.NET.100.165**:80

3.10a Recommendations

Leave rule as is. I would also provide some form of auditing and logging on the MY.NET.100.165 web server if this is not already in place do due some correlations if necessary. I would perform security audit on the MY.NET.100.165 box to ensure box has all the latest updates and patches.

None of the top source addresses or the MY.NET.100.65 web server showed up in the OOS logs.

3.11 Tiny Fragments - Possible Hostile Activity

Tiny Fragments - Possible Hostile Activity rule was triggered 4183 times over the five day audit period. The value for this rule is configured in options part of the Snort rule. Minfrag sets a minimum size threshold for a fragmented packet, generally used to set up a limit for the minimum fragment size that is accepted on the network. This rule is being

tripped because the defined thresh hold (I do not know what this threshold is configured at) is not being met. Possible reason for fragment of packets may be to try and bi pass IDS systems or local log systems which do not do recombine packet.

Top source addresses

Count	Source IP
1543	MY.NET.246.54
307	MY.NET.246.54
178	MY.NET.246.54
158	12.217.141.163
149	MY.NET.246.54

Top destination addresses

Count	DEST IP
1543	207.157.103.50
307	68.162.56.175
178	149.68.58.106
158	MY.NET.196.69
149	208.239.76.97

The following destination address appeared in the OOS logs. Note the destination port 1214 this is a known port used by P2P applications.

03/04-10:26:50.994009 66.187.105.13:4086 -> **MY.NET.196.69**:1214
03/04-10:30:03.005117 66.187.105.13:4086 -> **MY.NET.196.69**:1214
03/04-10:53:06.093179 66.187.105.13:4167 -> **MY.NET.196.69**:1214
03/04-10:59:30.113097 66.187.105.13:4167 -> **MY.NET.196.69**:1214
03/04-11:02:42.125893 66.187.105.13:4167 -> **MY.NET.196.69**:1214
03/04-11:12:56.165569 66.187.105.13:4210 -> **MY.NET.196.69**:1214
03/04-11:18:16.191391 66.187.105.13:4210 -> **MY.NET.196.69**:1214
03/04-11:19:49.690701 66.187.105.13:4218 -> **MY.NET.196.69**:1214
03/04-11:21:28.197826 66.187.105.13:4210 -> **MY.NET.196.69**:1214
03/04-13:55:19.309529 66.187.105.13:4595 -> **MY.NET.196.69**:1214
03/04-13:59:35.324204 66.187.105.13:4595 -> **MY.NET.196.69**:1214

3.11a Recommendations

As Donald Parker states in his CGIA practical it is best to drop this traffic at the earliest point of entry, this being the border router. Your firewall should also be tuned to drop this type of traffic.

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

The rule Port 55850 tcp - Possible myserver activity - ref. 010313-1 was triggered 3902 times during the five day audit period. This rule was tripped due the source or destination port being 55850. Some of the information ascertained about MyServer exposure is it a little known DDOS agent which binds to UDP port 55850 and will install a rootkit[23]. However in some cases were the rule was tripped I believe it was from using the Kazaa[24] P2P file sharing application and using 55850 as the ephemeral port. I come to this conclusion by the matching communication port of 1214. The 1214 port is a known port

used by Kazaa for their file transfers[25]. One should note this rule is subject to false positives, port 55850 can be selected as and emphemeral port.

3.12a Recommendations

At a minimum block ingress traffic into the network using port 55850. I would also review or implement policy in regards to the allowing of P2P traffic. Finally consider blocking port 55850 egress traffic.

Top five offenders

CountOfSOURC	SOURCE IP	SRC PORT	DEST IP	DST POR
967	MY.NET.201.66	1214	80.129.80.137	55850
787	65.79.79.242	55850	MY.NET.240.186	5190
764	MY.NET.238.30	6881	199.201.151.15	55850
359	80.129.80.137	55850	MY.NET.201.66	1214
267	MY.NET.204.102	1995	80.50.63.162	55850

The following top offender who also showed up in the OOS logs.

03/02-04:13:05.479216 62.248.146.123:61 -> **MY.NET.204.102**:2166

http://www.arin.net was used to provide network information from some of the top external source IP addresses

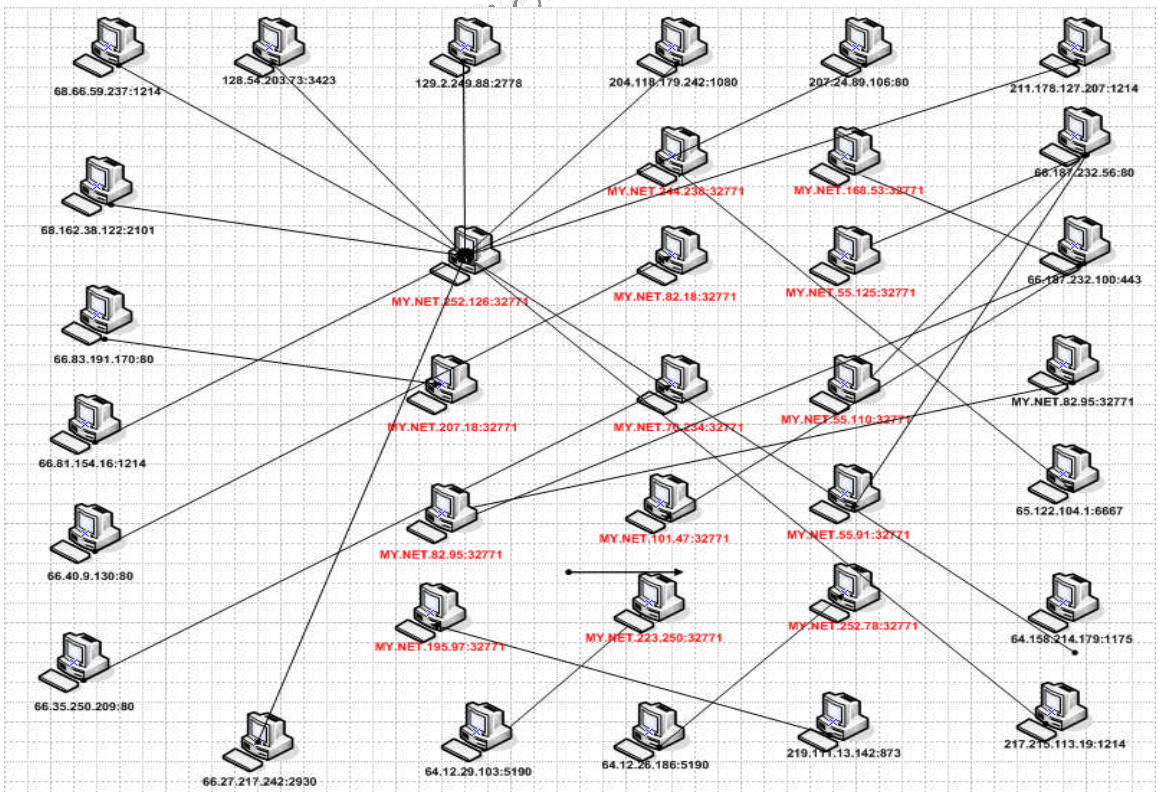
```
OrgName:    Illinois Century Network
OrgID:      ILTN
Address:    120 W Jefferson
Address:    Suite B
City:       Springfield
StateProv:  IL
PostalCode: 62702
Country:    US

NetRange:   65.79.0.0 - 65.79.127.255
CIDR:       65.79.0.0/17

inetnum:      80.128.0.0 - 80.146.159.255
netname:      DTAG-DIAL16
descr:        Deutsche Telekom AG
country:      DE
```

3.13 SUNRPC highport access!

The SUNRPC highport access rule was triggered 3366 times over the five day audit period. This rule is tripped when a query is sent and the destination port is 32771. This attack is an informational gathering attempt. It is targeting a Solaris machine and is querying the rpcbind/portmap daemon for port information for the rpc services[26]. If successful the attacker can obtain port services the host is offering.

02/26-16:01:24.635482 [**] SUNRPC highport access! [**] 68.162.38.122:2101 -> MY.NET.252.126:**32771**
02/26-16:01:25.043079 [**] SUNRPC highport access! [**] 68.162.38.122:2101 -> MY.NET.252.126:**32771**
02/26-16:01:25.474376 [**] SUNRPC highport access! [**] 68.162.38.122:2101 -> MY.NET.252.126:**32771**
02/26-16:01:26.239933 [**] SUNRPC highport access! [**] 68.162.38.122:2101 -> MY.NET.252.126:**32771**
02/26-16:01:26.321478 [**] SUNRPC highport access! [**] 68.162.38.122:2101 -> MY.NET.252.126:**32771**
02/26-15:50:15.936182 [**] SUNRPC highport access! [**] 68.162.38.122:2101 -> MY.NET.252.126:**32771**
02/26-15:50:16.024726 [**] SUNRPC highport access! [**] 68.162.38.122:2101 -> MY.NET.252.126:**32771**
02/26-15:50:17.378491 [**] SUNRPC highport access! [**] 68.162.38.122:2101 -> MY.NET.252.126:**32771**

This alert may have generated false positives associated with it. Port such as 5190 is known to be used by AOL instant messager service for file downloads. These connections may have chosen port 32771 as an emphemeral port. This would also apply with with port 80,443 and port 1214 P2P connections. The following link graph show us connections using the port 32771 and other service ports. Hosts using non standard server ports should be examined for possible compromise.



© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

02/25-02:00:44.248370 [**] SUNRPC highport access! [**] 64.12.26.116:5190 -> MY.NET.236.218:**32771**
02/25-12:32:53.328498 [**] SUNRPC highport access! [**] 64.12.26.116:5190 -> MY.NET.236.218:**32771**
02/25-12:45:27.341218 [**] SUNRPC highport access! [**] 64.12.26.116:5190 -> MY.NET.236.218:**32771**
02/25-15:01:19.161748 [**] SUNRPC highport access! [**] 64.12.163.72:5190 -> MY.NET.168.241:**32771**
02/25-14:51:15.888318 [**] SUNRPC highport access! [**] 64.12.163.72:5190 -> MY.NET.168.241:**32771**
02/25-18:50:11.229626 [**] SUNRPC highport access! [**] 66.187.232.56:80 -> MY.NET.55.110:**32771**
02/25-18:50:11.230369 [**] SUNRPC highport access! [**] 66.187.232.56:80 -> MY.NET.55.110:**32771**
02/25-18:50:11.230478 [**] SUNRPC highport access! [**] 66.187.232.56:80 -> MY.NET.55.110:**32771**

3.13a Recommendations

If possible, turn off RPC services. If not, put a firewall in front of the system that blocks external access to these services. Examine Hosts MY.NET.252.126, and 244.238 for compromise.

3.13b Correlations

Top Five offenders by source address

CountOfSOURCE IP	SOURCE IP	SRC PORT	DEST IP	DST POR
712	68.162.38.122	2101	MY.NET.252.126	32771
674	204.118.179.242	1080	MY.NET.252.126	32771
323	219.111.13.142	873	MY.NET.195.97	32771
234	128.183.16.204	22	MY.NET.97.34	32771
198	65.122.104.1	6667	MY.NET.244.238	32771

None of the top offenders showed up in the OOS log.

CVE information

Name	CAN-1999-0632 (under review)
Description	The RPC portmapper service is running.

http://www.arin.net was used to provide network information for the top three source networks

```
Verizon Internet Services VIS-68-160 (NET-68-160-0-0-1)
                                  68.160.0.0 - 68.163.255.255
```

```
Verizon VZ-DSLDIAL-BSTNMA-24 (NET-68-162-32-0-1)
                                  68.162.32.0 - 68.162.63.0


Sprint SPRINT-BLKB (NET-204-117-0-0-1)
                                  204.117.0.0 - 204.120.255.255
Access Toledo,LTD. FON-343033036863696 (NET-204-118-176-0-1)
                                  204.118.176.0 - 204.118.191.255

inetnum:        219.96.0.0 - 219.127.255.255
netname:        JPNIC-NET-JP
descr:          Japan Network Information Center
country:        JP
```

3.14 Possible trojan server activity

The Possible Trojan server activity alert was triggered 2261 times over the five day audit period. This alert was triggered because of the use of service port 27374. This port has been known to be associated with the SubSeven Trojan[27]. If device is infective it will allow unauthorized remote access to itself. This Trojan was first discovered in May, of 1999.

02/28-00:15:15.586727 [**] Possible trojan server activity [**] MY.NET.233.182:2117 -> 198.242.81.198:**27374**
02/28-00:15:15.781925 [**] Possible trojan server activity [**] MY.NET.233.182:2117 -> 198.242.81.198:**27374**
02/28-00:15:15.845821 [**] Possible trojan server activity [**] MY.NET.233.182:2117 -> 198.242.81.198:**27374**
02/28-00:15:15.849578 [**] Possible trojan server activity [**] MY.NET.233.182:2117 -> 198.242.81.198**:27374**
02/28-00:15:15.850909 [**] Possible trojan server activity [**] MY.NET.233.182:2117 -> 198.242.81.198:**27374**
02/28-00:15:15.860450 [**] Possible trojan server activity [**] MY.NET.233.182:2117 -> 198.242.81.198:**27374**

Top offenders by source address

CountOfSOURCE IP	SOURCE IP	SRC PORT	DEST IP	DST POR
1640	MY.NET.236.246	6347	213.67.143.245	27374
575	213.67.143.245	27374	MY.NET.236.246	6347
121	MY.NET.253.106	3424	217.215.88.11	27374
24	MY.NET.233.182	2117	198.242.81.198	27374
17	217.215.88.11	27374	MY.NET.253.106	3424
10	198.242.81.198	27374	MY.NET.233.182	2117

Top offenders showing up in the OOS logs.

03/04-09:00:22.189344 24.102.41.22:62311 -> **MY.NET.236.246**:6347
03/04-10:30:57.958209 24.102.41.22:63103 -> **MY.NET.236.246**:6347
03/04-11:01:28.669323 24.102.41.22:61229 -> **MY.NET.236.246**:6347
03/04-12:02:00.739004 24.102.41.22:64678 -> **MY.NET.236.246**:6347
03/04-13:32:26.015175 24.102.41.22:62663 -> **MY.NET.236.246**:6347
03/04-14:32:32.447901 24.102.41.22:63944 -> **MY.NET.236.246**:6347
03/01-10:40:40.491907 217.230.220.16:55863 -> **MY.NET.233.1**82:2117
03/01-10:43:13.281946 217.230.220.16:56124 -> **MY.NET.233.182**:39078
03/01-10:45:22.852251 217.230.220.16:56134 -> **MY.NET.233.182**:2075
03/01-11:47:22.463952 217.224.223.124:56128 -> **MY.NET.233.182**:2117

3.14a Recommendations

The Subseven Trojan is known to infect the following Microsoft operating systems, Windows 95, Windows 98, Windows NT, Windows 2000, Windows XP, and Windows Me. I would run a security audit on MY.NET.236.246, MY.NET253.106 and MY.NET.233.182 for possible infections. I would ensure that all systems are updated to current patch levels along with the Anti-virus definitions

3.14b Correlations

I agree with Mark Faske in his GIAC pratical with his summation that some of the machines communicating on port 27374 are in fact infected with the Subseven Trojan or Ramen worm. These would be the hosts which were connecting on non server ports.

Name	CAN-1999-0660 (under review)
Description	A hacker utility, back door, or Trojan Horse is installed on a system, e.g. NetBus, Back Orifice, Rootkit, etc.
References	
Phase	Proposed (19990804)
Votes	ACCEPT(4) Northcutt, Wall, Baker, Hill NOOP(1) Christey
Comments	Christey> Add "back door" to description.

3.15 Scan Logs

The following is the information and analysis ascertained from the Scan logs. The chart below show us the top ten scan types along with the count over the five day audit period.

SCAN TYPE	COUNT
UDP	64298
SYN	27178
NULL	1509
NOACK	632
INVALIDACK	291
FIN	237
VECNA	187
XMAS	129
UNKNOWN	120
NMAPID	26
FULLXMAS	12

The use and possible purpose of each scan is as follows.

3.15a UPD Scan

Informational in nature, it should also be noted that UDP is a connectionless communication protocol, meaning the attacker may or may not get any response back from the UPD port being scanned. The attacker is listening for UDP port to return an error message. If the attacker receives a "ICMP Port Unreachable" message the port is closed.

Top Five Scanned Destination UDP ports.

PROTOCOL	DST PORT	Count
UDP	53	6647
UDP	137	3800
UDP	27005	3736
UDP	7674	2406
UDP	22321	2043

3.15b Port Descriptions

PORT 53 Used in DNS Queries, attacker is trolling for a DNS server, if found attacker would try to determine what type of DNS server and launch known vulnerabilities. Local DNS server is to querrie External DNS server both would use UDP port 53.

Example scan alerts

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Feb 28 16:51:47 130.85.1.4:64706 -> 192.52.178.30:53 UDP
Feb 28 16:54:51 130.85.1.4:64706 -> 192.52.178.30:53 UDP
Feb 28 16:55:01 130.85.1.4:64706 -> 192.52.178.30:53 UDP
Feb 28 16:55:05 130.85.1.4:64706 -> 192.52.178.30:53 UDP
Feb 28 17:16:52 130.85.1.200:64049 -> 192.52.178.30:53 UDP
Feb 28 17:16:56 130.85.1.200:64049 -> 192.52.178.30:53 UDP
Feb 28 17:16:59 130.85.1.200:64049 -> 192.52.178.30:53 UDP
Feb 28 17:17:00 130.85.1.200:64049 -> 192.52.178.30:53 UDP
Feb 28 17:17:02 130.85.1.200:64049 -> 192.52.178.30:53 UDP

Top source addresses port 53 scans.

Count	SRC IP	DST PORT	DST IP
180	130.85.1.4	53	192.55.83.30
173	130.85.1.4	53	192.52.178.30
137	130.85.1.3	53	192.52.178.30
120	130.85.1.200	53	192.5.6.30

3.15c Recommendations

Interesting traffic, check the role on the local network for hosts 130.85.1.4, 130.85.1.3, and 130.85.1.200. They appear to be the local DNS servers for the network, troubling that the source port at times does not change.UDP port 53 is associated with DNS name querries; this may be producing some false positives with this scan.I Would like to have the tcpdumps for this traffic to see if these are in fact legimant DNS querries. I would perform a security audit on the hosts listed above.

None of the network addresses from the table above show up in the OOS logs.

3.15d Correlations

```
OrgName:    VeriSign Global Registry Services
OrgID:      VGRS
Address:    21345 Ridgetop Circle
City:       Dulles
StateProv:  VA
PostalCode: 20166
Country:    US

NetRange:   192.55.83.0 - 192.55.83.255
CIDR:       192.55.83.0/24

OrgName:    VeriSign Global Registry Services
OrgID:      VGRS
Address:    21345 Ridgetop Circle
City:       Dulles
StateProv:  VA
```

```
PostalCode: 20166
Country:    US

NetRange:   192.52.178.0 - 192.52.178.255
```

PORT 137 Used in NetBIOS name service, attackers trolling for Windows networking services such as computer name, system name, file shares etc. urrently most inbound scre the result of a number of such as BugBear and Opaserv which exploit open file shares to propagate.

Top source addresses port 137 scans.

Count	SRC IP	DST PORT	PROTOCOL
735	130.85.97.109	137	UDP
666	130.85.98.43	137	UDP
492	130.85.97.225	137	UDP
193	130.85.97.64	137	UDP
181	130.85.98.12	137	UDP
173	130.85.97.96	137	UDP
152	130.85.98.54	137	UDP
127	130.85.97.15	137	UDP
114	130.85.97.150	137	UDP

Example scan alerts

Feb 25 09:06:59 130.85.97.64:1026 -> 61.30.117.160:137 UDP
Feb 25 09:06:59 130.85.97.64:1025 -> 165.220.141.149:137 UDP
Feb 25 09:06:59 130.85.97.64:1026 -> 61.30.117.162:137 UDP
Feb 25 09:06:59 130.85.97.64:1028 -> 62.118.21.194:137 UDP
Feb 25 09:07:00 130.85.97.64:1026 -> 61.30.117.165:137 UDP
Feb 25 09:07:00 130.85.97.64:1025 -> 165.220.141.161:137 UDP
Feb 25 09:07:00 130.85.97.64:1027 -> 22.214.164.129:137 UDP
Feb 25 09:07:00 130.85.97.64:1028 -> 62.118.21.195:137 UDP
Feb 25 09:07:00 130.85.97.64:1025 -> 165.220.141.175:137 UDP
Feb 25 09:07:01 130.85.97.64:1029 -> 22.245.160.254:137 UDP
Feb 25 09:07:02 130.85.97.64:1027 -> 22.214.164.145:137 UDP
Feb 25 09:07:02 130.85.97.64:1026 -> 61.30.117.182:137 UDP
Feb 25 09:07:02 130.85.97.64:1028 -> 62.118.21.213:137 UDP
Feb 25 09:07:02 130.85.97.64:1026 -> 61.30.117.184:137 UDP
Feb 25 09:07:03 130.85.97.64:1027 -> 22.214.164.148:137 UDP
Feb 25 09:07:03 130.85.97.64:1025 -> 165.220.141.214:137 UDP
Feb 25 09:07:03 130.85.97.64:1025 -> 165.220.141.218:137 UDP
Feb 25 09:07:03 130.85.97.64:1027 -> 22.214.164.150:137 UDP

3.15e Recommendations

Perform a security audit on boxes listed in the top port 137 scanners chart. They appear to be showing signs of possible virus infection. This conclusion is made due the source port of all scans being 1025-1032. Update systems to latest security patches and anti-virus definitions.

None of the top network addresses from the table above show up in the OOS logs.

3.15f Correlations

NetBIOS worms
"Starting in 1999, numerous NetBIOS worms have been seen. These include ExploreZip virus/worm, Network.VBS Visual Basic script, and the 911 worm (which also calls 911 out your modem). All of these worms will attempt connection to you machine. In late 2002, the ALEVRIUS worm is the source of many of these queries in order to find names to connect to your machine with[28]."

Name	CVE-2000-0979
Description	File and Print Sharing service in Windows 95, Windows 98, and Windows Me does not properly check the password for a file share, which allows remote attackers to bypass share access controls by sending a 1-byte password that matches the first character of the real password, aka the "Share Level Password" vulnerability.

3.15g PORT 27005 Used generally by a Half-Life game client

Top source addresses port 27005 scans.

Count	SRC IP	DST PORT	PROTOCOL
3715	130.85.87.44	27005	UDP
9	130.85.206.70	27005	UDP
7	130.85.203.18	27005	UDP
5	130.85.219.14	27005	UDP

Example Scan alert

Feb 25 21:16:12 130.85.87.44:27021 -> 68.39.49.114:27005 UDP
Feb 25 21:16:10 130.85.87.44:27021 -> 24.43.44.53:27005 UDP
Feb 25 21:16:12 130.85.87.44:27021 -> 209.205.178.3:27005 UDP
Feb 25 21:16:10 130.85.87.44:27021 -> 66.67.105.207:27005 UDP

© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

Feb 25 21:16:11 130.85.87.44:27021 -> 24.82.159.127:27005 UDP
Feb 25 21:16:11 130.85.87.44:27021 -> 65.95.47.34:27005 UDP
Feb 25 21:16:11 130.85.87.44:27021 -> 12.237.242.42:27005 UDP
Feb 25 21:16:12 130.85.87.44:27021 -> 68.81.50.22:27005 UDP
Feb 25 21:16:13 130.85.87.44:27021 -> 24.43.44.53:27005 UDP
Feb 25 21:16:14 130.85.87.44:27021 -> 68.81.50.22:27005 UDP
Feb 25 21:16:14 130.85.87.44:27021 -> 66.67.105.207:27005 UDP

3.15h Recommendations

Policy will need to be reviewed on the playing of on line games such as "Half – Life[29]". Install a firewall and block UDP port 27005 outbound and inbound UDP port 27021.

None of the top sources from the table above were found in the OOS logs.

PORT 7674 Possible use of a Korean file sharing program named Soribada[30].

Count	SRC IP	DST PORT	PROTOCOL
979	130.85.196.179	7674	UDP
612	130.85.242.250	7674	UDP
485	130.85.97.60	7674	UDP
152	130.85.209.174	7674	UDP

Example scan alert

Feb 25 18:34:27 130.85.209.174:7674 -> 61.84.244.122:7674 UDP
Feb 25 18:34:27 130.85.209.174:7674 -> 220.91.131.166:7674 UDP
Feb 25 18:34:27 130.85.209.174:7674 -> 211.104.212.174:7674 UDP
Feb 25 18:34:27 130.85.209.174:7674 -> 220.77.195.169:7674 UDP
Feb 25 18:34:27 130.85.209.174:7674 -> 211.204.131.89:7674 UDP
Feb 25 18:34:27 130.85.209.174:7674 -> 129.32.80.134:7674 UDP
Feb 25 18:34:27 130.85.209.174:7674 -> 24.84.56.54:7674 UDP
Feb 25 18:34:28 130.85.209.174:7674 -> 128.2.162.21:7674 UDP
Feb 25 18:34:28 130.85.209.174:7674 -> 211.231.37.135:7674 UDP
Feb 25 18:34:28 130.85.209.174:7674 -> 211.230.117.207:7674 UDP
Feb 25 18:34:28 130.85.209.174:7674 -> 61.83.195.1:7674 UDP
Feb 25 18:34:28 130.85.209.174:7674 -> 211.55.195.195:7674 UDP
Feb 25 18:34:28 130.85.209.174:7674 -> 203.239.75.152:7674 UDP
Feb 25 18:34:29 130.85.209.174:7674 -> 211.221.130.15:7674 UDP

3.15i Recommendations

Review policy on P2P networks and file sharing. Block UDP port 7674 inbound and outbound.

PORT 22321 possibility this is yet another port being used by the Korean file sharing program Soribada.

Top source addresses port 22321 scans

Count	SRC IP	DST PORT	PROTOCOL
477	130.85.217.102	22321	UDP
383	130.85.97.60	22321	UDP
363	130.85.209.174	22321	UDP
312	130.85.88.227	22321	UDP
266	130.85.196.179	22321	UDP
240	130.85.242.250	22321	UDP

Example scan alert

Feb 25 18:34:12 130.85.209.174:22321 -> 210.114.158.116:22321 UDP
Feb 25 18:34:12 130.85.209.174:22321 -> 218.237.123.73:22321 UDP
Feb 25 18:34:12 130.85.209.174:22321 -> 63.163.161.205:22321 UDP
Feb 25 18:34:12 130.85.209.174:22321 -> 211.229.88.191:22321 UDP
Feb 25 18:34:12 130.85.209.174:22321 -> 211.208.67.77:22321 UDP
Feb 25 18:34:12 130.85.209.174:22321 -> 202.30.253.10:22321 UDP
Feb 25 18:34:12 130.85.209.174:22321 -> 218.37.161.205:22321 UDP
Feb 25 18:34:12 130.85.209.174:22321 -> 203.240.187.181:22321 UDP
Feb 25 18:34:12 130.85.209.174:22321 -> 203.255.181.169:22321 UDP
Feb 25 18:34:12 130.85.209.174:22321 -> 220.74.135.222:22321 UDP
Feb 25 18:34:12 130.85.209.174:22321 -> 211.243.108.219:22321 UDP
Feb 25 18:34:12 130.85.209.174:22321 -> 210.111.15.61:22321 UDP
Feb 25 18:34:12 130.85.209.174:22321 -> 61.102.62.83:22321 UDP
Feb 25 18:34:12 130.85.209.174:22321 -> 211.253.124.18:22321 UDP

3.15i Recommendations

Review of P2P file sharing software policy. Install firewall and block UDP port 22321 outbound and inbound.

None of the addresses from the table above were in the OOS logs.

3.16 SYN Scan

SYN scan is an informational gathering which the attacker can use for reconnaissance. This scan type is often referred to as a “half-open” scan. These do not open up a full TCP connection. Attacker sends a packet with the syn flag set, if he receives a packet with the SYN and ACK flag set the port is listening, if received packet has the reset flag set the port is not listening.

Top syn scanned ports

Count	DST PORT	SCAN TYPE
6532	1433	SYN
5730	80	SYN
3844	25	SYN
2624	21	SYN
2194	135	SYN
748	443	SYN
722	445	SYN

Port 1433 Port is used for SQL server; attackers target the SQL server systems using the MSSQL Hello buffer overflow attack[31]. Scan is used to locate boxes offering this service.

Top source addresses scanning port 1433.

Count	DST PORT	SCAN TYPE	SRC IP
1810	1433	SYN	61.221.128.34
1385	1433	SYN	148.235.160.90
1290	1433	SYN	61.82.125.35
1289	1433	SYN	81.224.46.230
366	1433	SYN	193.41.229.180
294	1433	SYN	203.125.224.50
98	1433	SYN	194.97.233.185

3.16a Recommendations

Complete security audit should be done on any boxes offering SQL server service. The boxes should be patched to the latest security updates, along with updated anti-virus definitions. Review if service should be offered publicly, if not block inbound 1433 traffic. Contact site administrator from source ip networks. Block traffic from these networks at the firewall.

None of the offenders listed in the table above show up in the OOS logs.

3.16b Correlations

```
inetnum:        61.220.0.0 - 61.227.255.255
netname:        HINET
descr:          Data Communication Business Group, Chunghwa Telecom Co.,
Ltd.
descr:          Commerical ISP
descr:          21, Section 1, Hsin-Yi Road, Taipei,
descr:          Taipei 100, Taiwan, R.O.C.
country:        TW

inetnum:      148.235/16
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

```
status:      reallocated
owner:       Uninet S.A. de C.V.
ownerid:     MX-USCV4-LACNIC
responsible: Arturo Zaldivar Mendez
address:     Periferico Sur, 3190,
address:     01900 - Ciudad de México - DF
country:     MX

inetnum:      61.78.0.0 - 61.85.255.255
netname:      KORNET
descr:        KOREA TELECOM
descr:        KOREA TELECOM Internet Operating Center
country:      KR

role:         TeliaNet Registry
address:      TeliaSonera AB Networks
address:      Carrier & Networks
address:      Marbackagatan 11
address:      SE-123 86 Farsta
address:      Sweden
fax-no:       +46 8 6047006
```

Port 80 This port is used for web server traffic, attacker is scanning for web servers, if found they can attempt multiple know web server vulnerabilities dependant on what type of web server is being used. I.E. Apache, IIS,

Top source addresses scanning port 80

Count	DST PORT	SCAN TYPE	SRC IP
3527	80	SYN	195.25.165.42
1072	80	SYN	195.199.41.13
891	80	SYN	61.1.192.77
123	80	SYN	150.208.149.6

3.16c Recommendations

Complete security audit on all web servers. Update to latest relevant patches dependant on server type. Contact site administrators of source ip addresses. Block source ip addresses at perimeter with a firewall.

None of the offenders listed in the table above show up in the OOS logs.

3.16d Correlations

```
inetnum:      195.25.165.0 - 195.25.165.255
netname:      FR-ALLNET
descr:        ALLNET
descr:        1 rue Georges Claude
descr:        14120 Mondeville
country:      FR
```

```
organisation: ORG-EN3-RIPE
org-name:     Elender Net
org-type:     LIR
address:      Elender Uzleti Kommunikacio
address:      Vaci ut 141.
address:      H-1138 Budapest
address:      Hungary

netname:      BSNLNET
descr:        National Internet Backbone
descr:        Bharat Sanchar Nigam Limited
descr:        Sanchar Bhawan, 20, Ashoka Road, New Delhi-110001, India
country:      IN
```

Port 25 Port 25 is used by SMTP, the outgoing mail protocol. This port has been known to have been exploited by some well known viruses such as "MY DOOM"[32].

3.16e Recommendations

Not much activity was sourcing from the internal network space. There was one external host 220.114.0.175 which tripped the scan alert 3710 times. Contact site administrator from this network. Block this network space with a firewall.

3.16f Correlations

```
netname:      GWBN-CHONGQING-NET1
country:      CN
descr:        FOR GREAT WALL BROADBAND NETWORK SERVICE ACCESS IN
CHONGQING NET1
person:       JIAN MENG
nic-hdl:      JM108-AP
e-mail:       mengjian@gwbnnet.cn
address:      2nd Floor, Building A
address: #9 Donghuan Plaza, Dong Zhong Street
address:      East District, Beijing, China (100027)
```

Port 21 FTP service is commonly run over this port. If port is found to be listening, the attacker will try to exploit the service with a multitude of vulnerabilities associated with the service and port[33].

Top source addresses scanning port 21

Count	DST PORT	SCAN TYPE	SRC IP
977	21	SYN	80.14.189.231
848	21	SYN	218.236.175.130
609	21	SYN	166.114.114.2
151	21	SYN	133.87.193.240

3.16g Recommendations

Hardening of all boxes which will be offering the ftp service, use of firewall to block top external networks which are scanning. Contact remote networks system administrators.

None of the offenders listed in the table above showed up in the OOS log.

3.16h Correlations

```
role:         Wanadoo France Technical Role
address:      WANADOO FRANCE
address:      48 rue Camille Desmoulins
address:      92791 ISSY LES MOULINEAUX CEDEX 9
address:      FR

218.234.0.0 - 218.239.255.255
netname:      HANANET
descr:        Hanaro Telecom Co.
descr:        Kukje Electornics Cneter Bldg. 1445-3 Seocho-Dong Seocho-
Ku

inetnum:      166.114/16
status:       allocated
owner:        Red Bolivina de Comunicacion de Datos
ownerid:      BO-RBCD2-LACNIC
responsible:  Gerente Técnico
address:      Ayacucho Street. Third Floor. Vice Presidency, 308,
address:      4864 - La Paz - LP
country:      BO
```

Port 135 Microsoft RPC service runs on port 135. Buffer over flow vulnerabilities, as well as exploits such as the Nachi[34] or MSBlast worms have been known to use this port.

3.16i Recommendations

Security Audit should be performed on host 130.85.150.210 it is showing signs of possible infection. This host has generated 2092 of the 2194 scan alerts. Externally port 135 inbound should be blocked by a firewall.

Port 443 SSL service is generally run on this port, attacker looking for web server to exploit.

3.16.j Recommendations

Secure any boxes which will be offering port 443 service to updated security patch levels. Use firewall to block the top offending source ip address 211.34.146.1.

3.16k Correlations

```
inetnum:      211.33.0.0 - 211.36.223.255
```

```
netname:        KRNIC-KR
descr:          KRNIC
descr:          Korea Network Information Center
country:        KR
```

Port 445 Microsoft port used to SMB over TCP/IP, this enabled them to eliminate the use of NetBIOS to run SMB. Many exploits[35] such as "w32.korgo[36]", "w32.welchia".

Top source addresses scanning port 445

Count	DST PORT	SCAN TYPE	SRC IP
374	445	SYN	130.85.150.210
120	445	SYN	130.85.218.62
82	445	SYN	130.85.89.253
54	445	SYN	130.85.252.82

3.16l Recommendations

Run security audit and check for infections for hosts listed in graph above. Patch all systems vulnerable to this exposure to correct patch level. Update and keep anti-virus definitions current. Use firewall to block inbound connections to port 445.

None of the top offenders listed from the table above showed up in the OOS log.

3.16M Correlations

Name	CAN-2003-0533 (under review)
Description	Stack-based buffer overflow in certain Active Directory service functions in LSASRV.DLL of the Local Security Authority Subsystem Service (LSASS) in Microsoft Windows NT 4.0 SP6a, 2000 SP2 through SP4, XP SP1, Server 2003, NetMeeting, Windows 98, and Windows ME, allows remote attackers to execute arbitrary code via a packet that causes the DsRolerUpgradeDownlevelServer function to create long debug entries for the DCPROMO.LOG log file, as exploited by the Sasser worm.
References	• FULLDISC:20040413 EEYE: Windows Local Security Authority Service Remote Buffer Overflow • URL:http://lists.netsys.com/pipermail/full-disclosure/2004-April/020069.html • EEYE:AD20040413C • URL:http://www.eeye.com/html/Research/Advisories/AD20040413C.html • BUGTRAQ:20040429 MS04011 Lsasrv.dll RPC buffer overflow remote exploit (PoC) • URL:http://marc.theaimsgroup.com/?l=bugtraq&m=108325860431471&w=2 • MS:MS04-011 • URL:http://www.microsoft.com/technet/security/bulletin/ms04-011.mspx • CERT:TA04-104A • URL:http://www.us-cert.gov/cas/techalerts/TA04-104A.html • CERT-VN:VU#753212 • URL:http://www.kb.cert.org/vuls/id/753212 • OVAL:OVAL883 • URL:http://oval.mitre.org/oval/definitions/pseudo/OVAL883.html • OVAL:OVAL898 • URL:http://oval.mitre.org/oval/definitions/pseudo/OVAL898.html • OVAL:OVAL919 • URL:http://oval.mitre.org/oval/definitions/pseudo/OVAL919.html
Phase	Assigned (20030708)
Votes	
Comments	

3.17 Null Scan

Null scans will have all bits in the flag field set to zero. This scan is informational in nature. Scans of this type will be used by attackers to try and evade packet filters, firewalls and IDS systems watching for syn scans. The scan returns a packet with the reset flag set for closed ports. Opened ports should drop the packet.

3.17a Recommendations

Perform a security audit on host 130.85.246.54, this host tripped the alert for this scan 522 times by far the most of any source host. Use firewall to block traffic from external network 80.60.247.181.

Neither of the fore mentioned boxes appeared in the OOS log.

3.17b Correlations

```
oute:         80.60.0.0/15
ole:          Planet Technologies
address:       Stationsstraat 115
address:       P.O. box 1042
address:       3800 BA Amersfoort
address:       The Netherlands
```

3.18 NOACK, VENCA, INVALIDACK, XMAS, and FULLXMAS Scans

I grouped all these types of scans together. These are usually crafted packets whose flag field bits are set to abnormal combinations such as RST, FIN, PUSH or ACK, FIN, URG.

3.18a Recommendations

Find out what local host 130.85.246.54 is up to. This host tripped the most alerts for all the scan types listed above. This host was always targeting the same destination host of 207.157.103.50. Externally the top attacker was coming from source address of 80.60.247.181; if possible contact system admin from this network. Use firewall to block this address.

None of the fore mentioned network addresses were found in the OOS log.

3.18b Correlations

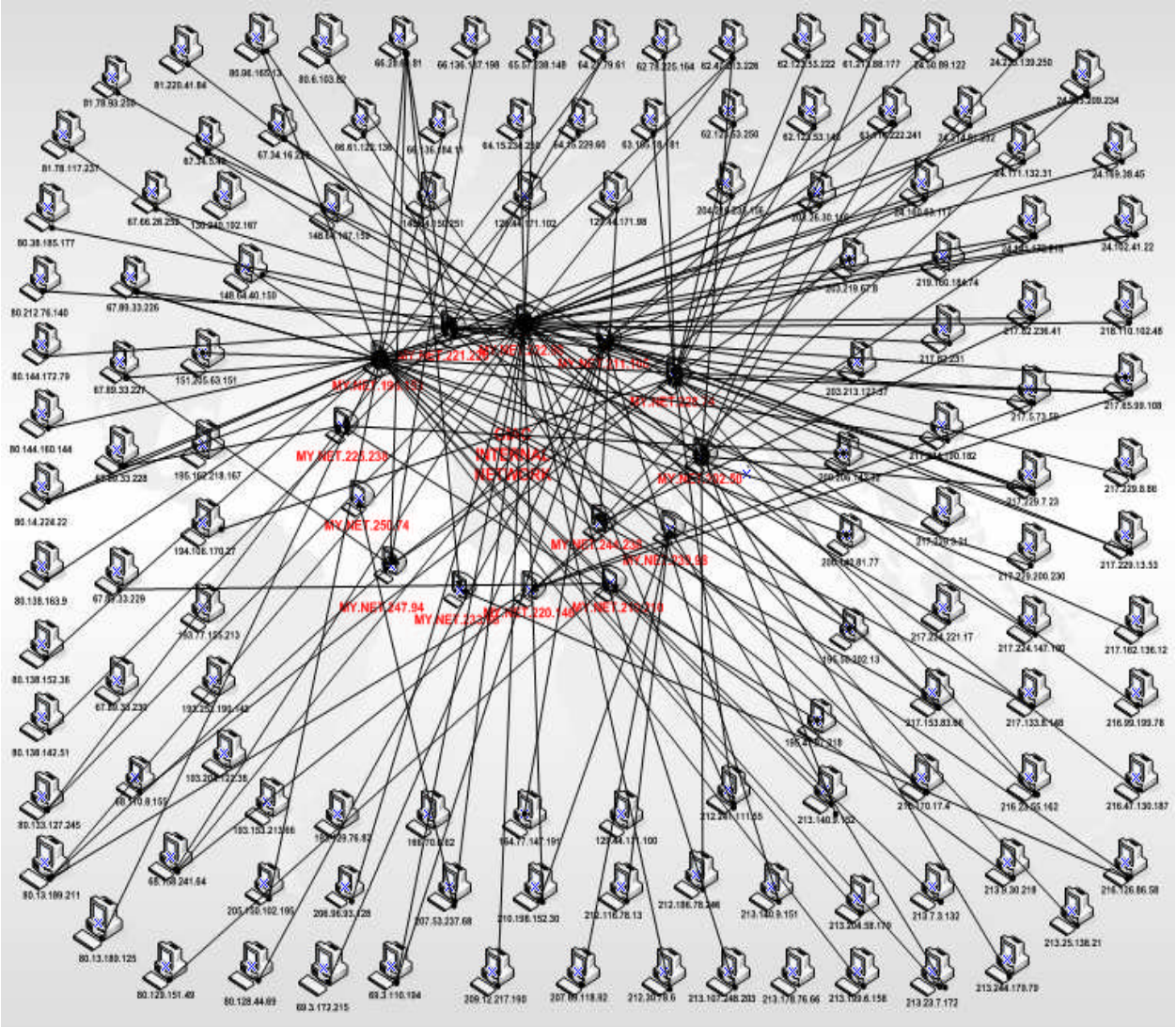
```
Alabama Supercomputer Network ASC-NET4 (NET-207-157-0-0-1)
                                  207.157.0.0 - 207.157.127.255
Tuskegee University TUSKEGEE-157-100 (NET-207-157-100-0-1)
                                  207.157.100.0 - 207.157.106.255

netnum:       80.60.0.0 - 80.60.255.255
netname:       NL-PMG-ADSL
descr:         ADSL5
country:       NL
```

4.1 OOS Files

The OOS files or out of spec files are packets that do not conform to TCP standards such as bogus flag combinations along with TCP sequence anomalies. There is high probability these packets may have been crafted. The majority of these packets were being used with some form of P2P file sharing. The following table shows the top offenders over the five day audit period which was using port 6346: this port is used with the Gnuetella[37] P2P program.

Count	Offender
1067	MY.NET.228.74:6346
243	MY.NET.222.98:6346
185	MY.NET.211.106:6346
159	MY.NET.196.153:6346
51	MY.NET.202.50:6346
41	MY.NET.247.94:6346
19	MY.NET.221.226:6346

Link Graph will Give an Overview of the top internal hosts connecting with PC's out on the internet with this P2P application. Internal hosts from my network are listed in red.



Gnuetella file sharing

The next table shows us the top offenders using the edonkey[38] P2P software: This application will use port 4662 to communicate.

Count	Offender
662	MY.NET.226.98:4662
162	MY.NET.222.174:4662
129	MY.NET.219.14:4662
84	MY.NET.203.2:4662
63	MY.NET.189.62:4662
47	MY.NET.233.150:4662
45	MY.NET.71.164:4662
26	MY.NET.168.130:4662

This traffic from a packet standpoint appears to be normal except for one anomaly. The syn flag is set, which is normal, however the higher order bits in the flag field are also set, and this is abnormal. The following OOS files show us just that.

=+

02/28-00:12:42.422058 24.102.41.22:64001 -> MY.NET.222.98:6346
TCP TTL:48 TOS:0x0 ID:263 IpLen:20 DgmLen:60 DF
12**S*** Seq: 0x3BE5B753 Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 206288769 0 NOP WS: 0

=+

02/28-00:14:47.799268 212.241.111.55:41832 -> MY.NET.228.74:6346
TCP TTL:50 TOS:0x0 ID:11048 IpLen:20 DgmLen:44 DF
12**S*** Seq: 0x483FE7A7 Ack: 0x0 Win: 0x16D0 TcpLen: 24
TCP Options (1) => MSS: 1460

=+

02/28-00:22:17.841987 212.241.111.55:41927 -> MY.NET.228.74:6346
TCP TTL:50 TOS:0x0 ID:22991 IpLen:20 DgmLen:44 DF
12**S*** Seq: 0x65822592 Ack: 0x0 Win: 0x16D0 TcpLen: 24
TCP Options (1) => MSS: 1460

There are many more examples of possible packet crafting. Observe the following packets and notice the illegal combination of the TCP flags.

02/28-00:17:21.518709 216.26.216.17:18245 -> MY.NET.24.34:21536
TCP TTL:114 TOS:0x0 ID:64816 IpLen:20 DgmLen:417 DF
*2U**RSF Seq: 0x2F746D70 Ack: 0x2F696E73

This packet is URGENT, it would like to RESET the connection, its SYN bit is set to start a new connection and lastly it'sFIN bit is set to close a connection. Here is another interesting one.

02/28-16:00:38.647717 68.32.118.5:1325 -> MY.NET.29.11:443
TCP TTL:112 TOS:0x0 ID:62726 IpLen:20 DgmLen:40 DF
****P*SF

This packet is saying PUSH the data with the PUSH bit set. While also wanting to SYN or set up a new communication connection with the SYN bit set and lastly finish the session with the FIN bit set.

5.1 Top External Talkers

12.222.59.215

```
CustName:   Insight Communications Company
Address:    810 7th Avenue
City:       New York
StateProv:  NY
PostalCode: 10019
Country:    US
RegDate:    2003-10-10
Updated:    2003-10-10

NetRange:   12.222.56.0 - 12.222.63.255
CIDR:       12.222.56.0/21
NetName:    INSIGHT-12-222-56-0-NOBLESVILLE
NetHandle:  NET-12-222-56-0-1
Parent:     NET-12-0-0-0-1
NetType:    Reassigned
Comment:
RegDate:    2003-10-10
Updated:    2003-10-10
```

66.28.249.232

```
OrgName:    Cogent Communications
OrgID:      COGC
Address:    1015 31st Street, NW
City:       Washington
StateProv:  DC
PostalCode: 20007
Country:    US

ReferralServer: rwhois://rwhois.cogentco.com:4321/

NetRange:   66.28.0.0 - 66.28.255.255
CIDR:       66.28.0.0/16
NetName:    COGENT-NB-0000
NetHandle:  NET-66-28-0-0-1
Parent:     NET-66-0-0-0-0
NetType:    Direct Allocation
NameServer: AUTH1.DNS.COGENTCO.COM
NameServer: AUTH2.DNS.COGENTCO.COM
Comment:    ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
Comment:    Reassignment information for this block can be found at
Comment:    rwhois.cogentco.com 4321
RegDate:    2000-10-12
Updated:    2001-12-05
```

68.162.130.22

```
OrgName:    Verizon Internet Services
OrgID:      VRIS
Address:    1880 Campus Commons Dr
City:       Reston
StateProv:  VA
PostalCode: 20191
Country:    US

NetRange:   68.160.0.0 - 68.163.255.255
CIDR:       68.160.0.0/14
NetName:    VIS-68-160
NetHandle:  NET-68-160-0-0-1
Parent:     NET-68-0-0-0-0
NetType:    Direct Allocation
NameServer: NSDC.BA-DSG.NET
NameServer: GTEPH.BA-DSG.NET
Comment:
RegDate:    2002-08-30
Updated:    2003-07-18
```

219.61.198.67

```
inetnum:      219.0.0.0 - 219.63.255.255
netname:      BBTECH
descr:        SOFTBANK BB CORP
descr:        Nation wide network in Japan
country:      JP
admin-c:      SA127-AP
tech-c:       SA127-AP
mnt-by:       APNIC-HM
mnt-lower:    MAINT-JP-BBTECH
changed:      hostmaster@apnic.net 20011031
changed:      hm-changed@apnic.net 20030616
status:       ALLOCATED PORTABLE
source:       APNIC
```

219.111.13.142

```
inetnum:      219.96.0.0 - 219.127.255.255
netname:      JPNIC-NET-JP
descr:        Japan Network Information Center
country:      JP
admin-c:      JNIC1-AP
tech-c:       JNIC1-AP
mnt-by:       APNIC-HM
mnt-lower:    MAINT-JPNIC
changed:      hostmaster@apnic.net 20020307
status:       ALLOCATED PORTABLE
source:       APNIC
```


© SANS Institute 2004, As part of GIAC practical repository. Author retains full rights.

51.a Top Internal talkers

MY.NET.83.205
MY.NET.246.54
MY.NET.105.204
MY.NET.236.246
MY.NET.88.193

6.1 Analasys and Tools

The tools and techniques used to evaluate the data were from both UNIX based and Win32 based operating systems. First files analyzed were the Alert files. The required five days of Alert files were downloaded; these files originally were compressed and zipped. The Unix "gzip" command with the –d (decompress) option was used on them. This in turn unzipped and decompressed them in to five individual files. The "cat" command was then used to combine all the five days into one file. The Unix command 'sed' was then used to remove some of the data fields generated by the alerts. The remaining data was then separated to use a common deliminater to allow for an easier import into a database application. The data was then imported into a Microsoft Access database. Custom query's where then used to produce the top ten alerts that were tripped during these past five days. The proceeding process was also used for the scan and oos files. With some difference in the data which was removed along with data which was kept. Other commands used to manipulate the data were the UNIX based, "grep", "sort" and "awk" and "uniq". TCPDUMP was also used to evaluate some of the data which was captured over the five day period.

Websites such as google.com and yahoo.com were used to querrie for port relevant information. Sites such as DSHIELD.org, mynetwatchman.com were used in the correlation of the external networks found to be generating some of the rules being tripped. ARIN.NET was used to provide ownership of the fore mentioned network addresses. Some previous student practical were also used to provide correlation.

6.1 Foot notes and references

1	Microsoft Corporation	http://www.microsoft.com/
2	Sun Micro Systems	http://www.sun.com
3	HP-UX	http://hpux.cs.utah.edu/
4	BSD	http://www.bsd.org/
5	Linux	http://www.linux.org/
6	F5 Networks	http://www.f5.com/
7	Cisco Systems	http://www.cisco.com/
8	Nortel Networks	http://www.nortelnetworks.com/index.html
9	IPSEC	http://www.ietf.org/html.charters/ipsec-charter.html
10	Check Point Systems	http://www.checkpointsystems.com/content/home/default.aspx
11	SourcFire Network Security	http://www.sourcefire.com/
12	RedHat	http://www.redhat.com/
13	Intel Corporation	http://www.intel.com/
14	Snort	http://www.snort.org/
15	Oinkmaster	http://www.oinkmaster.com/
16	Idlescan	http://www.insecure.org/nmap/idlescan.html
17	P0f	http://lcamtuf.coredump.cx/p0f.shtml
18	Mynetwatchman.com	http://www.mynetwatchman.com/
19	Arin	http://www.arin.net
20	NBTscan	http://www.inetcat.org/software/nbtscan.html
21	Adore worm	http://www.sans.org/y2k/adore.html
22	RemotelyAnywhere	http://www.majorgeeks.com/download1019.html
23	Myserver	http://www.securityfocus.com/archive/75/140891
24	Kazaa	http://www.kazaa.com/us/index.htm
25	Port 1214	http://www.cites.uiuc.edu/newsletter/spring02/1214.html
26	Port 32771	http://www.whitehats.com/info/IDS429
27	SubSeven	http://www.f-secure.com/v-descs/subseven.shtml
28	NetBIOS worms	http://www.robertgraham.com/pubs/firewall-seen.html#10.7
29	Half – Life	http://games.sierra.com/games/half-life/
30	Soribada	http://www.soribada.com
31	SQL Buffer Overflow	http:// www.xfocus.org/documents/200308/3.html
32	My Doom	http://techlibrary.networkcomputing.com/detail/RES/1086978395_946.html
33	FTP Exposures	http://www.iss.net/security_center/advice/Exploits/Services/FTP/default.htm
34	Nachi	http://www.microsoft.com/security/incident/nachi.mspx
35	445 Exploits	http://www.esecurityplanet.com/alerts/article.php/3350211
36	W32.Korgo	http://vil.nai.com/vil/content/v_126344.htm
37	Gnutella	http://www.gnutella.com/
38	edonkey	http://www.edonkey2000.com/

Koo, Alfred, Intrusion Detection in Depth CGIA Practical Assignment, September 7,2003
url:http://www.giac.org/practical/GCIA/Alfred_Koo_GCIA.pdf

King, Tom, GIAC Intrusion Detection in Depth Practical Assighnment, November 19, 2003
url:http://www.giac.org/practical/GCIA/Tom_King_GCIA.pdf

Hoover, James, CGIA Practical Intrusion Detection in Depth, December 23, 2001
url:http://www.giac.org/practical/James_Hoover_GCIA.doc

Moncalm, Eric, Pratical Assignment for CGIA Intrusion Detection, November 12, 2003
url: http://www.giac.org/practical/GCIA/Erik_Montcalm_GCIA.pdf

Fiddler, Matthew, Mathew Fiddler
url:http://www.giac.org/practical/Matthew_Fiddler_GCIA.doc

Binde, Beth, GCIA Practical Intrusion Detection in Depth, May 12 2003
url:http://www.giac.org/practical/GCIA/Beth_Binde_GCIA.pdf

Thompson, Jason, CGIA Practical Intrusion Detection in Depth, July 21, 2003
url:http:// www.giac.org/practical/GCIA/Jason_Thompson_GCIA.pdf

Parker, Donald , GCIA Certified Intrusion Analyst Practical, 2003
url: http//http://www.giac.org/practical/GCIA/Donald_Parker_GCIA.pdf

Faske, Robert, GCIA Certified Intrusion Analyst Practical , December 7, 2003
url: http://www.giac.org/practical/GCIA/Mark_Faske_GCIA.pdf