Global Information Assurance Certification Paper

SANS Intrusion Detection in Depth
GCIA Practical Assignment
Version 3.1

Suchun Wu

Toronto, September, 2002

1. Assignment 1 - A Comparison of Intrusion Detection Systems

Table of Contents

1.1. Introduction and Objective

In the protection of the organization from hazards, it is generally considered that firewall provides the first line of defense. However, for a completed protection, using only firewall is not sufficient [5]. One needs to build the defense in depth and in hierarch. This means that one needs to build the second and even third line of defenses. In order to achieve this, people are increasingly using IDS (Intrusion Detection Systems) to make the defense more solid and complete than that with only firewall in place.

In order to gain a good understanding of different ID systems and make a faire recommendation of IDS deployment to the enterprise, we attempt, in this paper, to make a comparison with different network IDSs, namely RealSecure [13], NFR [3], Shadow [12], and Snort [2]. The former two are commercialized products and the latter are in the public domain. These four IDSs are actually most deployed by information security professionals.

Generally speaking, each of IDSs has its own strength and limitations. For example, some IDSs are blind to some types of attacks, and some are not. Some may suffer occasional failure to synchronize data between sensors and console. As a result, the loss of critical log information could be a serious problem to make an intrusion detection system more useful. So it is the purpose of this paper to find out the strength and limitation for the IDSs in questions.

What is an IDS?

Without giving out their definition (please refer to [1]), ID systems can be classified into the following four categories based on the types of data they examine:

- Application-based IDSs;
- Host-based IDSs
- Network-based IDSs
- Multi-network/infrastructure IDSs.

In this paper, we concentrate on network IDSs. A network IDS can monitor packets on the network wire and attempt to discover if a hacker/cracker is attempting to break into a system (or cause a denial of service attack).

Based on the intrusion detection method or approach an IDS deploys, it can be either misuse-based or anomaly-based. The former relies on a predefined set of attack signatures created by the vendors, and/or IDS' authors, or by network administrators. By looking for specific patterns, the IDS attempts to match every incoming package on the network segment to the signature of a known attack. The latter focuses on building a statistical model of network's normal behavior and generating an alarm when the activity falls outside the modeled norm. In terms of their implementation, misuse-based IDS is much easier than anomaly-based ones. That is probably the

reason why the network IDS that we are actually using almost exclusively belongs to misuse-based IDSs. In this paper, we will not compare with these two approaches. We will concentrate on the functionalities, usefulness, and scalability of the IDSs (i.e. RealSecure [13], NFR [3], Shadow [12], and Snort [2]) we are going to evaluate. All these IDSs belong to miuse-based category.

The logical architecture of an IDS normally consists of three components: Sensors, analyzers, and user interface (often called console). Sensors are responsible for collecting data over the network; analyzers receive the data from the sensors and determine if an intrusion has occurred; and the user interface enables the users to view the output from the system or control the behavior of the system.

Why do we need using different IDSs?

Although the general intrusion detection approach could be the same to the IDSs in question, there should be many ways to realize and implement by different vendors or IDS authors. For example, one IDS can support real-time detection by matching known attack signatures and the other can support post-analysis by comparing actual system status with known 'perfect and intact' system status. Furthermore, each vendor or author of IDS has his/her own understanding and exposure to different and ever changing attacks. This fact could heavily influence the design of attack signatures database, which is a very important feature for evaluating an IDS.

Again, it is the purpose of this paper to make a comparison with different ID products along with their ability of intrusion detection and their distinct strength and weaknesses. Based on this comparison, a recommendation on wise deployment of different IDSs will be made.

Which IDS do we choose for testing?

We have chosen the following four IDSs for our study and test:

- ISS' RealSecure 6.5 (Commercialized at http://www.iss.net)
- NFR's Network Flight Recorder 5.0 (Commercialized at http://www.nfr.com)
- Shadow by Bill Palph (Freeware at http://www.nswc.navy.mil/ISSEC/CID)
- Snort by Martin Roesch (Freeware at www.snort.org)

What are the features of an IDS to be considered?

When doing a comparison, we consider that the following aspects serving as criteria need to be taken into account:

1. The capacity of intrusion detection against different attacks
2. The types of data they examine: host-based, application-based, network-based, and multiple-networks/infrastructure-based
3. The ways of intrusion detection: real-time, and post-analysis

4. Analysis methods: attack signature detection, and anomaly detection;
5. Alert/notification mechanisms (Email, Console, Paging, or Sound)
6. Data and storage management (signatures, policy/rules customization, resource overload)
7. Report features (powerfulness and customization)
8. User interface (user-friendliness)
9. Technical support for commercialized IDSs.

Attack Tools

In order to make the test more realistic, we need to deploy different attack tools currently used by "hackers". According to our research and the current literature, we have identified the following tools which will be used as our testing tools:

- nmap (for Unix) nmapfe (graphical version of nmap) or nmapnt (for NT)
- targa2
- nessus
- Cybercop (commercial from NAI at the time of testing.)

With these tools, we can generate many types of misuses attacks to the IDSs under test:

- Denial of service
- Unauthorized access to the testing environment
- Surveillance and probing
- Anomalous network usage

1.2. The Evaluation

The IDS evaluation will be processed in the way described in section Testing Steps.

Testing Steps

1) Creating the same environment for different kinds of sensors. This is to create a totally isolated network within the lab.
2) Identifying different and current attack tools (Nmap, Cybercop, Nessus, …)
3) Using the identified tools to attack the isolated network.
4) Monitoring closely the reactions from the different kinds of sensors.
5) Documenting the experimental data.
6) Making a proposal for the deployment of different IDS for the IPC.

Testing Environment Setup

Figure-1 shows our IDS testing environment setup. We have used five different computers to construct it:

1) IBM ThinkPad E600 laptop on which Window 2000 Pro. Is the OS. Both RealSecure sensor and console are installed on it.
2) Dell GX150 desktop on which Linux mandrake 8.2 is installed. It is mainly used as an analysis station for both Shadow and Snort. In addition, a number of attack tools, such as, nmap, nessus, targa2 …are installed.
3) Sparc/Ultra5 on which Solaris8 is the OS. It's used as NFR's central station.
4) Dell GX150 desktop which is a dedicated NFR sensor with NFR's proprietary OS.
5) Toshiba Tecra laptop on which Linux mandrake 8.2 is the OS. It is dedicated as a Shadow's sensor.

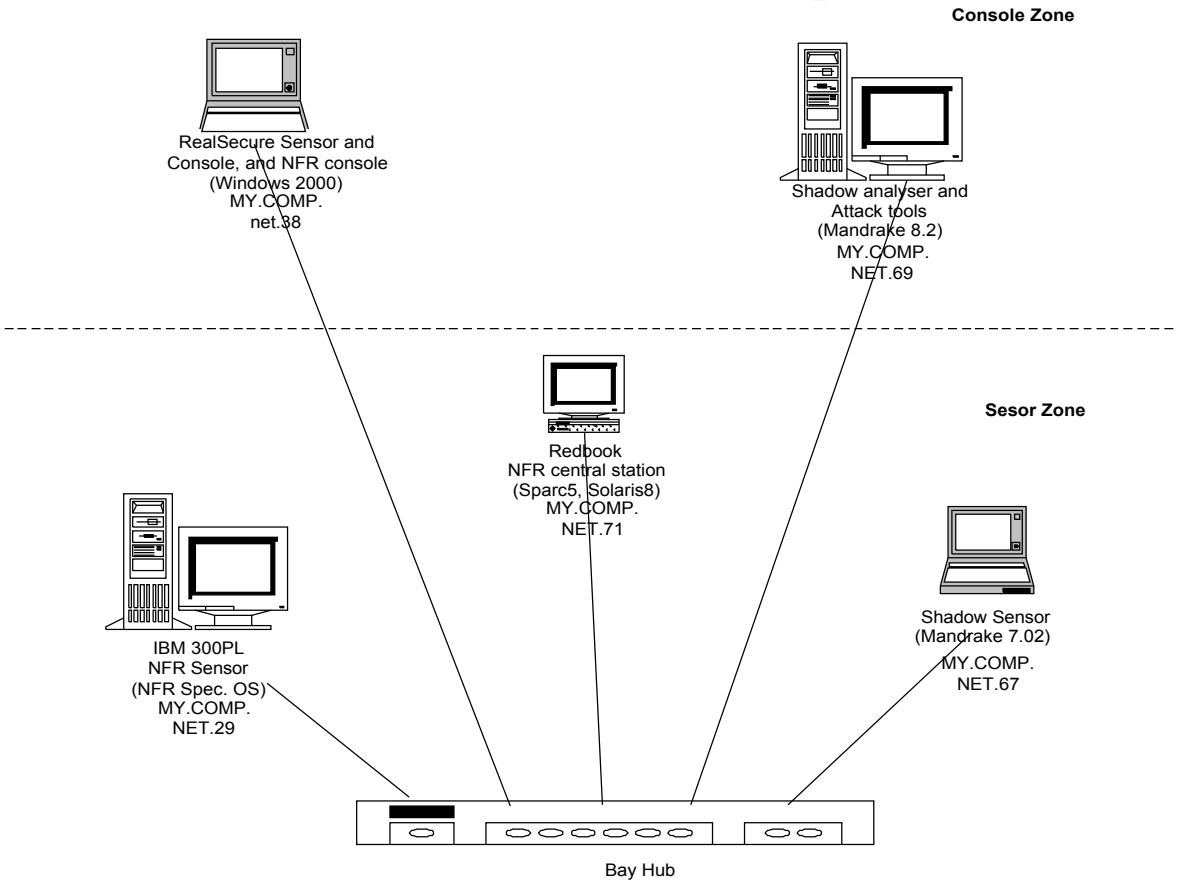


Figure-1.1 Testing Environment Diagram

1.3. IDS Comparison

Brief Description of IDS in question

RealSecure

RealSecure from Internet Security Systems uses a three level architecture consisting of a network-based recognition engine, a host-based recognition engine, and an administrator's module. In this paper, we only concentrate on the network recognition engine. Each network recognition engine watches the packet traffic traveling over a specific network segment for attack signatures evidence that an attempted intrusion is taking place. When a network recognition engine detects unauthorized activity, it can respond by terminating the connection, sending email or pager alerts, recording the session, reconfiguring selected firewalls, or taking other user-definable actions. In addition, a network recognition engine passes an alarm to the administrator's module or a third-party management console for administrative follow-up and review.

All recognition engines report to and are configured by the administrative module, a management console that monitors the status of any combination of UNIX and Windows NT recognition engines. The result is comprehensive protection, easily configured and administered from a single location. The administrative module ships with either recognition engine and is also available as a plug-in module for a variety of network and systems management environments.

Network Flight Recoder

Network Flight Recorder™ (NFR) Network Intrusion Detection (NID) is an ID system that was previously available in both a commercial version and a public domain version. In the following, we use the term NFR for NFR NID (see http://www.nfr.com/products/NID/docs/NFR_NID_Product_Overview.pdf for more detail).

NFR uses a modified version of libpcap to promiscuously and passively extract packets off the network. A NFR sensor should be installed on a proprietary OS with PentiumIII PC.

NFR goes beyond simply analyzing headers as it reassembles TCP streams. Data collection is performed by a sensor. In a distributed environment, the raw data are stored onto a central station (usually a SUN box). NFR sensor and console or analyzer can be placed at strategic internal points to detect potential insider threats. Above the packet extraction level, a decision engine filters and reassembles packets.

NFR includes a complete programming language, called N, designed for packet analysis. Filters are written in the language, which is compiled into byte-code and interpreted by the execution engine. Through programs written in N, pattern matching is performed to allow packets to be reassembled.

The functions *alert* and *record* are used to extract data after the filtering operation and to support back ends. The *alert* function can send events via email or fax. The *record* function tailors the data into formats required by the various backend analysis modules.

Histogram and *list* are two primary examples of multi-purpose backends. *Histogram* provides a facility for capturing data in a multi-dimensional matrix. Totals of relevant categories are accumulated in the cells of the matrix. Alerts can be generated based on the absolute numbers or relative frequencies within the cells. The *list* backend records chronological records, thus providing a level of detail (at the expense of storage) not provided by the histogram function.

NFR also provides *query* backends that allow you to analyze the data. *Query* backends were designed so as not to degrade the performance of the execution engine since this could lead to dropped packets. *Query* backends have their own CGI interface. Also, *query* backends provide graphical capability to allow data to be viewed more easily.

Shadow

Shadow uses what it calls sensor and analysis stations. Sensors extract the packet headers and save them to a file. This file is, by default, read hourly by the analyst station, which then performs the filtering operation and generates a second log file. The Shadow philosophy is not to provide alerts when events are identified. This approach was motivated through experience with other ID systems, where many alerts turned out to be false alarms and were distracting and annoying. This needs user to customize the traffic filters in a more specific way.

The sensor station uses the libpcap utility developed by the Lawrence Berkeley Laboratories Network Research Group to provide a basic sniffer capability. The sensor station does not preprocess the data, thus preventing an intruder from checking what is done with the packets.

Major support for analysis is provided by tcpdump through which packet filters are defined and executed. However, some intrusions were difficult to detect with tcpdump filters, particularly those involving infrequent probes. For these types of events, Shadow provides a Perl-based tool, *one_day_pat.pl,* as part of its kit. This allows one to scan for low-frequency patterns that may occur in more than one log file. Filters can be simple or compound (Boolean) collections of simpler filters. An example of a simple filter is *tcp and dest port 23*. This simple filter selects packets with the TCP protocol and destination port 23 (i.e., telnet).

The analysis station uses a Web-based interface to display information from the sensors, or to display the results of filtering operations on the raw data. Shadow runs on many UNIX systems and on open source systems like FreeBSD or Linux.

Snort

Snort has recently become very popular. This so-called lightweight network intrusion detection tool can be deployed to monitor small TCP/IP networks and detect a wide variety of suspicious network traffic as well as outright attacks. It can provide administrators with enough data to make informed decisions on the proper course of action in the face of suspicious activity. Snort can also be deployed rapidly to fill potential holes in a network's security coverage, such as when a new attack emerges and commercial security vendors are slow to release new attack recognition signatures.

Snort is also a libpcap-based packet sniffer and logger. It features rules based logging to perform content pattern matching and detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, and much more. Snort has real-time alerting capability, with alerts being sent to syslog or a separate "alert" file. Snort is configured using command line switches and optional Berkeley Packet Filter commands. The detection engine is

programmed using a simple language that describes per packet tests and actions.

Since snort is in the public domain and popular, there are a lot of contributions that can be added on top of the original packages. One of the most important contributions is ACID by Roman Danyliw and Jed Pickel. ACID is a PHP-Apache based graphical analysis engine to search and process a database of security incidents log generated by the Snort. In order to make it work, Mysql or Postgre should be used as database for Snort to store the alert information. PHP 4 enabled apache server needs to be installed. Another popular tool is Snort log analysis tool called Snortsnarf. For more information, go to the web page at http://www.snort.org.

System Requirements

By it's nature, an ID system is also a computer resources consumer. In particular, there is no maximum requirement for both data storage and CPU speed. It is all depending upon the traffic of the network to which the IDS is monitoring. So, most IDSs require using a dedicated high-capacity computer for both the sensor and console. Table-1.1 shows the minimum system requirements for the IDSs in question.

It should be noted that NFR introduces a central station to collect the data from sensors in a distributed environment. This can mitigate the disk space burden on each sensor. For this reason, the console needs not to be a dedicated computer which can be your normal working desktop or laptop.

	RealSecure 6.5	Network Flight Recorder 5.0	Shadow by Bill Palph	Snort by Martin Roesch
Sensor with Windows/NT	PIII 400 MHz 128 MB RAM 20 MB install + 100MB for each sensor dedicated computer recommended	PIII 800 MHz or AMD 1000 MHz 512 MB RAM 20 MB Disk Must be a dedicated computer With NFR OS		PIII 400 MHz 128 MB RAM 20 MB install + 100MB for each sensor dedicated computer recommended
Sensor with Unix/Linux	Sparc Processor 128 MB RAM 25MB install +150MB for log files and database Solaris 2.6,2.7,2.8 Dedicated computer recommended	Sparc Processor 128MB RAM > 6GB Disk Solaris 2.6, 7, or 8	PIII 400 MHz 256MB RAM > 6GB Disk dedicated computer recommended	PIII 400MHZ 256MB RAM 6GB Disk or Sparc Processor 128MB RAM > 6GB Disk dedicated computer recommended
Central Station		Sparc Processor 128 MB RAM Solaris 2.6, 7, or 8		

Console with Windows/NT	PIII 400 MHz 128 MB RAM 20 MB install + 100MB for each sensor managed dedicated computer recommended	PIII 800 MHz or AMD 1000 MHz 512 MB RAM 20 DB Disk dedicated computer recommended		
Console with Unix/Linux			PIII 400 MHz 256MB RAM > 6GB Disk	PIII 400 MHz 256MB RAM > 6GB Disk

Table-1.1 Minimum IDS System requirements

IDS Characteristics

We consider that the items listed in Table-1.2 are of interest for us to compare the IDS in question.

It is true that in terms of documentation and ease of implementation of the IDSs, the both commercial ones are very good.

As far as alert mechanisms are concerned, both commercialized products have integrated alert system, such as email, console screen pop up, and paging. However, NFR is able to pop up the alerts with a sound onto the console, while RealSecure does not. This feature is interesting for an IDS operator to take a look when an audio alert comes out, rather than looking at the console screen every time. It should be noted, that Snort is also able to make alerts via email, log-file, web-browser. One thing we found is that there is no IDS offers an 'intelligent' alerting system. Here, 'intelligent' means that the console can correlate the alerts received from the sensors and inform the IDS analysts only those alerts meeting certain criteria. This feature is very important and challenging in the view of the fact that one difficulty in managing IDS is how to deal with its false positives.

Reporting systems in both RealSecure and NFR are quite good. Especially, RealSecure 6.0 or up can send events directly into MSSQL server. This gives a easier and scalable way for IDS analysts to do their reporting and data mining. For Snort alerts reporting, one has to use a third part software like ACID and SnortSnaf to make some snap shots, but not comprehensive reports and alerting system. It is of user's charge to make make it work. We also experience some difficulty to use SnortSnarf to analysis Snort alert file with a big size (e.g. more than 15GB).

One important shortcoming of RealSecure is that there is no way for its users to customize the filtering rules or signature database in order to cope with some newly occurred attacks. Although the company will create new signatures, these new signatures cannot directly put into users' IDS of the existing version until they get a new version. By contrary, the users can do so with other

IDS we evaluated. For example, when a new attack occurs, when the user knows the attack's characteristics, he/she can make new filter by using so-called N-Code language with NFR. Of course, it needs some effort to learn the language. But, when he/she gets familiar with it, he/she can really profit from it by making an IDS more useful and meaningful to his/her environment.

It should be noted that since its popularity, Snort's signature coverage is certainly more complete than its commercial rivals' (see Table-1.2). Although we cannot obtain an exact signature number for each IDSs based on the same criteria, the numbers we collected in Table-1.2 supports this point. These numbers can be obtained from [2], [3], [11], and [12].

	RealSecure 6.5	Network Flight Recorder 5.0	Shadow by Bill Palph	Snort by Martin Roesch
Type of IDS	Network-based	Network-based	Network-based	Network-based
Way of Detecting	Real-time	Real-time	Post Analysis	Real-time
Analysis Method	Attack signature-based	Attack signatures-based	TCP-Dump Filtering rules	TCP-Dump Filtering rules
Number of Attack Signatures	1200 including BlackIce's	1300	User defined tcpdump filter rules	1643 alerting rules + filters
Alerts	E-mail or pager alerts Alert sent to Console	Popup alerts to console with sound, email,	Dumped raw data	Log-message, paging
User Interface	Graphical and easy to use	Graphical and easy to use	View results via a browser	View results via a browser
Signature Customizable	No	Yes	Customizable filtering rules	Yes
Report	Powerful & customizable	Graphical and customizable	Searchable & customizable	Searchable by ACID package
Ease of Implement. & config.	Very easy	Very easy	Need some knowledge	Easy need some knowledge for setting up a graphical console

Table-1.2 IDS features comparison

Experimental Attack Testing Data

This section is to test the strength of intrusion detection for each IDS in question. The attack tools used are Targa2, nmap, nessus, and cybercop. These scanner or attack tools include many well-known attacks. For example, targa2 contains mainly DOS oriented attacks like Land, teadrop, etc. Nmap contains a wide range of port probing tools. Both Nessus and Cybercop contain many denial of service attacks. For each of these tools, we test the IDSs by choosing the individual attacks, then the full attacks of the tools.

The test data are recorded in the following four separated tables.

As shown in Table-1.3 – Table-1.6, RealSecure detects most attacks we launched. NFR can also

generally detect them, but less precise then RealSecure does. For example, RealSecure can detect teadrop and mstream, while NFR tells you only an "invalid net attack" on the network. Interesting enough, snort can detect most of them by giving more detailed and lower level information. As far as Shadow is concerned, it is able to record the raw data in TCPDUMP format about these attacks. It is at the user's control that he/she can make more specific tcpdump filters to avoid the messy of the logs and false alarms produced by the IDSs.

IDS Testing for RealSecure

S T	E T	Source IP	Dest IP	Attack Tool/Name	Rpt	Rct	Alert Message	Additional Information
2:04	2:07	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Land	100	Yes	Land	
2:13	2:15	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Nestea	100	Yes	Tear Drop	
2:54	2:54	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Newtear	100	Yes	Tear Drop	
2:59	3:00	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Syndrop	100	Yes	Tear Drop	
3:02	3:03	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Teardrop	100	Yes	Tear Drop	
3:03	3:04	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Winnuke	100	Yes	Windows_OOB	Also showed Netbios session
3:05	3:05	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/1234	100	Yes	IPFragment	
3:06	3:07	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Sailhyousen	100	No	None	
3:09	3:11	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Oshare	100	No	None	
3:11	3:23	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/All DoS	100	Yes	IPFragment	
3:39	3:40	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Bonk	100	Yes	Tear Drop	
3:40	3:41	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Jolt	100	No	None	
10:38	10:39	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/UDP portscan	2	Yes	UDP Port Scan	Nmap -SU -F -PT - O MY.COMP.NET.38
10:46	10:46	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/connect	1	Yes	Port Scan, IPHalfScan, Nmap	Nmap -ST -F -O -P0 -I MY.COMP.NET.38
10:49	10:49	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/SYN Stealth	2	Yes	IPHalfScan, Nmap, Port scan	Nmap -sS -O MY.COMP.NET.38

11:03	11:04	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/FIN Stealth	2	No	None	Nmap -sF -F -O MY.COMP.NET.38
11:32	11:38	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/5 attacks	1	Yes	Port Scan, IPHalfScan, BackOrfice,	
							TrinOO, Nmap, Mstream Zombie,	
							Windows Access Errors	
2:15	2:20	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/5 attacks	1	Yes	IPHalfScan, Gaining Shell Remotely	
							Nmap	
2:25	2:31	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/6 attacks	1	Yes	HP OpenView SNMP Backdoor, Land	
							Sun SNMP Backdoor, Port scan,	
							IPHalfScan, Nmap, Windows OOB	
2:40	3:00	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/All attacks	1	Yes	Port scan, IPHalfScan, Nmap, TrinOO,	
							Sun SNMP Backdoor, BackOrfice,	
							Mstream Zombie, Land, Windows OOB	
							HP OpenView SNMP Backdoor	
							Windows Access Errors	
3:44	3:46	MY.COMP.NET.69	MY.COMP.NET.38	CyberCop/Default	1	Yes	IPHalfScan, Chargen DoS, Portscan,	
							SYNFlood, Land, TearDrop	

Table-1.3 Experimental Data for RealSecure

Where ST: Start time; ET: End time; Rpt: Repeated attack times; Rct: Console React.

IDS Testing for NFR								
S T	**E T**	**Source IP**	**Dest IP**	**Attack Tool/Name**	**Rpt**	**Rct**	**Alert Message**	**Additional Information**
2:04	2:07	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Land	100	Yes	Land	
2:13	2:15	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Nestea	100	No	None	

2:54	2:54	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Newtear	100	Yes	Invalid Net Attack	
2:59	3:00	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Syndrop	100	Yes	Invalid Net Attack	
3:02	3:03	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Teardro p	100	No	None	
3:03	3:04	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Winnuk e	100	Yes	WinNuke DoS	
3:05	3:05	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/1234	100	Yes	Invalid Net Attack	
3:06	3:07	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Sailhyo usen	100	Yes	Invalid Net Attack	
3:09	3:11	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Oshare	100	Yes	Invalid Net Attack	
3:11	3:23	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/All DoS	100	Yes	Invalid Net Attack	
3:39	3:40	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Bonk	100	Yes	Invalid Net Attack	
3:40	3:41	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Jolt	100	Yes	Invalid Net Attack	
10:38	10:39	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/UDP portscan	2	Yes	Portscan	Nmap -SU -F -PT - O MY.COMP.NET.38
10:46	10:46	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/connect	1	Yes	Portscan	Nmap -ST -F -O -P0 -I MY.COMP.NET.38
10:49	10:49	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/SYN Stealth	2	No	None	Nmap -sS -O MY.COMP.NET.38
11:03	11:04	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/FIN Stealth	2	No	None	Nmap -sF -F -O MY.COMP.NET.38
11:32	11:38	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/5 attacks	1	Yes	Portscan	
2:15	2:20	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/5 attacks	1	Yes	Portscan, SNMP, DNS, TFTP	
2:25	2:31	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/6 attacks	1	Yes	Portscan, Land, WinNuke,	
							Suspicious activity	
2:40	3:00	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/All attacks	1	Yes	Suspicous activity, Land, TFTP,	
							WinNuke, SNMP, Portscan	
3:44	3:46	MY.COMP.NET.69	MY.COMP.NET.38	CyberCop/Defa ult	1	Yes	Echo-CharGen, Land,	

							Annex Terminal Server DoS	
							Portscan, Invalid Net Attack	

Table-1.4 Experimental Data for NFR

IDS Testing for Snort								
S T	E T	Source IP	Dest IP	Attack Tool/Name	Rpt	Rct	Alert Message	Additional Information
2:04	2:07	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Land	100	Raw Data	None	
2:13	2:15	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Nestea	100	Yes	Tiny Fragments	Possible hostile environment
2:54	2:54	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Newtear	100	Raw Data	None	
2:59	3:00	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Syndrop	100	Yes	Tiny Fragments	Possible hostile environment
3:02	3:03	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Teardrop	100	Yes	Tiny Fragments	Possible hostile environment
3:03	3:04	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Winnuke	100	Raw Data	None	
3:05	3:05	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/1234	100	Yes	Tiny Fragments	Possible hostile environment
3:06	3:07	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Sailhyousen	100	Raw Data	None	
3:09	3:11	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Oshare	100	Raw Data	None	
3:11	3:23	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/All DoS	100	Yes	Tiny Fragments	Possible hostile environment
3:39	3:40	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Bonk	100	Yes	Tiny Fragments	Possible hostile environment
3:40	3:41	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Jolt	100	Raw Data	None	
10:38	10:39	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/UDP portscan	2	Yes	Nmap	Nmap -SU -F -PT - O MY.COMP.NET.38
10:46	10:46	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/connect	1	Yes	Nmap	Nmap -ST -F -O -P0 -I MY.COMP.NET.38
10:49	10:49	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/SYN Stealth	2	Yes	Nmap	Nmap -sS -O MY.COMP.NET.38
11:03	11:04	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/FIN Stealth	2	Yes	Nmap	Nmap -sF -F -O MY.COMP.NET.38
11:32	11:38	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/5 attacks	1	Yes	DNS, Nmap, BackOrfice	
2:15	2:20	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/5 attacks	1	Yes	DNS, SNMP, Portmap, SMB	
2:25	2:31	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/6 attacks	1	Yes	Nmap, FTP-useless services	
2:40	3:00	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/All attacks	1	Yes	Nmap, RPC, DDoS-Stacheldraht,	
							DDoS-TrinOO, BackOrfice, TFTP	

							Tiny Fragments, IIS Attacks	
3:44	3:46	MY.COMP.NET.69	MY.COMP.NET.38	CyberCop/Default	1	Yes	Nmap, OS Probe, RFPanalyze	

Table-1.5 Experimental Data for Snort

IDS Testing for Shadow								
S T	**E T**	**Source IP**	**Dest IP**	**Attack Tool/Name**	**Rpt**	**Rct**	**Alert**	**Additional Information**
2:04	2:07	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Land	100	Raw Data	None	Data in tcpdump format
2:13	2:15	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Nestea	100	Raw Data	None	Data in tcpdump format
2:54	2:54	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Newtear	100	Raw Data	None	Data in tcpdump format
2:59	3:00	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Syndrop	100	Raw Data	None	Data in tcpdump format
3:02	3:03	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Teardrop	100	Raw Data	None	Data in tcpdump format
3:03	3:04	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Winnuke	100	Raw Data	None	Data in tcpdump format
3:05	3:05	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/1234	100	Raw Data	None	Data in tcpdump format
3:06	3:07	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Saihyousen	100	Raw Data	None	Data in tcpdump format
3:09	3:11	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Oshare	100	Raw Data	None	Data in tcpdump format
3:11	3:23	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/AllDoS	100	Raw Data	None	Data in tcpdump format
3:39	3:40	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Bonk	100	Raw Data	None	Data in tcpdump format
3:40	3:41	MY.COMP.NET.69	MY.COMP.NET.38	Targa2/Jolt	100	Raw Data	None	Data in tcpdump format

10:38	10:39	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/UDP portscan	2	Raw Data	None	Data in tcpdump format; Nmap -SU -F -PT - O MY.COMP.NET.38
10:46	10:46	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/connect	1	Raw Data	None	Data in tcpdump format; Nmap -ST -F -O -P0 -I MY.COMP.NET.38
10:49	10:49	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/SYN Stealth	2	Raw Data	None	Data in tcpdump format; Nmap -sS -O MY.COMP.NET.38
11:03	11:04	MY.COMP.NET.69	MY.COMP.NET.38	Nmap/FIN Stealth	2	Raw Data	None	Data in tcpdump format; Nmap -sF -F -O MY.COMP.NET.38
11:32	11:38	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/5 attacks	1	Raw Data	None	Data in tcpdump format
2:15	2:20	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/5 attacks	1	Raw Data	None	Data in tcpdump format
2:25	2:31	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/6 attacks	1	Raw Data	None	Data in tcpdump format
2:40	3:00	MY.COMP.NET.69	MY.COMP.NET.38	Nessus/All attacks	1	Raw Data	None	Data in tcpdump format
3:44	3:46	MY.COMP.NET.69	MY.COMP.NET.38	CyberCop/ Default	1	Raw Data	None	Data in tcpdump format

*Note: Filters should be created to maximize efficiency

Table-1.6 Experimental Data for Shadow

Technical Support Comparison for RealSecure and NFR

According to our experience, when signing a maintenance contract with the both ISS and NFR vendors, we can normally get reasonably good technical supports. Both companies support their products in a professional way. Note that one can now also get Snort technical supports at Silicon Defense [14].

Security Concern with IDS Computer

An IDS sensor is often put in the DeMilitarized zone (DMZ) of a protected network. The whole ID system relies on the data collected from its sensors. If the computer where the IDS sensor resides is widely open for attacks, the integrity of the data will be compromised. Some vendors provide users with some computer hardening scripts to help IDS administrator making the computer less vulnerable to attacks. For example, ISS RealSecure will ask you if you want to install the sensor in a secure environment.

As far as NFR is concerned, since NFR uses its own proprietary operating system for the sensor, some box security hardening measurements are taken into account in its original design.

For comparison purpose, we had run a nmap scan (nmap –sT –n –P0 sensor IP) on computers where both RealSecure and NFR sensors are resided. The result is shown in Table-1.8.

	Port	State	Service
RealSecure sensor (MY.COMP.NET.38)	135/tcp 139/tcp 445/tcp 901/tcp 1025/tcp	Open Open Open Open Open	Loc-srv Netbios-ssn Microsoft-ds Smaba-swat Listen
NFR sensor (MY.COMP.NET.29)	2010/tcp	Open	Search

Table-1.8 Scanning results for RealSecure and NFR sensors

As shown from table, NFR opens fewer ports (only one, in fact) on the sensor computer. RealSecure does not open as many as a normal NT installation does. However, some hardening work still need to be done after the sensor installation. Helpfully, ISS Inc. provides also an online computer security hardening guideline for the RealSecure post-

installation actions.

Noted that normally, this feature is not provided with the IDS in the public domain. So, when installing the IDS in public domain, we need hardening the computer on which such a system is installed.

1.4. Conclusions and Recommendations

Each of the reviewed IDSs (RealSecure, NFR, Shadow, and Snort) has its own strength and weaknesses. These IDSs have been evaluated with regards to the 9 features mentioned at the introduction of this paper. We note that due to resource limitation, we have not evaluated the performance for these IDSs. We hope to be able to do so in the near future.

All of the IDSs are able to do their job to certain extends. ISS RealSecure remains in the leading position in terms of broadness of detection range, in our opinion. However, other tools show their strength at price and attack signature customizations. This latter point is very important because when an IDS user has the ability to customize the attack signatures which are appropriate to the particular networking environment, he/she can avoid a lot of false alarms produced by the IDS. Furthermore, broad detection range, sometimes, might be not necessary if the IDS monitors a network segment that is protected from many attacks by other security mechanisms or tools, for example, on an internal network behind a correctly configured firewall.

Based up the finding in this paper, I would recommend, firstly, that we use RealSecure for relative bigger and busier network segments. Some strategically critical points may use shadow as a complementary security enforcement, especially using its raw data of tcpdump format for post analysis. (RealSecure 7.0 has a new feature for packet inspection. At the time of doing this experiment, it's just not possible to do so with old versions of RealSecure.) The reason for this is that Shadow sensor sniffs the traffic and dumps the traffic to the detailed log files. These detailed tcpdmp files can be valuable for post analysis of the incidents which might not be detected by 'real-time' sensor, like RealSecure, and NFR.

Secondly, for some small and relatively well protected networks, it suffice to use NFR and/or Snort. It should be noted that Snort is really a flexible tool in terms of filter rules customizations. It does not require a lot of knowledge for doing so. It just requires to understand the TCPDUMP format and the very simple syntax of Snort rules. Most importantly, since it is an open source software, a lot of developments and contributions comes from the public. This leads Snort having its attack signature coverage really in time and scalable.

One final note we would like to raise is that the real powerfulness of an IDS relies on how to use it effectively by its users. In order to reduce false alarms produced by an IDS, it requires the users having a good knowledge of the IDS itself and knowing how to customize its signatures database, alert mechanisms, and report mechanisms. It is more difficult with commercial products. One needs more particular training to be able to make full of use these commercialized products. Furthermore, since they are not open sourced, you cannot really accomplish this job as with other open source IDSs.

References

[1] http://secinf.net/info/ids/intrusion ICSA Inc. "Introduction to Intrusion Detection & Assessment - For System & Network Security Management" 1999
[2] http://www.snort.org/ – An open source signature based network intrusion detection system.
[3] http://www.nfr.com - NFR home page
[4] Northcutt, Stephen and Novak, Judy Network Intrusion Detection An Analyst's Handbook Second Edition New Riders 2001. P203-213
[5] Ranum, Marcus J. "Coverage in Intrusion Detection Systems" 6 June 2001
URL: http://philby.ucsd.edu/~cse291_IDVA/papers/orig_names/Coverage-in-IDS-White-Paper-final.pdf (16 Jan. 2002)
[6] Tanase, Matthew "The Future of IDS" 4 Dec. 2001
URL: http://www.securityfocus.com/infocus/1518 (16 Jan. 2002)
[7] ITL Bulletin "Acquiring and Deploying Intrusion Detection Systems" Nov. 1999.
URL: http://www.itl.nist.gov/lab/bulletns/nov99.htm (16 Jan. 2002)
[8] http://documents.iss.net/whitepapers/nva.pdf
[9] SANS Reading Room: http://rr.sans.org/intrusion/intrusion_list.php "How to Choose Intrusion Detection Solution" by Baiju Shah July 24, 2001
[10] SANS Reading Room: http://rr.sans.org/intrusion/selecting.php "Selecting an Intrusion Detection System" by Kathleen Buonocore August 19, 2001
[11] http://documents.iss.net/literature/RealSecure/RS_FAQ.pdf
[12] Shadow home page http://www.nswc.navy.mil/ISSEC/CID
[13] ISS home page http://www.iss.net
[14] http://www.silicondefense.com/index.htm

Assignment 2 – Network Detects

Table of Contents

2.1. Introduction

This part of assignment consists of three network detects. The first detect is from the web site http://www.ncidents.org/logs/Raw assigned by SANS, and the other two are from my home network as shown in Fig 2.1.

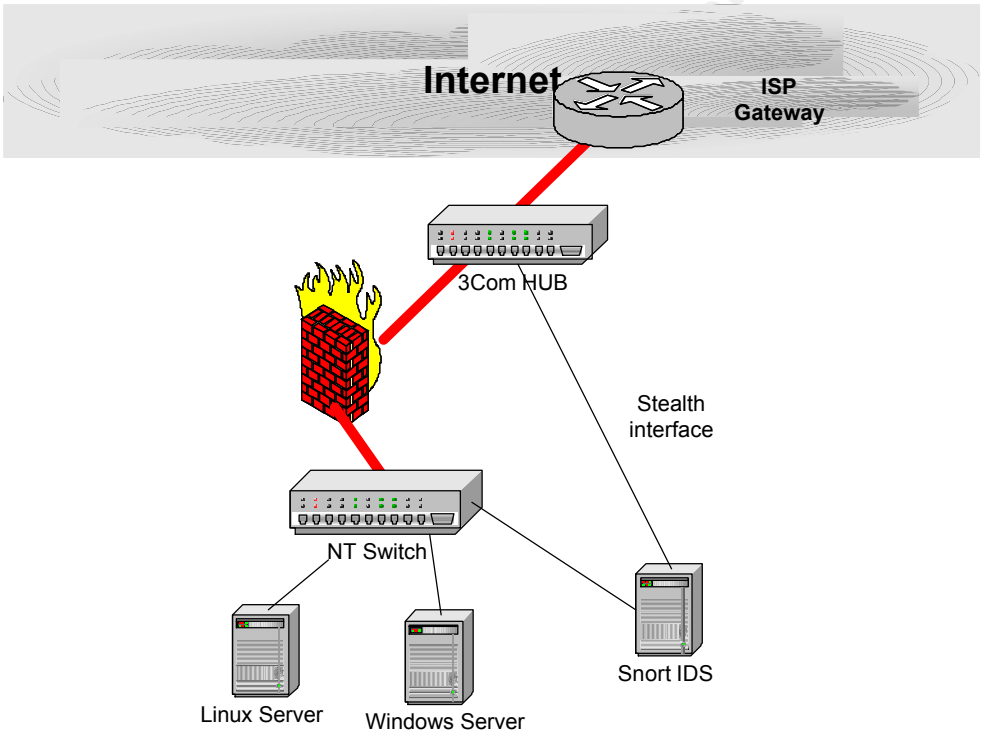


Fig. 2.1. MY.HOME network diagram

Nowadays, there is a lot of suspicious traffic circulating over the Internet. In order to learn effectively how to detect and analyze the attacks, we have to decide which attacks to be chosen among so many old and new ones. For learning purpose and by referring [11], I have chosen three detects in three categories:

1. Possible system abuse by buffer overflow (see Detect 1)
2. Distributed Denial of Service (DDoS) (see Detect 2)
3. Inadequate Argument Checking (see Detect 3)

2.2. DETECT 1 - SHELLCODE x86 inc ebx NOOP

2.2.1. Source of Trace

These packet traces are extracted from http://www.incidents.org/logs/Raw/2002.6.15. The target network is sanitized as SANS.Detect.x.x.

I first run Snort over the logged packets of tcpdump format. Then I run Snortsnarf against the Snort alert log file. The following are some excerpts from the Snortsnarf output.

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
07/15-07:11:53.894488 207.229.152.40:80 -> SANS.Detect.180.250:63637
TCP TTL:53 TOS:0x0 ID:52880 IpLen:20 DgmLen:1500 DF
AP Seq: 0x595491F3 Ack: 0x48802EA0 Win: 0x7D78 TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
07/15-07:36:22.034488 66.186.38.10:80 -> SANS.Detect.180.250:61142
TCP TTL:52 TOS:0x0 ID:2650 IpLen:20 DgmLen:1500 DF
AP Seq: 0xB70A56CB Ack: 0x12A55025 Win: 0x7D78 TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
07/15-07:54:23.964488 206.137.30.33:80 -> SANS.Detect.180.250:61764
TCP TTL:51 TOS:0x0 ID:50253 IpLen:20 DgmLen:1500 DF
A* Seq: 0x97AF7B60 Ack: 0xB431A4CD Win: 0x2238 TcpLen: 20

[**] [1:1390:3] SHELLCODE x86 inc ebx NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
07/15-08:30:41.014488 65.119.30.151:80 -> SANS.Detect.180.250:63425
TCP TTL:110 TOS:0x0 ID:63121 IpLen:20 DgmLen:1500 DF
A* Seq: 0x3D7B1D49 Ack: 0xCEE9D765 Win: 0x4344 TcpLen: 20

--- < snipped > ---

Excerpt of one detected packages with the payload is shown as follows:

07:11:53.894488 207.229.152.40.80 > SANS.Detect.180.250.63637: P 1498714611:1498716071(1460) ack 1216360096 win 32120 (DF)

```
0x0000   4500 05dc ce90 4000 3506 2c84 cfe5 9828      E.....@.5.,....(
0x0010   2e05 b4fa 0050 f895 5954 91f3 4880 2ea0      .....P..YT..H...
0x0020   5018 7d78 d012 0000 4854 5450 2f31 2e30      P.}x....HTTP/1.0
0x0030   2032 3030 204f 4b0d 0a43 6f6e 7465 6e74      .200.OK..Content
0x0040   2d54 7970 653a 2061 7070 6c69 6361 7469      -Type:.applicati
0x0050   6f6e 2f78 2d73 686f 636b 7761 7665 2d66      on/x-shockwave-f
0x0060   6c61 7368 0d0a 436f 6e74 656e 742d 4c65      lash..Content-Le
0x0070   6e67 7468 3a20 3239 3131 320d 0a4c 6173      ngth:.29112..Las
0x0080   742d 4d6f 6469 6669 6564 3a20 4672 692c      t-Modified:.Fri,
```

```
0x0090   2031 3520 4170 7220 3139 3934 2030 303a      .15.Apr.1994.00:
0x00a0   3030 3a30 3020 474d 540d 0a44 6174 653a      00:00.GMT..Date:
0x00b0   204d 6f6e 2c20 3135 204a 756c 2032 3030      .Mon,.15.Jul.200
0x00c0   3220 3132 3a31 333a 3132 2047 4d54 0d0a      2.12:13:12.GMT..
0x00d0   436f 6e6e 6563 7469 6f6e 3a20 6b65 6570      Connection:.keep
0x00e0   2d61 6c69 7665 0d0a 4578 7069 7265 733a      -alive..Expires:
0x00f0   2054 6875 2c20 3135 2041 7072 2032 3031      .Thu,.15.Apr.201
0x0100   3020 3230 3a30 303a 3030 2047 4d54 0d0a      0.20:00:00.GMT..
0x0110   0d0a 4657 5304 b871 0000 7800 0426 8000      ..FWS..q..x..&..
--- < snipped > ---
0x01b0   0f0f 1711 1725 1616 252f 241d 242f 2c24      .....%../$.$/,$
0x01c0   2323 242c 3a32 3232 3232 3a43 3d3d 3d3d      ##$,:22222:C====
0x01d0   3d3d 4343 4343 4343 4343 4343 4343 4343      ==CCCCCCCCCCCCCC
0x01e0   4343 4343 4343 4343 4343 4343 4343 4301      CCCCCCCCCCCCCCC.
0x01f0   1417 171e 1a1e 2418 1824 3324 1e24 3342      ......$..$3$.$3B
0x0200   3329 2933 4243 423e 323e 4243 4343 4343      3))3BCB>2>BCCCCC
0x0210   4343 4343 4343 4343 4343 4343 4343 4343      CCCCCCCCCCCCCCCC
0x0220   4343 4343 4343 4343 4343 4343 4343 4343      CCCCCCCCCCCCCCCC
0x0230   ffc0 0011 0800 c803 5303 0122 0002 1101      ........S..".....
0x0240   0311 01ff c400 8800 0002 0301 0100 0000      ................
0x0250   0000 0000 0000 0000 0304 0002 0501 0601      ................
--- < snipped > ---
0x0530   5630 c39b 0339 809f 65f8 8b29 c6cb 7ce0      V0...9..e..)..|.
0x0540   6548 e40c e5dd 8505 a885 974a c7ab 0b6f      eH.........J...o
0x0550   5669 eaf2 2ee2 e3b9 d545 55c6 4859 b1bf      Vi.......EU.HY..
0x0560   afed 3a2c 597f abdc f535 cb92 c9e2 8f4d      ..:,Y....5.....M
0x0570   f537 bb29 59dd 6873 7ed5 31dd 7f26 abf0      .7.)Y.hs~.1..&..
0x0580   6a24 472c 6c57 927a 0394 f079 e914 4c17      j$G,lW.z...y..L.
0x0590   6167 542d 1b5a d18c ca59 12b7 72f4 e218      agT-.Z...Y..r...
0x05a0   43f3 c9de 4c04 5e4b e42d 182f 32ae 40f2      C...L.^K.-./2.@.
0x05b0   5652 3241 3b2b 05ac 9fec bf27 6766 a2b6      VR2A;+.....'gf..
0x05c0   d98d 4bd2 a693 d441 e034 58bd 5252 8a6b      ..K....A.4X.RR.k
0x05d0   d02a 91e6 d918 6232 c6a7 5c9b                .*....b2..\.
```

2.2.2. Detect was generated by

The Snort alert and log data are generated by Snort IDS of 1.8.7 version with the Snort 1.8.7 rules set. Please see http://www.snort.org for detailed interpretation of the Snort alert data.

2.2.3. Probability the source address was spoofed

By looking closely into the logs shown above, Time stamp, TCP sequence numbers and acknowledgement numbers are all normal. There is no sign of packet crafting as the IP numbers, since this attack event is a part of established legitimate http session (we see all incoming packets are from port 80) and a TCP 3-way handshaking should have already taken before. As such, the probability of spoofing is low.

2.2.4. Description of Attack

An attacker with some carefully crafted shellcodes aims at getting a command line prompt

with system privileges. As everyone knows what it means if some one gains a "root" privilege on a Unix box.

Shellcodes are the binary equivalent of assembler commands. They are mostly used in buffer overflow exploits. These exploits send excessive data into a program buffer so that the program cannot handle it. As a consequence, the program may change its normal execution path and return an address to those instructions that can spawn a command shell (e.g. /bin/sh in Unix systems).

More specifically, SHELLCODE x86 inc ebx NOOP code is a generic alert that the IDS detected 0x43 characters in the payload of a packet to and from your network. Typically it refers to as a NOOP SLED. There are a series of character sets that an attacker can use to make a Noop Sled. The most used Noop Sled is 0x90 (see [12]). If you take a look at the Snort shellcode.rules there is a series of Snort rules established for this type of attacks.

2.2.5. Attack Mechanism

It is common technique for an attacker to write up an unchecked user input beyond buffer boundary so as to make buffer overflow. As described in [13], one of the buffer overflow strategies is to put the shellcodes as/for program preclude, relative addressing to the shell command string (i.e. /bin/sh) plus the shell command string itself within the buffer and to change the return address pointing to somewhere in the buffer.

There is more than one ways to perform a NOOP on a processor. Opcodes like 'inc ebx' or many more others do not perform any useful operation before the shell command [1].

2.2.6. Correlations

A) Other report on using the exact Noop Sled

Buffer overflow attacks using a series of 0x43 has been reported in exploiting FTPD and Sniffit vulnerabilities at http://cert.uni-stuttgart.de/archive/bugtraq/2001/08/msg00177.html and http://www.securiteam.com/exploits/5RP0O0K60O.html.

B) Looking into who are the top talkers:

In file 2002.6.15, there are 29 IPs which sent this kind of packets, In order to know who these 'guys' are and whether they are known 'bad guys', I have used the whois service at http://www.arin.net to the following three addresses: 207.229.152.40, 66.186.38.10, and 206.137.30.33.

ARIN Search for: 207.229.152.40

```
RCN Corporation RCN-BLK-22 (NET-207-229-128-0-1)
                                  207.229.128.0 -
```

```
207.229.191.255
Brand X FilmWorks EACT-CUST-DS0-BX (NET-207-229-152-0-1)
                                    207.229.152.0 -
207.229.152.63
```

Nslookup 207.229.152.40:
Server: dns.ym.rnc.net.cable.rogers.com
Address: 24.153.22.195

Name: a207-229-152-40.deploy.akamaitechnologies.com
Address: 207.229.152.40

ARIN Search for: 66.186.38.10:

```
OrgName:    Cable & Wireless
OrgID:      EXCW

NetRange:   66.186.32.0 - 66.186.47.255
CIDR:       66.186.32.0/20
NetName:    NY1-5
NetHandle:  NET-66-186-32-0-1
Parent:     NET-66-0-0-0-0
NetType:    Direct Allocation
--- Snipped  ---
RegDate:
Updated:    2002-08-21

TechHandle: ZC221-ARIN
TechName:   Cable & Wireless
TechPhone:  +1-919-465-4023
TechEmail:  ip@gnoc.cw.net

OrgAbuseHandle: ABUSE11-ARIN
OrgAbuseName:   Abuse
OrgAbusePhone:  +1-877-393-7878
OrgAbuseEmail:  abuse@exodus.net
```

ARIN Search for: 206.137.30.33

```
OrgName:    UUNET Technologies, Inc.
OrgID:      UU

NetRange:   206.136.0.0 - 206.139.255.255
CIDR:       206.136.0.0/14
NetName:    NETBLK-UUNETCBLK136
NetHandle:  NET-206-136-0-0-1
Parent:     NET-206-0-0-0-0
NetType:    Direct Allocation
NameServer: AUTH00.NS.UU.NET
---Snipped---
TechHandle: OA12-ARIN
TechName:   UUNet, Technologies
TechPhone:  +1-800-900-0241
TechEmail:  help@uu.net

OrgAbuseHandle: ABUSE3-ARIN
OrgAbuseName:   abuse
```

```
OrgAbusePhone:  +1-800-900-0241
OrgAbuseEmail:  abuse-mail@wcom.com
```

> nslookup 206.137.30.33
Name: gerberfurniture.com
Address: 206.137.30.33
ARIN Search for: 65.119.30.151

As we see, most of these talkers are from IP addresses managed by ISPs.

C) It is reported that a lot of attacks like the one shown here are likely false positive. If you download some gif files containing the same data patterns from the web servers on the Internet, you can get the same alert from Snort IDS.

In order to eliminate the kind of false positive to certain extend, newer version of Snort have excluded web traffic for this rule to eliminate false positives.

D) For more detail about this attack, some references are worthwhile to read: [14],[15].

2.2.7. Evidence of Active Targeting

I have made a thorough search for the evidence of compromised targets by looking into the payload of the all related packets. I did not find any shell command or prompt. I also did not find that the attacks destined to the ports that are associated with any well-known services or trojans. If the talkers in this attack are legitimate web servers, this attack could be most likely false positive in this case since the image packets can contain this pattern with high probability.

2.2.8. Severity

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

Criticality 5:
By looking into the raw data file, I try to determine the criticality for the destination host SANS.Detect.180.250. It seems that it is not only a web server, but also a DNS server. Assume this is the case, the host should be a critical one. So a value 5 is assigned.

Lethality value 3:
I assume that the host SANS.detect.180.250 is located in the DMZ behind a firewall/proxy server. Assume there is the firewall which protects the network as a whole.
In view of the consequence of this attack if it is successful, this value could be assigned a highest one, i.e. 5. However, I assign this value to 3 since I am not sure whether it is false positive or not (see above analysis).

System Countermeasures 2:
Here I assume the firewall/proxy server is properly configured and the host itself is

appropriately hardened. There should have minimum the defensive mechanisms and security configurations in place. So the value is 2.

Network Countermeasures 2:

Obviously, there is a network-based IDS is installed outside the firewall/proxy server to capture the network detects. Assume that the firewall/proxy server is properly configured, a value of 2 is assigned.

Therefore, Severity = (5 + 3) - (2 + 2) = 4.

2.2.9. Defensive Recommendation

The following defensive measures are recommended for prevention purposes:

- Be informed by subscribing to security vulnerability notification or advisory from www.securityfocus.com or www.cert.org
- Timely apply the appropriate patches to the systems.
- Set up content filtering at firewall level to drop those packets containing a string to spawn a command shell.
- Adjust the network-based IDS rule to log and terminate those connections when found.

2.2.10. Multiple Choice Test Question

It is well known that there is a series of carefully crafted "SHELLCODES". They are mostly used for:

A. Distributed Denial of Service (DDOS)
B. System Reconnaissance
C. Trojan Horse
D. Buffer Overflow

Answer: D

2.3. DETECT 2 - DDOS shaft agent to handler

The target network in the next two detects are sanitized as MY.HOME.NET.x.

[**] [1:240:1] DDOS shaft agent to handler [**]
[Classification: Attempted Denial of Service] [Priority: 2]
08/29-18:40:14.240764 24.57.122.37:35169 -> MY.HOME.NET.189:20433
UDP TTL:186 TOS:0x0 ID:15420 IpLen:20 DgmLen:87
Len: 67
[Xref => http://www.whitehats.com/info/IDS256]

2.3.1. Source of Trace

From my home network (see Fig.1)

2.3.2. Source of Detect

This detect was generated by a Snort IDS with the Snort 1.8.7 rule set. Specifically, the detect was generated by the following Snort rule:

```
alert udp $EXTERNAL_NET any -> $HOME_NET 20433 (msg:"DDOS shaft agent
to handler"; content: "alive"; reference:arachnids,256;
classtype:attempted-dos; sid:240; rev:1;)
```

2.3.3. Probability the source address was spoofed

Shaft uses UDP protocol, so the source IP can be easily spoofed. In this case, the attacker as a handler (e.g. 24.57.122.37) seems expecting for agent's response (if MY.HOME.NET.189 has an active agent installed by previous attack). If the source address was spoofed the attacker would not receive any confirmation that the agent is ready. So, in this case the probability is low.

2.3.4. Description of Attack

Shaft, like **Stacheldraht**, **Mstream**, and **Trinity** are four known distributed denial-of-service (DDOS) tools, which consist of a **handler** and many **agents**. Shaft not like others DDOS, it communicates using UDP. It can make UDP flood, TCP SYN flood, or ICMP flood attacks. Like other DDOS, the attacker uses telnet as a client to communicate with the handler. After some controlling communications, the handler can command one or more agents to launch packet flooding or other attacks against victims.

From the below a cut-pasted packet with the payload, we can see that the handler sent out an "alive" message along with the default password "tijgu" to agent. Here the password is a way to keep others from taking over the handler.

```
18:15:19.894107 24.57.122.37 > MY.HOME.NET.189: truncated-udplength 0
0x0000   4500 00b4 3079 0000 c411 5b25 4130 16ba      E...0y....[%A0..
0x0010   c7f3 4abd b835 0000 0000 0000 c7f3 4abd      ..J..5........J.
0x0020   5038 477d 891d 0000 0000 0000 0000 0000      P8G}............
0x0030   0000 0000 0000 0000 0000 0000 616c 6976      ............aliv
0x0040   6520 7469 6a67 7500 0000 0000 0000 0000      e.tijgu.........
0x0050   0000 0000 0000 0000 0000 0000 0000 0000      ................
0x0060   0000 0000 0000 0000 0000 0000 0000 0000      ................
0x0070   0000 0000 0000 0000 0000 0000 0000 0000      ................
0x0080   0000 0000 0000 0000 0000 0000 0000 0000      ................
0x0090   0000 0000 0000 0000 0000 0000 0000 0000      ................
0x00a0   0000 0000 0000 0000 0000 0000 0000 0000      ................
0x00b0   0000 0000                                    ....
```

A detailed analysis of Shaft can be found at [3].

2.3.5. Attack Mechanism

This alert does not indicate a successful attack, but an attempt of a Shaft handler server (24.57.122.37) to check for the status of, or existence of a Shaft agent (MY.HOME.NET.189). If the Shaft agent exists it is usually the result of a previous attack target. Attackers scan networks for systems that are vulnerable to remote buffer overrun exploits and use those exploits to set up Shaft network as follows:

Client(s)→handler(s)→agent(s)→victim(s).

The Shaft uses different ports for the communications between two Shaft network components:

Client to handler(s):	20432/tcp
Handler to agent(s):	18753/udp
Agent to handler(s):	20433/udp

Here a client is usually a telnet program used by the attacker to communicate with the handler server. The Shaft has a noticeable feature that it can switch handler servers and handler ports on the fly. This feature makes an IDS difficult to detect the communications between the attacker and a handler. If you open Snort DDOS rules files, you will find that there is only one rule for detecting the attacking traffic between the handler and the agent. There is no rule for detecting the Shaft communication between a client and a handler.

2.3.6. Correlation

A) ARIN Search result for 24.57.122.37:

```
Cogeco Cable Inc. CGOC-3BLK (NET-24-57-0-0-1)
                                  24.57.0.0 - 24.57.127.255
Cogeco Cable Solutions CGOC-WITE1-2 (NET-24-57-112-0-1)
                                  24.57.112.0 -
24.57.127.255
```

nslookup 24.57.122.37:
Server: dns.ym.rnc.net.cable.rogers.com
Address: 24.153.22.195

Name: d57-122-37.home.cgocable.net
Address: 24.57.122.37

It is obvious that this is a home cable user.

B) I have not found any report on 24.57.122.37 that used Shaft or others DDOS activities to attack the Internet. One can find the use of other DDoS attacks in CERT incident note IN-99-07. (http://www.cert.org/incident_notes/IN-99-07.html). The Shaft came out at the end of 1999. It is reported that Shaft is one of Distributed Denial of Service attacks that had been responsible for the interruption for a number of well known networks including, CNN, ZDNet, Amazon etc. in the first two months of 2000.

2.3.7. Evidence of active targeting

There is no prior history of Shaft activity to or from the host on my home network. The firewall policy will do work for denying connection attempts of this type. This makes it highly unlikely that this host was specifically targeted. Most likely, this is a person who tried to use this tool to scan networks in the little hope of a pre-existing shaft agent to take over.

2.3.8 Severity

- **Criticality: 5**
  The target system is the firewall itself since there is only one valid IP address for a home cable user. If the firewall is compromised the whole internal network is open to attack.
- **Lethality: 4**
  If this attack succeeds, it produces a serious consequence. However, according to the above analysis, this case is not considered a real attack. No harm has been done. So one point is deduced from the highest one.
- **Severity, System Countermeasures: 5**
  The firewall system with the targeting IP is an appliance with original security considerations for its hardware design (see www.netscreen.com). Furthermore, it offers no other services and is configured by myself with confidence.
- **Severity, Network Countermeasures: 5**
  The whole network system is firewall protected with its appropriate rules. All the internal boxes use invalid IP addresses. The incoming traffic of the whole network is monitored by a Snort IDS.

(5 + 4) – (5 + 5) = -1

2.3.9. Defensive Recommendations

A number of actions can be taken to help prevent the systems from being used as a DDOS agent for attackers.

- Update the knowledge of how to counter against the Shaft and other types of DDOS by review references [2] and [3].
- Keep systems up to date with security patches to prevent compromises that will allow DDoS tools from being installed.
- DDoS attacks usually use spoofed source addresses in the packets. It is a good idea to establish an egress filter to prevent network traffic with spoofed source addresses from leaving your network.
- Rate-limiting described in [10] is an effective mechanism against ICMP packet flooding by the Shaft attacks.

2.3.10. Multiple choice test question

Shaft uses which port for the communications between a handler and an agent?

a) 20433/TCP
b) 18753/UDP
c) 20433/UDP
d) 65535/TCP

Answer C: By default, the Shaft uses 20433/UDP for the communications between handler and agent. The communication port between attacker and handler can be changed on the fly.

2.4. DETECT 3 - web-cgi_http-cgi-formmail

[**] [1:884:6] WEB-CGI formmail access [**]
[Classification: access to a potentially vulnerable web application] [Priority: 2]
08/16-05:00:35.434488 24.57.122.37:3035 -> MY.HOME.NET.189:80
TCP TTL:109 TOS:0x0 ID:57069 IpLen:20 DgmLen:542 DF
AP Seq: 0xFDB44053 Ack: 0xAF9EDCC4 Win: 0x2530 TcpLen: 20
[Xref => http://www.securityfocus.com/bid/1187]
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0172]
[Xref => http://www.whitehats.com/info/IDS226]

[**] [1:884:6] WEB-CGI formmail access [**]
[Classification: access to a potentially vulnerable web application] [Priority: 2]
08/16-05:01:06.694488 24.57.122.37:3150 -> MY.HOME.NET.189:80
TCP TTL:109 TOS:0x0 ID:57614 IpLen:20 DgmLen:542 DF
AP Seq: 0xFE498023 Ack: 0xB0EB9F7A Win: 0x2530 TcpLen: 20
[Xref => http://www.securityfocus.com/bid/1187]
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0172]
[Xref => http://www.whitehats.com/info/IDS226]

[**] [1:884:6] WEB-CGI formmail access [**]
[Classification: access to a potentially vulnerable web application] [Priority: 2]
08/16-05:01:41.054488 24.57.122.37:3252 -> MY.HOME.NET.189:80
TCP TTL:109 TOS:0x0 ID:58232 IpLen:20 DgmLen:542 DF
AP Seq: 0xFEE85DAA Ack: 0xB37789D6 Win: 0x2530 TcpLen: 20
[Xref => http://www.securityfocus.com/bid/1187]
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0172]
[Xref => http://www.whitehats.com/info/IDS226]

The output from the command "TCPDUMP –r tcpdump.log –Xn"

05:00:35.434488 24.57.122.37.3035 > MY.HOME.NET.189.80: P 4256448595:4256449097(502) ack 2946424004 win 9520 (DF)
0x0000 4500 021e deed 4000 6d06 3278 a579 7876 E.....@.m.2x.yxv
0x0010 2e05 b485 0bdb 0050 fdb4 4053 af9e dcc4 ....P..@S....
0x0020 5018 2530 78f8 0000 504f 5354 202f 6367 P.%0x...POST./cg
0x0030 692d 6269 6e2f 466f 726d 4d61 696c 2e63 i-bin/FormMail.c

```
0x0040   6769 2048 5454 502f 312e 300d 0a52 6566     gi.HTTP/1.0..Ref
0x0050   6572 6572 3a20 6874 7470 3a2f 2f77 7777     erer:.http://www
0x0060   2e58 5858 582e 636f 6d0d 0a55 7365 722d     .XXXX.com..User-
0x0070   4167 656e 743a 204d 6f7a 696c 6c61 2f34     Agent:.Mozilla/4
0x0080   2e30 3620 2857 696e 3935 3b20 4929 0d0a     .06.(Win95;.I)..
0x0090   436f 6e6e 6563 7469 6f6e 3a20 4b65 6570     Connection:.Keep
0x00a0   2d41 6c69 7665 0d0a 486f 7374 3a20 7777     -Alive..Host:.ww
0x00b0   772e 5858 5858 2e63 6f6d 3a38 300d 0a43     w.XXXX.com:80..C
0x00c0   6f6e 7465 6e74 2d74 7970 653a 2061 7070     ontent-type:.app
0x00d0   6c69 6361 7469 6f6e 2f78 2d77 7777 2d66     lication/x-www-f
0x00e0   6f72 6d2d 7572 6c65 6e63 6f64 6564 0d0a     orm-urlencoded..
0x00f0   436f 6e74 656e 742d 6c65 6e67 7468 3a20     Content-length:.
0x0100   3230 390d 0a41 6363 6570 743a 2069 6d61       209..Accept:.ima
0x0110   6765 2f67 6966 2c20 696d 6167 652f 782d     ge/gif,.image/x-
0x0120   7862 6974 6d61 702c 2069 6d61 6765 2f6a     xbitmap,.image/j
0x0130   7065 672c 2069 6d61 6765 2f70 6a70 6567     peg,.image/pjpeg
0x0140   2c20 2a2f 2a0d 0a0d 0a65 6d61 696c 3d43     ,.*/*....email=C
0x0150   6172 626f 7840 5369 6c6c 7941 7373 5363     arbox@SillyAssSc
0x0160   616e 6e65 722e 676f 7626 7265 6369 7069     anner.gov&recipi
0x0170   656e 743d 6372 6974 696b 3034 4061 6f6c     ent=critik04@aol
0x0180   2e63 6f6d 2673 7562 6a65 6374 3d46 6f72     .com&subject=For
0x0190   6d6d 6169 6c25 3230 2877 7777 2e58 5858     mmail%20(www.XXX
0x01a0   582e 636f 6d2f 6367 692d 6269 6e2f 466f     X.com/cgi-bin/Fo
0x01b0   726d 4d61 696c 2e63 6769 2926 4162 6f75     rmMail.cgi)&Abou
0x01c0   743d 5369 6c6c 7925 3230 4173 7325 3230     t=Silly%20Ass%20
0x01d0   466f 726d 2532 304d 6169 6c25 3230 5363     Form%20Mail%20Sc
0x01e0   616e 6e65 7225 3230 4279 2532 3043 6172     anner%20By%20Car
0x01f0   626f 7826 7369 7465 3d77 7777 2e58 5858     box&site=www.XXX
0x0200   582e 636f 6d2f 6367 692d 6269 6e2f 466f     X.com/cgi-bin/Fo
0x0210   726d 4d61 696c 2e63 6769 0d0a 0d0a         rmMail.cgi....
```

2.4.1. Source of trace:

My home network (see fig.1).

2.4.2. Detect was generated by:

This detect was generated by SNORT IDS with Snort rules set 8.17. Specifically the following is the Snort rule:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"WEB-CGI
formmail access";flags:A+; uricontent:"/formmail"; nocase;
reference:bugtraq,1187; reference:cve,CVE-1999-0172;
reference:arachnids,226; classtype:web-application-activity; sid:884;
rev:6;)
```

2.4.3. Probability the source address was spoofed

As indicated at www.whitehats.com's IDS database: http://www.whitehats.com/IDS/226, the packet that triggered this event is normally a part of an established TCP session, indicating that the source IP address has not been spoofed. In this case, although the intruder did not reveal his/her email identity or address, the IP address of the web server

with the FormMail script could not be hidden to the IDS. For this reason, the source address was not likely spoofed.

2.4.4. Description of attack

Within 10 minutes, the alerts showed up 18 connection attempts to port 80. The attacker is attempting to exploit a FormMail.pl cgi-script by modifying the recipient and message parameters and thus allowing him/her to send anonymous e-mail to the victim since the e-mail will not indicate any sender.

Two source IP's were detected in the alerts:
24.57.122.37 18 occurrences
64.156.206.18 2 occurrences

Search 24.57.122.37 at www.arins.net/whois revealed the following:

```
Cogeco Cable Inc. CGOC-3BLK (NET-24-57-0-0-1)
                                  24.57.0.0 - 24.57.127.255
Cogeco Cable Solutions CGOC-WITE1-2 (NET-24-57-112-0-1)
                                  24.57.112.0 -
24.57.127.255
nslookup 24.57.122.37:
Server:  dns.ym.rnc.net.cable.rogers.com
Address:  24.153.22.195

Name:    d57-122-37.home.cgocable.net
Address:  24.57.122.37
```

Search 64.156.206.18 at www.arins.net/whois revealed the following:

```
OrgName:    Level 3 Communications, Inc.
OrgID:      LVLT

NetRange:   64.152.0.0 - 64.159.255.255
CIDR:       64.152.0.0/13
NetName:    LC-ORG-ARIN
NetHandle:  NET-64-152-0-0-1
Parent:     NET-64-0-0-0-0
NetType:    Direct Allocation
NameServer: NS1.LEVEL3.NET
NameServer: NS2.LEVEL3.NET
Comment:    ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate:    2000-06-08
Updated:    2001-05-30

TechHandle: LC-ORG-ARIN
TechName:   level Communications
TechPhone:  +1-877-453-8353
TechEmail:  ipaddressing@level3.com
```

Nslookup 24.153.22.195
Server: dns.ym.rnc.net.cable.rogers.com
Address: 24.153.22.195

Name: dialup-64.156.206.18.Dial1.SanDiego1.Level3.net
Address: 64.156.206.18

It seems that both "intruders" are dailup or cable modem ISP users.

2.4.5. Attack mechanism:

FormMail of the original version was a Perl script written by Matt Wright (http://-worldwidemart.com/scripts-/formmail.shtml) in July 9, 1995. The actual version 1.9 of FormMail was released by its author on August 3, 2001.

By manipulating inputs to the FormMail CGI script and taking advantage of problems in the scripts, i.e. exploiting the way addresses are verified and inputs are checked, remote users (mostly spammers) may abuse the functionality provided by FormMail to cause the local mail server on the same (web) server system to send arbitrary e-mail messages to arbitrary e-mail destination addresses. The script is designed to accept variables in HTML form and mail them to a specified email address. FormMail comes with many formatting and operational options that can be specified through the form via HTTP variables. This feature greatly facilitates users' CGI access without having the extensive programming knowledge.

FormMail uses an HTTP variable to specify the destination e-mail address, and this allows spammers to use this script to distribute their messages to specified recipients. This vulnerability can be exploited with a web browser. Furthermore, the script relies on an http variable for the source address as well which allows for sending e-mail with no source address or forge e-mails.

- More information about how this attack works can be found at: At bugtraq see the security advisory of 23-Jan-2002 by Ronald Guilmette "Anonymous Mail Forwarding Vulnerabilities in FormMail 1.9" at http://online.securityfocus.com/archive/1/252232
- More information about this vulnerability can be found at: cve.mitre.org site, and the CVE number is: CAN-2001-0357 (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0357.

2.4.6. Correlations

A) www.securityfocus.com rates the formmail attack as the third most common in the online world for the first quarter of 2002. (see http://www.securityfocus.com-/corporate/research/top10attacks_q1_2002.shtml)

B) Unfortunately, apache web server log on MY.HOME.NET was not turned at the time this detect was recorded. However, some formmail.pl scanning activities detected by web server log are reported at:
http://citadelle.intrinsec.com/mailing/current/HTML/ml_mobile_code/0487.html. The following log entries are copied from the discussion mail from the above URL.

The "formmail.pl" mail gateway is already being scanned for:

172.144.145.117 - - [29/Nov/2001:13:33:47 -0600] "GET
/cgi-
bin/formmail.pl?email=f2%40aol%2Ecom&subject=www%2Edigitaloffense%2Enet%2Fcgi%2Dbin%2Fformmail%2Epl&recipient=formmail%20sn%40aol%2Ecom&msg=w00t
HTTP/1.1Content-Type: application/x-www-form-urlencoded" 404 238 "-"
"Gozilla/4.0 (compatible; MSIE 5.5; windows 2000)"
24.92.238.75 - - [29/Nov/2001:19:27:14 -0600] "GET
/cgi-
bin/formmail.pl?email=f2%40aol%2Ecom&subject=www%2Edigitaloffense%2Enet%2Fcgi%2Dbin%2Fformmail%2Epl&recipient=af088%40hotmail%2Ecom&msg=w00t
HTTP/1.1Content-Type: application/x-www-form-urlencoded"
404 238 "-" "Gozilla/4.0 (compatible; MSIE 5.5; windows 2000)"
138.89.88.102 - - [01/Dec/2001:20:44:31 -0600] "GET
/cgi-
bin/formmail.pl?email=f2%40aol%2Ecom&subject=www%2Edigitaloffense%2Enet%2Fcgi%2Dbin%2Fformmail%2Epl&recipient=nofxdan%40hotmail%2Ecom&msg=w00t
HTTP/1.1Content-Type: application/x-www-form-urlencoded" 404 238 "-"
"Gozilla/4.0 (compatible; MSIE 5.5; windows 2000)"
207.14.189.11 - - [03/Dec/2001:00:31:34 -0600] "GET
/cgi-
bin/formmail.pl?email=f2%40aol%2Ecom&subject=www%2Edigitaloffense%2Enet%2Fcgi%2Dbin%2Fformmail%2Epl&recipient=knight827%40earthlink%2Enet&msg=w00t
HTTP/1.1Content-Type: application/x-www-form-urlencoded" 404 238 "-"
"Gozilla/4.0 (compatible; MSIE 5.5; windows 2000)"
24.49.93.29 - - [03/Dec/2001:15:41:45 -0600] "GET
/cgi-
bin/formmail.pl?email=f2%40aol%2Ecom&subject=www%2Edigitaloffense%2Enet%2Fcgi%2Dbin%2Fformmail%2Epl&recipient=guiltybiz%40aol%2Ecom&msg=w00t
HTTP/1.1Content-Type: application/x-www-form-urlencoded" 404 238 "-"
"Gozilla/4.0 (compatible; MSIE 5.5; windows 2000)"

C) It is interesting to read a discussion mail about formmail attack at http://online.securityfocus.com/archive/75/250141. In this email, the sender Mike Lewinski <mike@rockynet.com> displayed two example detects. The one is a failed formmail probe and the other a successful formmail relay:

```
1) Failed probe:

GMT offset is -0700. This is a probe for a formmail.pl cgi script that
can
be used to relay spam. It generated a 404 here.

Session Details
  IP Address    65.34.109.21
  Reverse DNS    6534109hfc21.tampabay.rr.com
 Time Spent   0 min
  Hits / Kilobytes    1 / 0.61Kb
 Browser Tag   Gozilla/4.0 (compatible; MSIE 5.5; windows 2000)
 Referring URL

Date and Time URL
 2002-01-07 19:20:24
/cgi-
```

```
bin/formmail.pl?email=f2%40aol%2ecom&subject=www%2ecoloradowild%2eorg%
2
fcgi%2dbin%2fformmail%2epl&recipient=bxw%40aol%2ecom&msg=w00t
--------------------------------------------------------------------------

2) Successful relays:

The log times below are set to UTC, and were recorded on Jan 01, 2001.
Also
attached is a sample of the bounced spam that was relayed through this
client's script (now disabled).

00:52:59 63.199.200.93 POST /cgi-bin/formmail.pl - 502 564 343 80
Microsoft+URL+Control+-+6.00.8862 -
00:52:59 63.199.200.93 POST /cgi-bin/formmail.cgi - 200 10590 345 80
Microsoft+URL+Control+-+6.00.8862 -
13:17:51 66.125.153.7 POST /cgi-bin/formmail.cgi - 200 9515 1737 80
Microsoft+URL+Control+-+6.00.8862 -
21:07:30 66.125.153.7 POST /cgi-bin/formmail.cgi - 200 11401 1182 80
Microsoft+URL+Control+-+6.00.8862 -
21:15:23 66.125.153.7 POST /cgi-bin/formmail.cgi - 200 11562 1495 80
```

--- Snipped ---

2.4.7. Evidence of active targeting:

There is no formmail.pl installed on MY.HOME.NET. This case seems to be a scan of all known hosts looking for the formmail.pl script. There is no evidence showing that there was a reconnaissance scan processed previously by using some known scan tools.

2.4.8. Severity:

Criticality: 3
This attack is looking for a hole on the Formmail.pl script located on web servers. My web is behind my firewall. In this case the criticality value is assigned to 3.
Lethality: 2
The attack was actually to the firewall. The likelihood of the attack success is low. Furthermore, the attack succeeds, there is no direct physical damage to the local system. So 2 is chosen for the value.
System: 4
The systems on my internal network are with modern operating systems with all the latest patches installed. An Apache server is installed on Linux Mandrake 8.2. There is no Formail.pl installed. Furthermore it is behind the firewall configured by myself with confidence.
Network Countermeasures: 5
The most recent version Snort IDS system is used to listen for connection attempts to this port, and a correctly configured firewall itself is the only access point to and from the internal network.

The calculated severity is:
(Critical + Lethal) – (System + Net Countermeasures) = (3+2) – (4+5) = -4

2.4.9. Defensive recommendation:

- Apply timely the appropriate patches to the system where the web server resides.
- Implemente a security measure by making FormMail to check the HTTP_REFERRER field. In this way, you can force the script to accept requests of sending mail only from certain domains that can be specified.
- Secure the FormMail by hard-coding the recipient's email address in the script. This is a best way to stop spamming by preventing the FormMail from taking the recipient's address from a HTTP variable.
- Ensure the address check against a specified list of authorized recipients when one has to use the HTTP variable to get the recipient's address.
- It is strongly recommended that FormMail never be intentionally employed in conjunction with any kind of e-mail auto-responder. If the case, it may leave a room for an attacker to repeatedly invoke FormMail while setting the emailCGI parameter to the e-mail address of his/her intended victim.
- Consider alternative solution, cgiemail from MIT is recommended:
  http://web.mit.edu/wwwdev/cgiemail/
  It takes all it's information from a plain text file template, so spoofing from fields should not come in to play.

2.4.10. Multiple choice test question:

What is the well-known vulnerability in FormMail v.1.9?

a) Anonymous Mail Forwarding Vulnerability
a) Sendmail vulnerability
b) Vulnerability in metamail
c) CGI buffer overflow vulnerability.

Answer: a) is the correct answer

References:

[1] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids181&view=research
[2] Sven Dietrich NASA Goddard Space Flight Center "An analysis of the ``Shaft'' distributed denial of service tool" at http://security.royans.net/info/posts/bugtraq_ddos3.shtml
[3] An addition to [2] at http://biocserver.cwru.edu/~jose/shaft_analysis/node-analysis.txt
[4] http://www.cert.org/reports/dsit_workshop.pdf
[5] http://www.incidents.org/diary/july2001.php
[6] http://email.about.com/library/weekly/aa052801a.htm?once=true&
[7] http://www.securityfocus.com (search: formmail)
[8] http://worldwidemart.com/scripts/formmail.shtml
[9] http://www.securityfocus.com/bid/2469
[10] Dittrich, David. The DoS Project's "trinoo" distributed denial of service attack tool http://staff.washington.edu/dittrich/misc/trinoo.analysis
[11] Classification of CERT/CC Advisories 1993--1998

http://seclab.cs.sunysb.edu/sekar/papers/cert.htm
[12] http://www.whitehats.com/IDS/181
[13] http://www.ngsec.com/docs/whitepapers/polymorphic_shellcodes_vs_app_IDSs.PDF
[14] http://www.tlsecurity.net/Textware/Security/Buffer.Heap.Overflows/understanding-bof.html
[15] http://www.tlsecurity.net/Textware/Security/Buffer.Heap.Overflows/stealthcode.txt

Assignment 3 - "Analyze This" Scenario

Table of Contents

3.1. Executive Summary

This is a security audit report of a University's 5 days log files. The log files are generated from the University's Snort implementation. Specifically, the data to be analyzed was spanned from August 1 – 5, 2002. The log output during this time was divided into three different sets of files: scans, alerts, and Out Of Specifications (OOS).

This audit report aims at providing security assessment, analysis, and recommendations over the University's B-class network, MY.NET.x.x. The report is organized as follows: after showing the origin of the data to be analyzed in section 3.2, we identify in section 3.3 the University's network and it components. In the next three sections a summary and brief description of scans, alerts, and OOS logs are presented respectively. Some interesting scanning phenomena, critical alerts, and OOS are analyzed in depth in these sections. In section 3.7, some graphs are drawn for showing the relationship between scans and alerts files, and an intuitive and clear traffic flow with all critical alerts analyzed.

After the above analysis, some security recommendations are presented in section 3.8. And finally in order to facilitate reader's understanding, a description of analysis process used and references is given in the last two sections respectively.

3.2. Data files analyzed

The data files that I have chosen to analyze are as follows:

Alert files:

alert.020801.gz.txt	10,175KB
alert.020802.gz.txt	12,324KB
alert.020803.gz.txt	15,596KB

alert.020804.gz.txt	73,824KB
alert.020805.gz.txt	85,069KB

In the following text, we refer 'alerts log file' or 'alerts file' to a concatenated file from the above five files. The size of the file is 239,641KB.

Note that it is not simply a sum of the sizes of the above five size because I have to get rid of port-scan entries in the raw data files (see section 3.9).

Scan files:

scans.020801.gz.txt	11,239KB
scans.020802.gz.txt	38,972KB
scans.020803.gz.txt	61,823KB
scans.020804.gz.txt	104,807KB
scans.020805.gz.txt	45,442KB

we refer 'scans log file' or 'scans file' to a concatenated file from the above five files. The file size is 262,225KB.

OSS files:

oos_Aug.1.2002.gz.txt	2KB
oos_Aug.2.2002.gz.txt	309KB
oos_Aug.3.2002.gz.txt	139KB
oos_Aug.4.2002.gz.txt	1KB
oos_Aug.5.2002.gz.txt	1KB

A 'oos log file' or 'oos file' is referred to a concatenated file from the above five files.

At beginning of doing this assignment, I wanted to use Snortsnartf as a main analysis tool to deal with these raw data. However, the size of all five days' log files together becomes too big to be dealt with by using Snortsnarf. I had to first convert the log files into CSV files, and then input them into MS SQL server (see section 3.9 for more detail). As such, the following analysis is mostly executed via SQL queries over the related database tables.

3.3. Network and critical hosts profile

In order to make an effective analysis, it is a good idea to start it by building a profile for the University's network and its components or public servers. Since there is no information given about the network topology, we need to come up a way to figure out it according to the available information. So the only resource we can use is from the log files shown in section3.2.

As every one knows, 20K hosts are a realistic number of hosts for a B-class campus network. So the exploitation of the network topology only based on scan and alert log files

could be challenging. Furthermore, it is also true in the light of the fact that source addresses could be spoofed, and many attacks can attempt to non-existent hosts. Bearing all these facts in mind, the network profile building is processed in the following way:

- By using the scans log file, one can gain some hints about the fact that some *possible* known services may reside on some hosts targeted by the scanners if one can focus on possible traffic flow in both directions.
- By using the alerts log file, some real hosts on which some popular services reside can be identified. First the alerts needed to be identified as coming from a stimulus packet or a response packet thus identifying if the server was located on the source or destination. For example, the DNS Server is identified by looking for a large number of traffic to a destination port 53. Some traffic with source port 53 could indicate some response packets back to the DNS server request. Note that the DNS zone transfer is a stimulus from a client. Another example is that most snort WEB-* alerts could be used to identify web servers. This is because the snort rules require A+ flags indicating the session has been established. In our case, there are numerous web servers identified.
- From the alerts file, you can find 106,883 UDP SRC and DST outside network alerts. If you dig further you will find that this traffic does not appear to have originated from the campus network and destined for any IP address on the same network. That is, none of the source or destination IP addresses belong to the campus network. A plausible explanation of this is that the University may have multiple Internet access points provided by other ISP via cable modem, SDL, or T1 and T3. This leads us to find some Internet access points shown in Table-3.2.

It must note that there are three assumptions about the validation of the network profile identification. Firstly, it is assumed that there could have been some previous malicious or no malicious activities priori to the activities shown in our collected five days' log. Secondly, some detects could be ignored by this analysis since hosts could be compromised at an earlier or later date. The last assumption is that some detects could be ignored because there exists the fact that some log raw data entries are corrupted (i.e. the raw data entries are logged in an unusual format).

As a result of this exercise a host table along with the services provided is established in table-3.1.

Host	Service	Host	Service
MY.NET.137.7	DNS	MY.NET.70.198	1080
		MY.NET.163.78	1080
MY.NET.70.50	FTP	MY.NET.6.59	POP3
MY.NET.70.69	FTP, Telnet		
MY.NET.70.49	FTP	MY.NET.28.10	Printer
MY.NET.182.98	FTP	MY.NET.28.8	Printer
		MY.NET.154.12	Printer
MY.NET.111.140	HTTP	MY.NET.154.9	Printer

MY.NET.5.96	HTTP	MY.NET.28.5	Printer
MY.NET.70.58	HTTP		
MY.NET.105.10	HTTP	MY.NET.157.108	TFTP
My.NET.91.8	HTTP	MY.NET.180.39	TFTP
		MY.NET.178.76	TFTP
MY.NET.100.217	SMTP	MY.NET.117.25	TFTP
MY.NET.6.40	SMTP	MY.NET.84.189	TFTP
MY.NET.145.9	SMTP	MY.NET.84.223	SQL Server
MY.NET.139.230	SMTP	MY.NET.53.143	SQL Server
		MY.NET.1.205	SQL Server
MY.NET.159.104	DHCP server	MY.NET.106.228	SQL Server
MY.NET.163.97	SSH Server		
MY.NET.104.204	KaZaA	MY.NET.70.198	IRC(6667)
MY.NET.108.42	KaZaA	MY.NET.82.130	IRC
MY.NET.117.137	KaZaA	MY.NeT.178.199	IRC
My.NET.70.210	KaZaA	MY.NET.82.87	IRC
MY.NET.198.204	KaZaA	MY.NET.163.93	IRC
MY.NET.28.10	RPC (111)		
MY.NET.28.3	RPC	MY.NET.38.8	RPC
MY.NET.38.5	RPC	MY.NET.38.6	RPC

Table-3.1 Main Hosts and Public Servers on MY.NET

Other Internet access points:

Access gateway	Count	Associated port	
233.28.65.148	32115	5779	
233.28.65.173	4975	5779	
233.40.70.50	172	5779	
233.2.171.1	179545	56464	

Table-3.2 Other internet access points

From the available data, I found that public services such as DNS, HTTP, FTP, Socks, and SMTP appear to exist on the B campus network. There are many HTTP servers (around 500), only top 5 are listed in Table-3.1. There are also a lot of printers (around 600) on the campus network. Only the top 5 are shown in Table-3.1.

The hosts shown in Table-3.1 are of high value to potential attackers. So it is recommended that the University system administrators take close attention to the listed servers.

During the course of determining the network profile, I have made some observations:

- The network MY.NET seems quite open in the sense that we can find many kinds of

traffic circulating on the network.

- The hosts of KaZaA server generated a high volume of traffic. It also resulted in a large number of ICMP error messages as clients to these servers dropped off the Internet or had the return traffic filtered.
- Almost all NIMDA attacks are launched from one host MY.NET.100.208. This resulted a huge amount of risks to the outside of the campus network. It would be advised that the University security officer follow upon this case and take some measure against this.
- Only one gateway MY.NET.88.162 is identified because it accepts broadcast messages in the scans file and returns ICMP error messages in alert file.
- Surprisingly enough, few DNS and Telnet servers can be identified according to the alert file. By checking out the scans file, there is, however, no lack of exploitations on both ports 53 and 21. This is probably a good sign that the University has taken some security measures to protect these servers. One effective protective way is to put these most vulnerable servers behind the firewalls that prevent the attackers' initial reconnaissance step from success at least.

3.4. Portscan file analysis

A common way for an attacker to make attacks to such a big network is to make a thorough scan first. Through the scanning results, the attacker can gain a quite good knowledge about the network he/she is about to attack. This is often called network reconnaissance attacks. In this section, we first analyze the scans log file to see what are ports attackers are mostly interested in, what types of scans they are deployed. The results are shown in forms of statistic table for the sack of readability and clarity.

A total of 4,109,293 scans are reported in the five-day's scans file. The top ten scanning destination ports and scan types are shown in Table-3.3 and Table-3.4.

Scan type	Count
UDP	3111549
SYN	997435
INVALIDACK	60
NULL	59
NOACK	52
SYNFIN	5
XMAS	4
FULLXMAS	4
FIN	4
NMAPID	1

Destination Port	Count
41170	2441874
80	815894
6257	204251
1433	72379

21	35331
28800	29492
53	17388
27005	16382
139	16185
7003	14915

Table-3.3 top 10 Scan Destination Ports

Table-3.4 top 10 scan types

Some observations are made from the above two tables:

From table-3.3-3.4, we see that two types of scans (UDP and TCP SYN) are overwhelmingly present. Ports 41170 (mp3 Blubster music), 80 (http), 6257 (P2P), 1433 (SQL), and 21 (Telnet) are mostly used as scanning ports.

The large amount of scan attempts towards ports 41170 and 6257 may indicate that some hosts in the campus network are assigned as P2P clients. From table-3.3, we also see that there is a large amount of scans towards ports 80. This phenomenon conforms to the current attack trend that port 80 is mostly used port for attackers (see http://www.dshield.org/topports.html).

From the scans file, 47 internal hosts are found to initiate traffic to port 1214 of other hosts. It indicates that many internal hosts were installed with the KaZaA Media Desktop software. Special attention should be paid to these hosts as this kind of peer-to-peer file sharing software can be a backdoor to your internal network and result in information theft and leakage. Files infected with virus can also be transmitted into the internal hosts by this mean. You are highly suggested using a stateful firewall to block incoming to port 1214. Besides, you should consider performing a thorough virus scan on the internal hosts, and removing the software installed at the hosts. It is even better and encouraged to pursue a regular virus scan on the internal hosts.

Portscans' Top 10 Scan Talkers

Internal Src	Count		External Src	Count
my.net.70.200	2438671		216.228.171.81	25940
my.net.84.234	478411		24.138.61.171	21019
my.net.100.208	170345		161.132.205.100	20330
my.net.70.207	137226		211.232.192.153	17730
my.net.82.2	127792		67.104.84.142	16264
my.net.165.24	104491		219.96.171.20	15741
my.net.83.150	90049		80.137.90.34	15693
my.net.137.7	49208		24.101.152.5	12593
my.net.70.133	42744		202.98.223.86	10739
my.net.81.27	31926		66.224.37.26	10139

Table 3.5 Top 10 Scan Talkers

It is always interesting to know who has mostly initiated the scans and who they are with which possible intention. So the following list shows the ARIN search or nslookup results relating top ten talkers with external IP addresses.

Please note that most traffic is from the internal network!! We will address this issue in section 3.8. Here we only look for external ones.

ARIN search for top external talkers: Server used for this query: [whois.arin.net]

External Src	
216.228.171.81	Bend Cable (NETBLK-BENDCABLE) Box 5067 BEND, OR 97701 US Netname: BENDCABLE Netblock: 216.228.160.0 - 216.228.191.255 Maintainer: BCCI Coordinator: cotton, byron (BC17-ARIN) bcotton@bendcable.com 541-382-5551
24.138.61.171	Access Cable Television (NETBLK-ACCESS-BLK1) 190 Victoria Rd Dartmouth, Nova Scotia B2Y 4A4 CA Netname: ACCESS-BLK1 Netblock: 24.138.0.0 - 24.138.79.255 Maintainer: ACCA Coordinator: Potvin, Jeff (JP1495-ARIN) jpotvin@accesscable.com (902) 469-9540 (FAX) (902) 466-6482 Domain System inverse mapping provided by: EUROPA.ACCESSCABLE.NET 24.138.0.5 PEGGY.ACCESSCABLE.NET 24.138.0.7
161.132.205.100	Red Cientifica Peruana (NET-RCP) Augusto Tamayo 125 San Isidro, Lima 27 PE Netname: RCP Netblock: 161.132.0.0 - 161.132.255.255 Maintainer: RCP Coordinator: RCP, Operador (ET45-ARIN) operador@rcp.net.pe +51-1-2415689 (FAX) +51-1-2411320

211.232.192.153	Asia Pacific Network Information Center (NETBLK-APNIC-CIDR-BLK) APNIC AU Netname: APNIC-CIDR-BLK2 Netblock: 210.0.0.0 - 211.255.255.255 Coordinator: Administrator, System (SA90-ARIN) [No mailbox] +61 7 3858 3100 Domain System inverse mapping provided by: NS1.APNIC.NET 202.12.29.25 NS3.APNIC.NET 202.12.28.131 NS.RIPE.NET 193.0.0.193 RS2.ARIN.NET 192.149.252.22 NS.TELSTRA.NET 203.50.0.137
67.104.84.142	XO Communications (NET-XOXO-BLK-17) 1400 Parkmoor Avenue San Jose, CA 95126-3429 US Netname: XOXO-BLK-17 Netblock: 67.104.0.0 - 67.105.255.255 Maintainer: XOXO Coordinator: DNS and IP ADMIN (DIA-ORG-ARIN) hostmaster@concentric.net (408) 817-2800 Fax- - - (408) 817-2630
219.96.171.20	Asia Pacific Network Information Center (NET-APNIC5) APNIC AU Netname: APNIC5 Netblock: 219.0.0.0 - 219.255.255.255 Maintainer: AP Coordinator: Administrator, System (SA90-ARIN) [No mailbox] +61 7 3858 3100

80.137.90.34	European Regional Internet Registry/RIPE NCC (NET-80-RIPE) These addresses have been further assigned to European users. Contact information can be found in the RIPE database at whois.ripe.net NL Netname: 80-RIPE Netblock: 80.0.0.0 - 80.255.255.255 Maintainer: RIPE Coordinator: Reseaux IP European Network Co-ordination Centre Singel 258 (RIPE-NCC-ARIN) nicdb@ripe.net +31 20 535 4444
24.101.152.5	Rogers Cable Inc. (NET-ROGERS-CAB-6) 1 Mount Pleasant Road Toronto, Ontario M4Y 2Y5 CA Netname: ROGERS-CAB-6 Netblock: 24.100.0.0 - 24.103.255.255 Maintainer: ROCB Coordinator: Budd, Paul (AD30-ARIN) abuse@rogers.com 416) 935-4729
202.98.223.86	Asia Pacific Network Information Center (APNIC2) APNIC AU Netname: APNIC-CIDR-BLK Netblock: 202.0.0.0 - 203.255.255.255 Maintainer: AP Coordinator: Administrator, System (SA90-ARIN) [No mailbox]+61 7 3858 3100
66.224.37.26	Advanced Telcom Group (NETBLK-ATG-IP-BLK5) 120 Stony Point Road Santa Rosa, CA 95401 US Netname: ATG-IP-BLK5 Netblock: 66.224.0.0 - 66.224.255.255 Maintainer: ATG Coordinator: Advanced Telcom Group (ZA22-ARIN) noc@atgi.net 707.535.8900

Table-3.6 shows the top targeted hosts by the scanners

Internal Dest	Count		External Dest.	Count
MY.NET.117.25	6220		216.254.108.19	27885
MY.NET.85.91	999		204.183.84.240	12633

MY.NET.114.4	842		204.183.84.225	10562
MY.NET.53.51	784		67.68.113.139	9179
MY.NET.178.181	761		62.229.74.253	8703
MY.NET.100.217	648		216.254.108.22	8663
MY.NET.15.222	641		140.192.175.183	8301
MY.NET.146.15	621		66.130.178.166	7476
MY.NET.150.71	601		210.187.110.110	6810
MY.NET.151.72	597		12.245.28.142	6602

Table-3.6 Top 10 Scan Destinations

3.5. Alerts file analysis

This section deals with the alerts from the alerts file. I first display all kinds of alerts logged by the Snort system. In this section, I do not follow the traditional way to make analysis on the alerts only based on their high occurrence number. I first rate these alerts according to my knowledge of alerts and in terms of their criticality. So the analysis is carried out on 5 top critical ones. A brief description for each alerts listed in Table-3.7 is shown in Appendix A.

The alerts are divided into the following rating classes in order of decreasing criticality:

R1. *Critical alerts* give a sign that there exists a high probability that the internal system could be compromised. Attacks like Denial of Service (DOS) can knock down the system, while attacks like Trojan Horse can have a high success rat in exploiting an internal system.
R2. *Dangerous alerts* refer to some motivated attacks such as port scanning, buffer overflow, and system vulnerability exploit.
R3. *Low risk alerts* refer to the attacks like OS fingerprint, network status monitoring, and application vulnerability exploit.

Note that I have also taken the alerts' occurrence number into account when rating for a particular alert to be addressed in the following text. But it is not the major fact for my choices hereafter.

#	Rate	Alert	Alert count	SrcIP count	DesIP count
1	R1	NIMDA - Attempt to execute cmd from campus host	877538	877538	874497
2	R2	IDS552/web-iis_IIS ISAPI Overflow ida INTERNAL nosize	482402	482402	481322
3	R2	spp_http_decode: IIS Unicode attack detected	342978	342978	341516
4	R1	NIMDA - Attempt to execute root from campus host	123305	123305	122875
5	R3	UDP SRC and DST outside network	106883	106883	106849
6	R3	SMB Name Wildcard	30083	30083	30074

7	R1	TFTP - External UDP connection to internal tftp server	24220	24220	24214
8	R2	External RPC call	14578	14578	14577
9	R3	Watchlist 000220 IL-ISDNNET-990517	11921	11921	11917
10	R1	Possible Trojan server activity	4113	4113	4113
11	R2	spp_http_decode: CGI Null Byte attack detected	2578	2578	2577
12	R2	SUNRPC highport access!	2543	2543	2543
13	R2	IRC evil - running XDCC	2054	2054	2053
14	R2	Watchlist 000222 NET-NCFC	1305	1305	1305
15	R1	EXPLOIT x86 NOOP	1293	1293	1293
16	R3	Queso fingerprint	1120	1120	1120
17	R3	SNMP public access	927	927	927
18	R3	connect to 515 from outside	788	788	788
19	R2	Attempted Sun RPC high port access	730	730	730
20	R3	Samba client access	679	679	679
21	R2	High port 65535 udp - possible Red Worm – traffic	628	628	628
22	R2	IDS552/web-iis_IIS ISAPI Overflow ida nosize	314	314	314
23	R3	ICMP SRC and DST outside network	260	260	260
24	R3	SMB C access	236	236	236
25	R3	TFTP - Internal UDP connection to external tftp server	173	173	173
26	R3	Beetle.ucs	166	166	166
27	R2	Port 55850 tcp - Possible myserver activity - ref. 010313-1	147	147	147
28	R2	Incomplete Packet Fragments Discarded	136	136	136
29	R2	NMAP TCP ping!	88	88	88
30	R2	Null scan!	76	76	76
31	R1	EXPLOIT x86 setuid 0	58	58	58
32	R2	Tiny Fragments - Possible Hostile Activity	53	53	53
33	R2	EXPLOIT x86 stealth noop	48	48	48
34	R2	High port 65535 tcp - possible Red Worm – traffic	44	44	44
35	R3	STATDX UDP attack	42	42	42
36	R1	EXPLOIT x86 setgid 0	38	38	38
37	R2	Port 55850 udp - Possible myserver activity - ref. 010313-1	18	18	18
38	R3	TCP SRC and DST outside network	13	13	13
39	R3	SMB CD...	13	13	13
40	R3	External FTP to HelpDesk my.net.70.50	11	11	11
41	R2	EXPLOIT NTPDX buffer overflow	5	5	5
42	R3	RFB - Possible WinVNC - 010708-1	3	3	3
43	R1	Back Orifice	3	3	3
44	R2	DDOS shaft client to handler	3	3	3
45	R3	Traffic from port 53 to port 123	2	2	2
46	R3	SYN-FIN scan!	2	2	2

Table-3.7 Snort Alerts sorted by number of alerts for August1 –5, 2002

In the following subsections, five critical alerts have been chosen for analyzing: NIMDA - Attempt to execute cmd from campus host; TFTP - External UDP connection to internal

tftp server; EXPLOIT x86 NOOP; Possible trojan server activity; and Back Orifice.

3.5.1 Alert Analysis-1: NIMDA - Attempt to execute cmd from campus host

Related alerts in Table-3.7: #1, #2, #3, and #4

Src IP	Count	Dest IP	Count
MY.NET.100.208	997384	130.116.101.102	35
MY.NET.70.144	1	130.141.140.119	33
MY.NET.70.16	1	130.252.139.159	32
MY.NET.70.169	1	130.217.125.135	32
MY.NET.105.10	1	130.30.203.75	32

Table-3.8 Top 5 talkers and targets in Alert Analysis-1

From Table-3.8, we can see that this attack is almost exclusively launched by one internal attacker on the MY.NET network. Although there is no outside attackers launching this attack against the campus network, this internal attacker could make some damages to the Internet community. This can ruin the University's reputation.

Note that the statistic numbers shown in Table-3.8 include only those for alert #1.

a) Trace

08/04-17:44:33.001172 [**] NIMDA - Attempt to execute cmd from campus host [**] MY.NET.100.208:2142 -> 65.54.250.121:80
08/04-13:46:45.559476 [**] NIMDA - Attempt to execute cmd from campus host [**] MY.NET.70.144:4901 -> 207.46.235.162:80
08/04-19:55:53.607645 [**] NIMDA - Attempt to execute cmd from campus host [**] MY.NET.70.16:1345 -> 207.68.132.9:80

b) Description of the Nimda Attack

There are four ways Nimda can spread widely:

1) From an affected host that sends e-mail with random text in the subject line, no body text, and an attached file called readme.exe. If a Windows user opens the attached file, the worm will use Mailing API (MAPI) functions to read the user's e-mail address book and send out copies of itself to all of the addresses.

2) Nimda spreads via Internet scan. From an infected IIS Web server, Nimda scans other Web servers by looking for other systems vulnerable to the Unicode Web Traversal. Once Nimda gains access to a Web server, it may display a Web page and prompt users to download an infected file.

3) Nimda worm moves via Web page. JavaScript on an infected Web page will force a

download of the file readme.eml. Some versions of Internet Explorer vulnerable to the "Automatic Execution of Embedded MIME Types" problem will automatically execute the file and infect the client.

4) Nimda will spread itself via open share on a network by exploiting a known directory traversal vulnerability. In such, the worm can copy itself to all directories in which the infected user has permission.

c) Correlations

- IIS Web server log looks like the following as signature:
GET /scripts/root.exe?/c+dir
GET/MSADC/root.exe?/c+dir
GET /c/winnt/system32/cmd.exe?/c+dir
GET/d/winnt/system32/cmd.exe?/c+dir
GET/scripts/..%5c../winnt/system32/cmd.exe?/c+dir
GET/_vti_bin/..%5c../..%5c../..%5c../winnt/system32/cmd.exe?/c+dir

- The Nimda worm is Operating System specific. It affect mostly Microsoft IIS servers and in some printer, routers/switches. So knowing the IIS vulnerabilities is helpful for understanding the NIMDA itself:

http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0333
http://www.cert.org/advisories/CA-2001-12.html
http://www.cert.org/advisories/CA-2001-11.html
http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS01-033.asp

- The protocols to which the NIMDA worm can exploit

TCP/IP (Transmission Control Protocol/Internet Protocol)
UDP (User Datagram Protocol)
HTTP (HyperText Transfer Protocol)
SMTP (Simple Mail Transfer Protocol)
MAPI (Message Application Program Interface)
TFTP (Trivial File Transfer Protocol)
NetBIOS (Network Basic Input/Output System)
MIME (Multipurpose Internet Mail Extensions)
IIS (Microsoft Internet Information Server)
IE (Microsoft Internet Explorer) Browser

d) Defensive Recommendation

- Revise the University's security policy so that the kind of attacks from the internal can be eliminated. Also, a policy regarding how to deal with the infected machines should be in place.
- Technically, an egress filter rule on the firewall for blocking this traffic is appropriate.
- Have the latest signature files from a trusted Anti-virus software vendors and let it

run on your workstations and clients.

- Block this type of traffic at the router using the guide from Cisco: http://www.cisco.com/warp/public/63/nimda.shtml.
- Keep all systems up-to-date with security patches. Look at the Microsoft web site: http://www.microsoft.com/technet/security/bulletin.
- Educate the users not to open the email attachment with a name of Readme.exe.

3.5.2 Alert Analysis-2: TFTP - External UDP connection to internal tftp server

Related alerts in Table-3.7: #7, #25, and #40.

Src IP	Count	Dest IP	Count	Dest Port	Count
MY.NET.111.230	6090	192.168.0.216	24207	1806	48
MY.NET.111.231	6059	NULL	6	1800	48
MY.NET.109.105	6053	MY.NET.117.25	2	1062	48
MY.NET.111.219	6007	MY.NET.180.39	2	1298	47
209.61.187.112	2	MY.NET.114.44	1	1944	45

Table-3.9 Top 5 talkers, targets and destination ports in Alert Analysis-2

a) Trace

```
08/02-13:38:20.944624  [**] TFTP - External UDP connection to internal tftp server [**]
MY.NET.111.219:69 -> 192.168.0.216:7473
08/02-13:38:20.945295  [**] TFTP - External UDP connection to internal tftp server [**]
MY.NET.111.230:69 -> 192.168.0.216:7473
08/02-13:38:20.947486  [**] TFTP - External UDP connection to internal tftp server [**]
MY.NET.109.105:69 -> 192.168.0.216:7473
08/02-13:38:20.947543  [**] TFTP - External UDP connection to internal tftp server [**]
MY.NET.111.231:69 -> 192.168.0.216:7473
```

b) Description of Alert:

The TFTP (trivial file transfer protocol) service provides remote access to files without asking for a password. From the above excerpts of logs for this alert, we can find that some internal hosts on the MY.NET response to the outside hosts with source tftp port 69. This could strongly indicate that these internal hosts could be compromised due to the fact that the external hosts were connecting to them. It might be necessary to see who are they and what are their real intention of the tftp connection. In our case, only one outside IP 192.168.0.216 is associated with attack (see the ARIN search result for this IP).

c) Correlations

- ARIN search and nslookup results for 192.168.0.216.

```
OrgName:    IANA
OrgID:      IANA-2
```

```
NetRange:    192.168.0.0 - 192.168.255.255
CIDR:        192.168.0.0/16
NetName:     IANA-CBLK1
NetHandle:   NET-192-168-0-0-1
Parent:      NET-192-0-0-0-0
NetType:     Direct Assignment
NameServer:  BLACKHOLE-1.IANA.ORG
NameServer:  BLACKHOLE-2.IANA.ORG
Comment:     This block is reserved for special purposes.
             Please see RFC 1918 for additional information.

RegDate:     1994-03-15
Updated:     2001-10-12

TechHandle:  IANA-ARIN
TechName:    Internet Corporation for Assigned Names and
Number
TechPhone:   +1-310-823-9358
TechEmail:   res-ip@iana.org
```

- TFTP server is said one of the most vulnerable servers, because it can be accessed without authentication. see http://www.cert.org/advisories/CA-1991-18.html
http://www.kb.cert.org/vuls/id/211736
- It is also possible to let Remote Users Cause IOS-based Devices to Crash through Cisco ISO buffer overflow in Processing TFTP File Names. See:
http://www.securitytracker.com/alerts/2002/Jul/1004858.html

d) Defensive Recommendations:

- Establish a tftp usage rule in the University's security policy. In view of the tftp's vulnerabilities (see http://www.cert.org/advisories/CA-1991-18.html and http://www.kb.cert.org/vuls/id/211736), it is better to block this type of traffic unless it is really necessary.
- Using stateful firewall or VPN to make the traffic authenticated and encrypted.
- Harden the tftp servers listed in Table-3.1 and update the appropriate patches on these boxes.

3.5.3 Alert Analysis-3 SHELLCODE x86 NOOP

Related alerts in Table-3.7: #15, #31, #33, and #36.

a) Description of Attack

A similar attack is analyzed in Assignment-2. Although this attack and the one in section 2 are detected by two different Snort signatures (due to the fact that they use different Noop sleds), their attack mechanism and ultimate attack purpose are the same. For this reason, I do not spend more time to describe this attack. The reader is urged to see section 2.2.

Src IP	Count	Dest IP	Count	Dest Port	Count
129.123.19.136	621	MY.NET.100.214	300	139	930
128.95.160.221	390	MY.NET.116.53	193	445	228
64.81.195.164	147	MY.NET.70.28	179	2011	60
66.57.117.222	60	MY.NET.82.130	147	2438	47
140.172.180.165	47	MY.NET.115.86	116	1214	16

Table-3.10 Top 5 talkers, targets, and ports in Alert Analysis-3

b) Trace

```
[**] SHELLCODE x86 NOOP [**]
06/03-18:50:52.854488 206.105.2.76:80 -> XXX.YYY.106.176:64943
TCP TTL:53 TOS:0x0 ID:46866 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0xAC8C4EEC  Ack: 0x28984E38  Win: 0x1920  TcpLen: 20
0x0000: 00 00 0C 04 B2 33 00 03 E3 D9 26 C0 08 00 45 00  .....3....&...E.
0x0010: 05 DC B7 12 40 00 35 06 DB 5A CE 69 02 4C E2 B9  ....@.5..Z.i.L..
0x0020: 6A B0 00 50 FD AF AC 8C 4E EC 28 98 4E 38 50 10  j..P....N.(.N8P.
0x0030: 19 20 19 D0 00 00 90 90 90 90 90 90 90 90 90 90  . ..............
0x0040: 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
```

c) Nslookup result for 129.123.19.136:

```
> nslookup 129.123.19.136
Name:    khuang.bus.usu.edu
Address:  129.123.19.136
```

Nslookup result for 128.95.160.221

```
> nslookup 128.95.160.221
Name:    harlem.ebiz.washington.edu
Address:  128.95.160.221
```

By using the nslookup tool or the whois services provided by http://ws.arin.net/, we can see that the top two IP addresses belong to University of Washington and Utah State University, U.S.A.

d) Evidence of target

Since we have not got the raw data with payload, whether these attacks are false positive or not are undetermined.

e) Defensive Recommendation
See section 2.2.

3.5.4 Alert Analysis-4: Possible trojan server activity

Top 5 sources triggering this attack signature and top 5 destination hosts in MY.NET receiving

this attack are shown in Table-3.11.

Src IP	Count	Dest IP (MY.NET)	Count
63.196.247.234	624	MY.NET.84.155	16
218.154.202.148	455	MY.NET.84.95	15
217.136.63.141	322	MY.NET.85.184	15
80.62.155.240	321	MY.NET.85.189	15
61.102.149.115	311	MY.NET.85.38	14

Table-3.11 Top 5 talkers and targets in Alert-Analysis-4

a) Brief description of the attack

The alert is triggered when a packet arrives with a source or destination port of 27374. According to the following port list, http://www.SANS.ORG/newlook/resources/IDFAQ/oddports.htm. This port is associated with the trojans Bad Blood, SubSeven, SubSeven 2.1 Gold, Subseven 2.1.4, and DefCon 8. Most likely these alerts are related to a version of SubSeven, as this is currently one of the more popular trojans in use within the attacker community.

SubSeven is said an improved version of NetBus which was the first 'point and click' trojan that made it very easy for hackers to abuse an infected system (see [12]). SubSeven is a remote control trojan for Windows machines. It allows the attacker to control the victim machine. The attacker uses their SubSeven client to remotely connect to the victim machine infected with a SubSeven server, typically on TCP port 27374 for version 2.1. SubSeven even has the option to notify the attacker when a victim machine infected with the server is online, and provides the attacker with the IP address and port on which the server is listening. More information on SubSeven is located at: http://www.sans.org/newlook/resources/IDFAQ/subseven.htm.

b) Trace

Let us extract the alert log for the top source and destination one.

```
08/05-18:52:18.303129  [**] Possible trojan server activity [**] 63.196.247.234:1932 ->
MY.NET.84.184:27374
08/05-18:52:18.305672  [**] Possible trojan server activity [**] MY.NET.84.184:27374 ->
63.196.247.234:1932
08/05-18:52:18.352004  [**] Possible trojan server activity [**] 63.196.247.234:1971 ->
MY.NET.84.184:27374
08/05-18:52:18.352370  [**] Possible trojan server activity [**] MY.NET.84.184:27374 ->
63.196.247.234:1971
08/05-18:52:18.971927  [**] Possible trojan server activity [**] 63.196.247.234:1971 ->
MY.NET.84.184:27374
08/05-18:52:18.972275  [**] Possible trojan server activity [**] MY.NET.84.184:27374 ->
63.196.247.234:1971
08/05-18:52:18.991637  [**] Possible trojan server activity [**] 63.196.247.234:1932 ->
MY.NET.84.184:27374
```

08/05-18:52:18.991957 [**] Possible trojan server activity [**] MY.NET.84.184:27374 -> 63.196.247.234:1932
08/05-18:52:19.530042 [**] Possible trojan server activity [**] 63.196.247.234:1932 -> MY.NET.84.184:27374
08/05-18:52:19.530267 [**] Possible trojan server activity [**] MY.NET.84.184:27374 -> 63.196.247.234:1932
08/05-18:52:19.536708 [**] Possible trojan server activity [**] 63.196.247.234:1971 -> MY.NET.84.184:27374
08/05-18:52:19.536996 [**] Possible trojan server activity [**] MY.NET.84.184:27374 -> 63.196.247.234:1971

From the above extract, we notice that the host at MY.NET.84.184 was probed on the default port 27374 for the SubSeven trojan by the host 63.196.247.234. The host MY.NET.84.184 did response back. This could be a strong indication that this host is compromised.

Considering all the internal hosts that Snort detected as they did respond to outside hosts over port 27374, I found there are in total 318 internal hosts doing so. The top 10 ones are listed in Table-3.12.

Subseven client	Count
MY.NET.84.155	16
MY.NET.83.95	15
MY.NET.84.184	15
MY.NET.84.189	15
MY.NET.83.38	14
MY.NET.84.226	14
MY.NET.85.92	14
MY.NET.84.166	14
MY.NET.83.6	14
MY.NET.84.229	14

Table-3.12 Top 10 Internal hosts as Subseven clients

I also made a search on which internal hosts had served as a Subseven server. If a host is a Subseven server, it must have initiated the connection through destination port 27374, and some outside hosts responded back the connection to that internal host with source port 27374. The database search result shows there are three of them (see Table-3.13).

Internal host	Count
MY.NET.60.10	10
MY.NET.70.113	4
MY.NET.11.4	2

Table-3.13 Subseven server on the campus network

c) Defensive recommendation

- According to University's policy, set up a firewall rule to block port 27374 to or from the Internet.
- It should fully investigate this activity and determine whether it is authorized. Search for signs of compromise on the hosts listed in Table-3.12 and Table-3.13. Use the following steps recommended by SANS in the SubSeven IDFAQ URL: http://www.sans.org/newlook/resources/IDFAQ/subseven.htm to remove the trojan.

For more detail, please also visit the site:
http://www.nipc.gov/warnings/advisories/2001/01-014.htm and
http://www.hackfix.org/subseven/.

3.5.5 Alert Analysis-5: Back Orifice

a) Brief description:

BO2K is a widely distributed 'remote control' windows utility. It can be used by both 'good-guy' and 'bad-guy'. Here we rather consider this is a tool for 'bad-guys' since there is no any security mechanism is inherently implemented. It is usually distributed by malicious people in the form of Trojan Horse (see http://www.irchelp.org/irchelp/security/trojan.html for a comprehensive description of Trojan Hose attack). The web page http://www.nwinternet.com/~pchelp/bo/bo.html gives out a good description of the Back Orifice itself.

Using Back Orifice as a Trojan Horse received a lot of attention in public in early 1998 (see Cert advisory at http://www.cert.org/vul_notes/VN-98.07.backorifice.html).

SRC Ip	Count	Dest IP	Count
63.240.142.227	2	MY.NET.117.25	2
212.143.222.236	1	MY.NET.70.236	1

Table-3.14 Top talkers and targets in Alert Analysis-5

b) Trace:

```
08/05-16:36:39.582295  [**] Back Orifice [**] 63.240.142.227:18672 -> MY.NET.117.25:31337
08/05-16:36:39.707788  [**] Back Orifice [**] 63.240.142.227:18672 -> MY.NET.117.25:31337
```

In the above trace, it seems that the host MY.NET.117.25 did not answer the request from host 63.240.142.227.

c) ARIN search result for Source IP 63.240.142.227:

If you want to follow upon the originator of this attack, you may find the following information useful.

```
AT&T CERFnet (NETBLK-CERFNET-BLK-5)
```

```
   P.O. Box 919014
   San Diego, CA  92191
   US

   Netname: CERFNET-BLK-5
   Netblock: 63.240.0.0 - 63.242.255.255
   Maintainer: CERF

   Coordinator:
      AT&T Enhanced Network Services  (CERF-HM-ARIN)
notify@attens.com
      (858) 812-5000

   Domain System inverse mapping provided by:

   DBRU.BR.NS.ELS-GMS.ATT.NET  199.191.128.106
   CBRU.BR.NS.ELS-GMS.ATT.NET  199.191.128.105
   DMTU.MT.NS.ELS-GMS.ATT.NET  12.127.16.70
   CMTU.MT.NS.ELS-GMS.ATT.NET  12.127.16.69
```

c) Defensive recommendation:

Although these are false positives in this case, the following defensive measures are recommended for prevention purposes:

- If not allowed by the University's policy, set up a firewall rule to block it. Note that the port could be different to 31337.
- Update all antivirus software university wide.
- Check other systems for BO2K files. There are a couple of ways to check for this Trojan (see www.sans.org/infosecFAQ/malicious/back_orifice.htm)
- Check the sign of compromise for the hosts listed in Table-3.14.

3.5.6 Alerts' Top 10 Talkers and Destinations in the whole alert log file

Talker	Count		Destination	Count
my.net.100.208	1310181		10.0.0.1	51359
my.net.84.234	481329		233.28.65.148	32115
3.0.0.99	51359		192.168.0.216	24208
63.250.213.12	32117		233.2.171.1	17945
194.98.189.139	8375		NULL	6069
my.net.85.74	6990		233.28.65.173	4975
my.net.111.230	6090		my.net.104.204	4489
my.net.111.231	6059		207.200.86.97	4386
my.net.109.105	6053		my.net.117.137	3476
my.net.111.219	6007		207.200.86.66	3217

Table-3.15 Top 10 talkers and targets in the whole alert log file

Table-3.15 shows a fact that the most part of the alert traffic volume is from the internal network. Here I am not addressing the security issues for internal network. Let us

concentrate on the outside attacks to MY.NET.

From Table-3.15, the three top outside talkers are 3.0.0.99, 63.250.213.12, and 194.98.189.139. The following is the ARIN search results for these IP addresses:

1) ARIN search result for 3.0.0.99:

OrgName: General Electric Company
OrgID: GENERA-9

NetRange: 3.0.0.0 - 3.255.255.255
CIDR: 3.0.0.0/8
NetName: GE-INTERNET
NetHandle: NET-3-0-0-0-1
Parent:
NetType: Direct Assignment
Comment:
RegDate: 1988-02-23
Updated: 1998-11-12

TechHandle: GET2-ORG-ARIN
TechName: General Electric Company
TechPhone: +1-518-612-6672
TechEmail: GENICTech@ge.com

2) ARIN search and nslookup results for 63.250.213.12

OrgName: Yahoo! Broadcast Services, Inc.
OrgID: YAHO

NetRange: 63.250.192.0 - 63.250.223.255
CIDR: 63.250.192.0/19
NetName: NETBLK2-YAHOOBS
NetHandle: NET-63-250-192-0-1
Parent: NET-63-0-0-0-0
NetType: Direct Allocation
NameServer: NS1.YAHOO.COM
NameServer: NS2.YAHOO.COM
NameServer: NS3.YAHOO.COM
NameServer: NS4.YAHOO.COM
NameServer: NS5.YAHOO.COM
Comment: ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate: 1999-11-24
Updated: 2002-03-27

TechHandle: NA258-ARIN
TechName: Netblock Admin, Netblock
TechPhone: +1-408-349-7183
TechEmail: netblockadmin@yahoo-inc.com

> nslookup 63.250.213.12
Name: dal-qcwm213012.bcst.yahoo.com
Address: 63.250.213.12

3) ARIN search result for 194.98.189.139

OrgName: RIPE Network Coordination Centre
OrgID: RIPE

NetRange: 194.0.0.0 - 194.255.255.255
CIDR: 194.0.0.0/8
NetName: RIPE-CBLK2
NetHandle: NET-194-0-0-0-1
Parent:
NetType: Allocated to RIPE NCC
NameServer: NS.RIPE.NET
NameServer: AUTH03.NS.UU.NET
NameServer: NS2.NIC.FR
NameServer: SUNIC.SUNET.SE
NameServer: MUNNARI.OZ.AU
NameServer: NS.APNIC.NET
Comment: These addresses have been further assigned to users in
the RIPE NCC region. Contact information can be found in
the RIPE database at whois.ripe.net

RegDate: 1993-07-21
Updated: 1998-10-16

TechHandle: RIPE-NCC-ARIN
TechName: Reseaux IP European Network Co-ordination Centre S
TechPhone: +31 20 535 4444
TechEmail: nicdb@ripe.net

Again, from Table-3.15, there are two top hosts targeted by attacks from the outside: MY.NET.104.204 and MY.NET.117.137. These two hosts are identified as KaZaA servers or clients. Although this is not the exclusive traffic from the outside to internal network, it really indicates that there is quite a few amount of P2P file sharing traffic flew between the outside world and the internal network. Such traffic could seriously decrease the performance of campus' network if the University does not take an appropriate measure to this.

3.5.7. Top destination ports in the whole alert log file

Port	Count	Related service
80	1822638	http
137	81434	Netbois
5779	37262	Other Internet access point
56464	17945	Other Internet access point
111	14576	RPC
NULL	6382	Crafted package
1214	4892	KaZaA
32771	3273	SUNRPC
27374	2304	Trojan server
6667	2054	IRC

Table-3.16 Top 10 ports in the whole alert log file

From Table-3.16, we see that the alert traffic is overwhelmingly passed over port 80. This fact conforms to the actual trend shown at http://www.dshield.org in which port 80 is the most attacked port.

3.6. Out-Of-Spec (OOS) analysis

Out of Specification (OOS) file contains the packages which do not conform to the rules as stated in the applicable RFC. They do not match a particular and existing Snort alert rule. In more concrete terms, they have unusual TCP flag combinations. Some sophisticate attackers can intentionally craft mutant packages to fingerprint or attack systems. If an operating system cannot handle these packages to the TCP/IP stacks, a system crash could be then expected. Thus, OOS packets can reveal valuable information to a potentially hostile person, or they may directly cause a denial of service.

The OOS raw data files are treated manually in view of the fact that they have different formats and sizes. By a quick look at the files, I found the header part is most interesting for making statistics purpose. Then I took advantage of some standard UNIX utilities such as vi, grep, cut, and sed to extract the header information in creating a file containing only oos-headers. For sack of space, I will not present these small scripts in the assignment. For readers who really need to learn how to make such scripts, I suggest them to read some assignments written by previous GCIA students: Jeffrey A. Holland, Scott Shinberg, Kyle Haugsness, etc. at http://www.giac.org/GCIA.php.

In our case, Snort recorded 1,637 OOS packets during August 1 to August 5 period. There are 74 distinct talkers in total. Each alert contains packets with various combination of TCP flag bit settings, such as 2*SFRPAU, **SF*P**, **SFRP**, etc. Packets with these types of settings are often used for reconnaissance and fingerprinting. Packets of this nature can also be used as a DoS if the target operating system or application does not know how to handle them.

3.6.1 OOS' Top 10 Talkers and Destination Ports

Source IP	Count		Dest Port	Count
68.32.126.64	652		110	652
62.76.241.129	345		113	355
209.116.70.75	214		25	280
212.35.180.17	83		80	166
65.210.154.210	48		21	75
213.250.44.19	29		4662	54
61.132.74.239	18		6346	25
209.132.232.101	18		6347	3
202.155.91.142	18		888	2
211.154.85.159	17		4389	2

Table-3.17 Top 10 Out-of-Spec talkers sorted by Source IP activity

From Table-3.17, we can easily find out that the most popular port is 110 and 25 which are reserved for POP3 and SMTP. Both are Email related ports. The other three are 113 (Auth/Ident), 21 (FTP), and 80 (HTTP). The top 10 source IPs of OOS log data are listed in Table 3-17. All of them targeted the hosts on MY.NET, the top 10 of which are equally shown in Table-3.18.

3.6.2 Top 10 OOS Destinations

Destination	Count
MY.NET.6.7	660
MY.NET.97.217	241
MY.NET.97.238	104
MY.NET.100.217	95
MY.NET.253.20	85
MY.NET.111.198	54
MY.NET.100.165	43
MY.NET.253.125	41
MY.NET.253.114	37
MY.NET.6.40	34

Table-3.18 Top 10 OOS Destinations

3.6.3 Excerpts of OOS logs

a) Let us now extract some log data related to these ports. We pay special attention to the Window Size, TTL, MSS, DF bit, and TCP options.

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
08/01-00:04:05.437159 68.32.126.64:26053 -> MY.NET.6.7:110
TCP TTL:48 TOS:0x0 ID:54125  DF
21S***** Seq: 0x6FF77DB   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 51881142 0 EOL EOL EOL EOL

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
08/01-00:53:16.983809 209.116.70.75:35740 -> MY.NET.6.34:25
TCP TTL:51 TOS:0x0 ID:44237  DF
21S***** Seq: 0xCD8C9185   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 770391368 0 EOL EOL EOL EOL

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
08/01-00:25:36.030054 62.76.241.129:36622 -> MY.NET.97.217:113
TCP TTL:45 TOS:0x0 ID:39364  DF
21S***** Seq: 0x65BBACF8   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 59651388 0 EOL EOL EOL EOL
```

b) Some different oos records:

Most packets in OOS file are shown above with unusual numbers for ACK number and window size. Below let us show some different packets with both Congestion Window Reduced (CWR) and Explicit Congestion Notification (ECN-Echo) bits set. The traffic passes over the ports shown in Table-3.17. These packages could indicate transport layer crafted packets. But it is not so sure because while the packet is in transit, CWR and ECN-Echo bits can also be set by a router to ensure reliable delivery of this information to the receiver (see RFC 2481).

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
07/31-13:47:46.826996 217.81.180.174:1699 -> MY.NET.150.225:1214
TCP TTL:116 TOS:0x0 ID:32639  DF
21**RPAU Seq: 0x610000   Ack: 0xBD42331F   Win: 0x5010
BD 42 33 1F 2B FC 50 10 7F FF 60 7E 00 00 00 00  .B3.+.P...`~....
00 00                                            ..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
08/01-04:31:55.467461 202.155.91.142:30922 -> MY.NET.60.14:80
TCP TTL:47 TOS:0x0 ID:4836  DF
21S***** Seq: 0x6373735   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460
00 00                                            ..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
08/01-04:32:02.696591 202.155.91.142:31012 -> MY.NET.253.125:80
TCP TTL:47 TOS:0x0 ID:39048  DF
21S***** Seq: 0x6B7B02B   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460
00 00
```

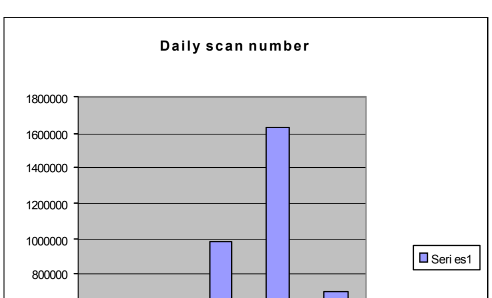
3.7. Link Graphs

Link graph is a way of graphing traffic based on the flow of data between IP addresses and/or ports. In our case, we are interested in some relationships between MY.NET.x.x and external networks. I will do it through different angles:

- Daily traffic patterns for both scans and alerts;
- The origin of the scans and alerts;
- Top 5 attacks (alerts) traffic flow.

3.7.1 Daily Traffic

From Fig.3.1, we can see that the sensor received much more attacks on August 5, 2002 than those on any other days. It is clear from the same figure at right side that on August 4, there were much more scanning activities going on. This is a common phenomenon that the attackers need to collect data about the network to be attacked before they can make the following attacks effective.



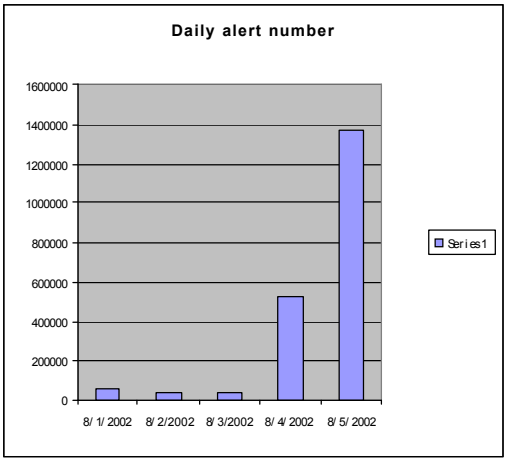


Fig.3.1 Daily traffic patterns for both scans and alert

3.7.2. Origin of the Attacks

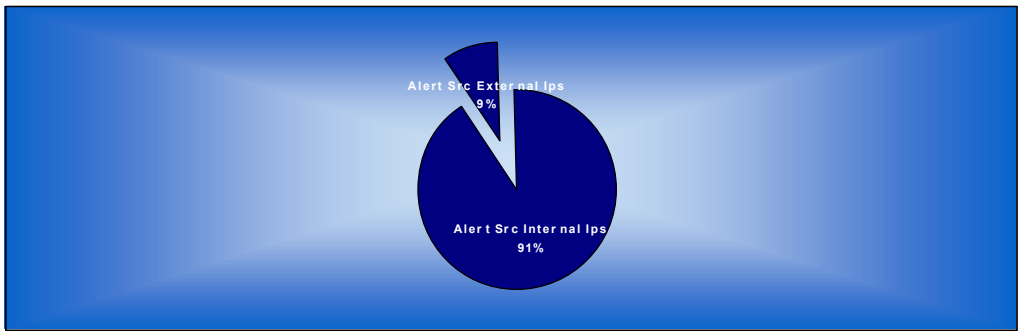


Fig.3.2 Scan source addresses percentage

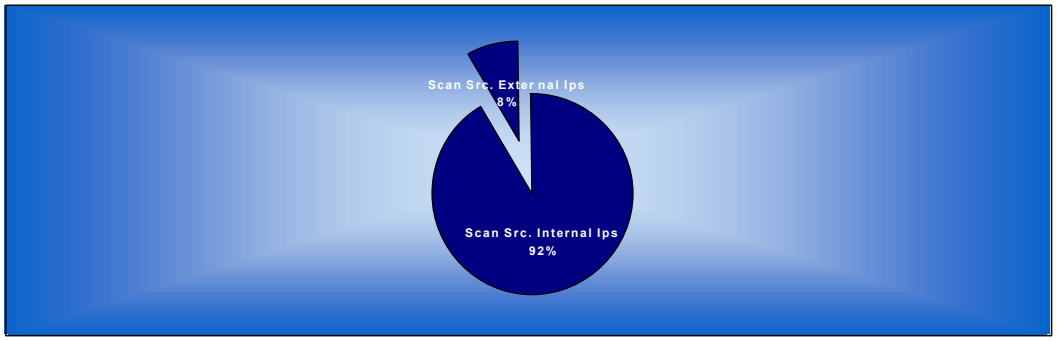
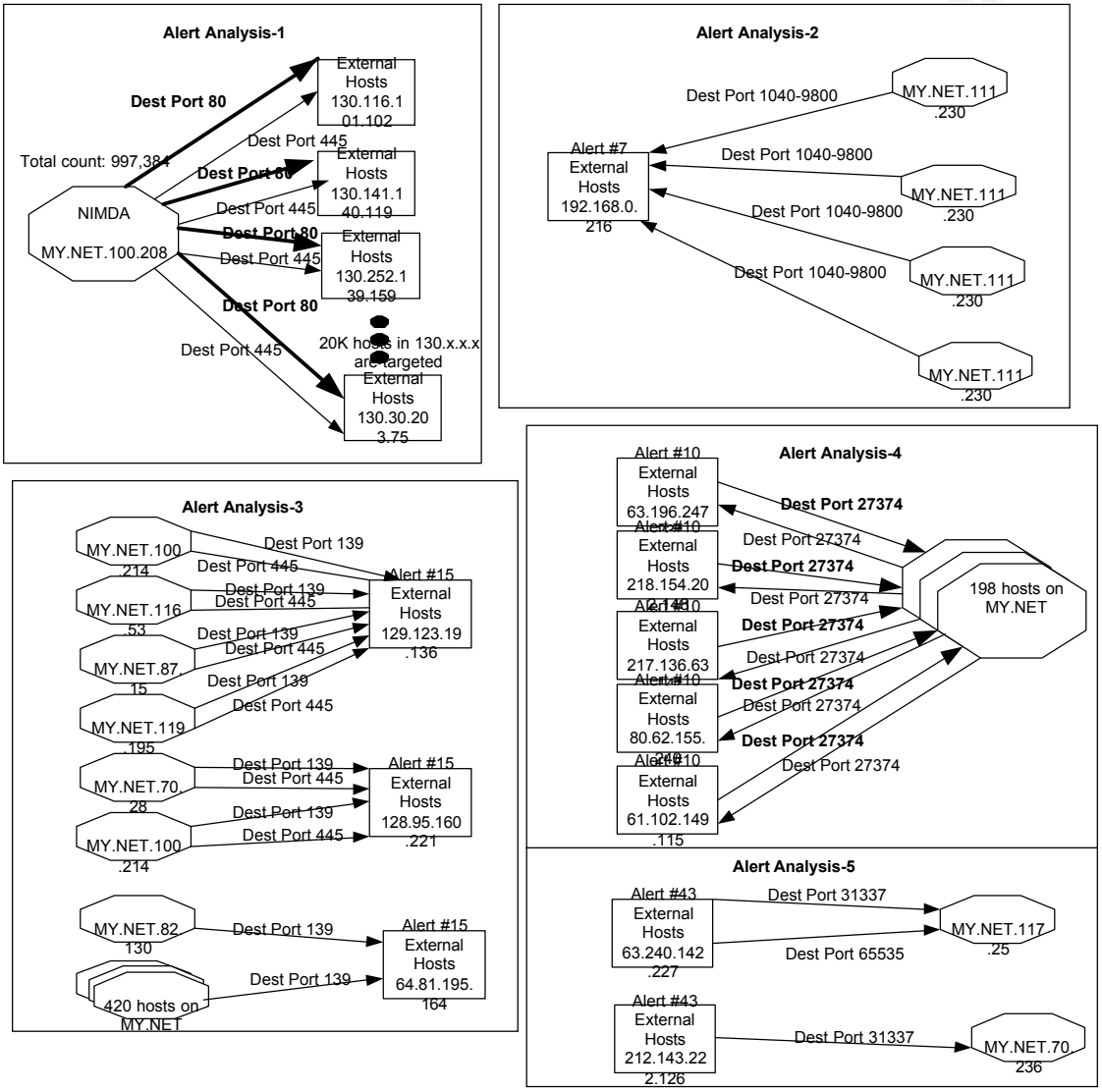


Fig.3.3 Alert source addresses percentage

People, even security professionals often overlook the importance of assessing security vulnerability and potential dangers to the outside world for their own internal networks. For example, in this University audit case, most people would concentrate on finding how attackers attack the MY.NET from outside, but not in reverse direction, i.e. from MY.NET to outside. Although it is not the purpose of this report to assess the 'damages' that could be caused by the attackers originated from MY.NET, I found that it is necessary to show

you a worth-to-concern phenomenon, i.e. most scans and alerts are originated from the internal network MY.NET! The pie charts in Fig.3.2 and Fig.3.3 show this clearly.

3.7.3 Top 5 Alerts traffic flow



Legend: A rectangle represents a host that is not on MY.NET.
An octagon represents a host on MY.NET.

Fig.3.4 Link graph of five top alerts

Fig.3.4 shows the traffic analysis for top 5 alerts analyzed in section 3.5. Based on the perceived importance of the alerts analyzed, I have identified the top 5 talkers for each alert if there exist. Fig.3.4 is manually generated and used for showing some aspects:

1. These alerts' top talkers target which hosts;

2. Through witch ports, and whether there exists the traffic in both directions for a particular alert. I do so in this way because I believe that the hosts which contain alerts occurred in both directions are worthwhile for a concerned system administrator to pay particular attention for their very possible compromise. This is even more realistic if the alerts analyzed are of nature of Trojan Horse and worm. This is also a helpful way for me to generate a possible compromised hosts list in the next section.

Alert Analysis-4 part in Fig,3.4 shows a sign that some hosts on MY.NET could be compromised. This problem has been pointed out in section 3.5.4. A full investigation on this attack activity is strongly recommended.

3.8. Defensive recommendations

3.8.1 Possibly Compromised Hosts on Campus Network

Before making my recommendations for enforcing campus network's security, I would present a list of possibly compromised hosts. This list is established based upon my knowledge about what kinds of attacks can lead host compromise in a high probability. Please note that it is not an exhaustive list, but the one that University's system administrators should pay particular attention in order to make a secure networking environment.

IP Address	Possible cause alert	IP Address	Possible cause alert
MY.NET.100.208	NIMDA	MY.NET.70.28	Buffer overflow (alert #15,33,36)
MY.NET.70.144	NIMDA	MY.NET.82.130	Buffer overflow (alert #15,33,36)
MY.NET.70.16	NIMDA	MY.NET.115.86	Buffer overflow (alert #15,33,36)
MY.NET.70.169	NIMDA	MY.NET.84.184	Trojan Horse (alert # 10)
MY.NET.105.10	NIMDA	MY.NET.84.189	Trojan Horse (alert # 10)
MY.NET.111.230	Trojan Hose (alert # 7)	MY.NET.85.66	Trojan Horse (alert # 10)
MY.NET.111.231	Trojan Hose (alert # 7)	MY.NET.85.92	Trojan Horse (alert # 10)
MY.NET.109.105	Trojan Hose (alert # 7)	MY.NET.85.96	Trojan Horse (alert # 10)
MY.NET.117.25	Trojan Hose (alert # 7 and #43)	MY.NET.218.150	Back Orifice (alert # 43)
MY.NET.111.219	Trojan Hose (alert # 7)	MY.NET.154.27	SUN RPC (alert #12)
MY.NET.100.214	Buffer overflow (alert #15,33,36)	MY.NET.163.132	SUN RPC (alert #12)
MY.NET.116.53	Buffer overflow (alert #15,33,36)	MY.NET.139.45	SUN RPC (alert #12)
		MY.NET.115.18	SUN RPC (alert #12)

Table 3-19 Possibly compromised campus hosts to be checked out

3.8.2 Recommendations

After my Audit of the data provided by the University, I would recommend the University pursuing the following in order to enhance the University's information security posture:

- Review the University's security policy. In order to have a secure and fast networking environment, a restriction rule of use of peer-to-peer file sharing services such as Napster, Gnutella, and KaZaA, may be necessary.
- Implement egress filtering at border routers and/or firewalls to prevent malicious packets from leaving the University's network. (see Fig.3.2 and Fig.3.3 and http://www.incidents.org/protect/egress.php.)
- Protect the campus network proactively by performing network assessment on a regular basis. Use the output information of the assessment to adjust the firewall and IDS rule-sets.
- Review each of the computers listed in Table 3-19 for possible compromise or mis-configurations.
- Unless it is absolutely necessary to allow remote printing from the outside of the University's network, port 515 should be blocked out by the firewalls.
- Add the IP addresses in Table 3-2, Table-3.12, and Table-3.13 to a watch list to determine if those computers are being used for hostile activities from non-University computers listed in Table-3.15 as top talkers. The ARIN search results for those talkers can be very helpful in such an investigation.
- Implement reliable and scalable Anti-Virus (AV) software onto Campus windows systems. A regular AV scan and software upgrading should be proceeded.
- Finally, security awareness program and training to campus network users and University's system administrators are always an important part of security plan as a whole.

3.9. Analysis process

Overall, the analysis process consists of four parts: preparation and raw data download, raw data analysis and transformation, SQL tables establishment, and analysis in depth for each type of log file. The description of these parts is shown in the following sub-sections.

3.9.1. Preparation and raw data downloading

In view of the data to be analyzed, a good analysis platform is necessary. The analysis platform is an IBM ThinkPad T23, which is a Pentium III 1.0G machine with 512MB memory and a 48GB hard drive. The operating system is Windows 2000 Professional. Cygwin is installed for using standard Unix tools like grep, cut, uniq, etc. A MS SQL 2000 personal edition is installed for data mining.

The analysis raw data was downloaded from the SANS Web site at

http://www.incidents.org/logs. The files downloaded along with their size are shown in section 3.2.

3.9.2. Raw data analysis and transformation

The following steps are taken in dealing with the alert raw log data

1. Review manually each downloaded log files shown in section 3.2 in order to have general idea about their contents and formats.
2. Remove "spp_portscans" and other duplicated entries in each alert log file
3. Concatenate log files of five days into a single alert.log file
4. Create a perl script to transform the alert.log file into a file of CSV form. This file called alert.log.csv.

Note that the process for dealing with scans log files is pretty much the same as described above for alert log files. The output of the scans.log files for 5 days is a concatenated file called scans.log.csv.

Note also that as indicated in section 3.2, OOS logs are handled manually. According to the needs of analysis, some small "quick-and-dirty" Unix shell scripts are written. As an example, in order to extract the OOS headers from the raw data, the follow command is used:

#> grep "\->" oos-log > oos-headers

3.9.3. SQL tables establishment

Having created two CSV files for both Alerts and Scans logs, I use the MS-SQL's DTS (Data Transfer Service) utility to transfer alert.log.csv file into the database table, let's call it alerts-8-01-05. This is a flat table with 6 columns and one real alert sample entry as shown in Table-3.20:

1	2	3	4	5	6
Date	Alert	Src IP	Src Poart	Dest IP	Dest Port
1/8/2002 0:00	UDP SRC and DST outside network	3.0.0.99	137	10.0.0.1	137

Table-3.20 SQL Alerts table

The Scans database table called scans-8-01-05 is shown in Table-3.21:

1	2	3	4	5	6	7	8
Date	Time	Src IP	Src Port	Dest IP	Dest Port	Protocol	Other
1-Aug	0:01:59	MY.NET.137.7	1588	209.240.213.109	25	SYN	******S*

Table-3.21 SQL Scans table

3.9.4. Analysis in depth for each type of log file

Base upon the established SQL tables for alerts and scans, and oos-headers file, I have performed a in-depth analysis for each tables and file by leveraging both intrusion detection analysis methodology such as I.W (Indication and Warning model) learned from the SANS online GCIA course material, and the relational database knowledge and technology. In brief, it consists of:

1. Based on all raw data, try to find intrusion indications and raise the warning by focusing on active targets (see section 3).
2. For all alert data analysis, assign an alerts metric in order to estimate their risk to the campus network.
3. Process the analysis by concentrating on the identified critical attacks or events, and then make some concerned and relevant data statistics by querying the database tables.
4. After making the analysis for each type of data, correlate the analysis results by using link graph method described also in SANS online GCIA course material.

The concrete analysis processing steps for all three types of log: scans, alerts, and OOS are described in sections 3.4, 3.5, and 3.6 respectively.

3.10. References

[1] O'Reilly Network: Morpheus Out of the Underworld.
URL: http://www.openp2p.com/lpt/a/p2p/2001/07/02/morpheus.html
[2] KaZaA Media Desktop
URL: http://www.kazaa.com
[3] SANS Institute Resources, Global Incident Analysis Center, Detects Analyzed 9/1/00
URL: http://www.sans.org/y2k/090100.htm
[4] NETSYS.COM SuSE Linux Security Mailing List Archives
URL: http://www.netsys.com/suse-linux-security/2001/01/msg00227.html
[5] American Registry for Internet Numbers
URL: http://www.arin.net
[6] Queso application on Apocalypse Online Security
URL: http://www.apocalypseonline.com/security/tools/tools.asp?exp_-category=Scanners
[7] CERT Advisory CA-2000-17 Input Validation Problem in rpc.statd
URL: http://www.cert.org/advisories/CA-2000-17.html
[8] Insecure.Org -- Nmap Free Stealth Network Port Scanner,
Linux/Windows/UNIX/Solaris Tools & Hacking
URL: http://www.insecure.org
[9] CERT Advisory CA-2001-31 Buffer Overflow in CDE Subprocess Control Service
URL: http://www.cert.org/advisories/CA-2001-31.html
[10] G-Lock Software, Trojan List, WinHole.
URL: http://www.glocksoft.com/trojan_list/WinHole.htm
[11] Snort - The Open Source Network IDS
URL: http://www.snort.org
[12] http://www.hackfix.org/subseven/

Appendix A: Brief Description of all alerts detected

Most information and definitions come from www.whitehats.com/IDS. I found that one can find more Snort alert definitions on whitehats.com than those on www.snort.org itself. Some other information comes from my search on the Internet and some from previous GCIA students whose names I will not enumerate here.

Alert	Description
NIMDA - Attempt to execute cmd from campus host	The Nimda worm is responsible for these attacks. The worm connects to the HTTP server at port 80/tcp on the target machine multiple times. Each time the worm connects it sends a specially crafted URL request designed to allow the worm to execute commands on the target. This attack exploits the back door vulnerability created by a previous infection with the Code Red II worm as well as the directory traversal via extended Unicode in URL vulnerability (CAN-2000-0884)
IDS552/web-iis_IIS ISAPI Overflow ida INTERNAL nosize	This event is likely the probe of the Code Red Worm trying to exploit a vulnerability in Microsoft IIS. An unchecked buffer in the Microsoft IIS Index Server ISAPI Extension could enable a remote intruder to gain SYSTEM access to the web server.
spp_http_decode: IIS Unicode attack detected	This is a Snort Preprocessor Plugin that converts Unicode traffic and null bytes in CGI's to non-obfuscated ASCII strings. By using Unicode and null bytes attackers can bypass content analysis strings used to examine HTTP traffic for suspicious activity.
NIMDA - Attempt to execute root from campus host	The Nimda worm is responsible for these attacks. The worm connects to the HTTP server at port 80/tcp on the target machine multiple times. Each time the worm connects it sends a specially crafted URL request designed to allow the worm to execute commands on the target. This attack exploits the back door vulnerability created by a previous infection with the Code Red II worm as well as the directory traversal via extended Unicode in URL vulnerability (CAN-2000-0884)
UDP SRC and DST outside network	This alert reports that neither the source nor the destination IP addresses are contained within the internal network. While this may be totally harmless, it is anomalous traffic and could indicate packet crafting.
SMB Name Wildcard	This event indicates a standard netbios name table retrieval query. Windows machines often exchange these queries as a part of the file-sharing protocol to determine NetBIOS names when only IP addresses are known. An attacker could use this same query to extract useful information such as workstation name, domain, and users who are currently logged in.

TFTP - External UDP connection to internal tftp server	This alert indicates that an external host is connecting to an internal tftp server. This could indicate a compromised host, a trojan, or an internal user violating policy.
External RPC call	This alert indicates that an external host, possibly hostile, has tried to access one of the internal hosts Remote Procedure Call (RPC) ports.
Watchlist 000220 IL-ISDNNET-990517	The watchlist is provided because of the frequency of scans that are launched from the offending network. The IL-ISDNNET indicates an ISP called ISDNNET located in Israel. It is provided as a signature, and the recommendation is to keep a close watch on the types of traffic coming into your network. If you are able to block these addresses at the firewall without impacting your business, it is recommended that you do so.
Possible Trojan server activity	This event alerts to the fact that an internal server is answering queries on a high port (> than 1024).
spp_http_decode: CGI Null Byte attack detected	This is a Snort Preprocessor Plugin that converts Unicode traffic and null bytes in CGI's to non-obfuscated ASCII strings. By using Unicode and null bytes attackers can bypass content analysis strings used to examine HTTP traffic for suspicious activity.
SUNRPC highport access!	This incident indicates that a SUNRPC port (in this case port 443) was probed from a port above 1024. This could be legitimate, a reconnaissance probe, or an actual exploit.
IRC evil - running XDCC	
Watchlist 000222 NET-NCFC	The watchlist is provided because of the frequency of scans that are launched from the offending network. The NET-NCFC is the Computer Network Center Chinese Academy of Sciences. It is provided as a signature, and the recommendation is to keep a close watch on the types of traffic coming into your network.
EXPLOIT x86 NOOP	This event may indicate that a string of the character 0x90 was detected. Depending on the context, this usually indicates the NOP operation in x86 machine code. Many remote buffer overflow exploits send a series of NOP (no-operation) bytes to pad their chances of successful exploitation.
Queso fingerprint	Queso is a tool used for OS fingerprinting on a targeted host.
SNMP public access	A lot of network devices (such as intelligent switches, WAN/LAN routers, ISDN/DSL modems, remote access machines and even some user-end operating systems) are by default configured with SNMP enabled and unlimited access with write privileges. This allows attackers to modify routing tables, get the status of network interfaces and other vital system data, and is considered extremely dangerous from a security perspective.

connect to 515 from outside	This event could signal a LPRng buffer overflow attack. LPRng is a linux printer server.
Attempted Sun RPC high port access	See alert SUNRPC highport access!
Samba client access	Samba is software which permits Unix file systems to be shared with Windows file systems.
High port 65535 udp - possible Red Worm – traffic	Normal traffic should never access port 65535. This alert indicates that whoever wrote the rules file for Snort noticed Code Red Worm traffic accesses port 65535
IDS552/web-iis_IIS ISAPI Overflow ida nosize	This event is likely the probe of the Code Red Worm trying to exploit a vulnerability in Microsoft IIS. An unchecked buffer in the Microsoft IIS Index Server ISAPI Extension could enable a remote intruder to gain SYSTEM access to the web server.
ICMP SRC and DST outside network	This alert reports that neither the source nor the destination IP addresses are contained within the internal network. While this may be totally harmless, it is anomalous traffic and could indicate packet crafting.
SMB C access	This event indicate that the attacker attempts to get access to the C$ default file share via NETBOIS session.
TFTP – Internal UDP connection to external tftp server	This alert indicates that an internal host is connecting to an external tftp server. This could indicate a compromised host, a trojan, or an internal user violating policy.
beetle.ucs	**Beetle.ucs** is a host that houses a CD-R. This alert indicates that users are copying information form the Internet and saving it to a CD-R.
Port 55850 tcp - Possible myserver activity - ref. 010313-1	MyServer is a Trinoo-style Denial of Service tool that usually communicates over port 55850.
Incomplete Packet Fragments Discarded	This event describes that an IP datagram was fragmented and all fragments did not arrive. This could be innocent or it could indicate an attacker performing some form of reconnaissance.
NMAP TCP ping!	This event indicates that a remote user has used the NMAP portscanning tool to probe the server. An NMAP TCP ping was sent to determine if a host is reachable.
Null scan!	This event indicates that a TCP frame has been seen with a sequence number of zero and all control bits are set to zero. This frame should never be seen in normal TCP operation. An attacker may be scanning the system by sending these specially formatted frames to see what services are available.
EXPLOIT x86 setuid 0	This event may indicate an exploit attempt where the attacker sent the setuid(0) system call for the x86 platform. This signature is the most effective when monitoring protocols that usually consist of plaintext printable ASCII to catch remote x86 exploits.

Tiny Fragments - Possible Hostile Activity	The smallest fragment that should be sent/receive is 25 bytes; this event triggered on a fragment that was smaller that 25 bytes.
EXPLOIT x86 stealth noop	This event may indicate that someone attempted to overflow one of your daemons with jmp 0x02 "stealth nops".
High port 65535 tcp - possible Red Worm – traffic	Normal traffic should never access port 65535. This alert indicates that whoever wrote the rules file for Snort noticed Code Red Worm traffic accesses port 65535.
STATDX UDP attack	This alert indicates that a buffer overrun condition has been discovered in the statd daemon program. The condition may be exploited both by a local user and a remote user. An intruder could force the statd daemon to execute commands as the user running statd, which is most often root. The statd daemon is usually part of the NFS environment under UNIX. For more information, http://www.securityfocus.com/bid/1480
EXPLOIT x86 setgid 0	This event may indicate an exploit attempt where the attacker sent the setgid(0) system call for the x86 platform. This signature is the most effective when monitoring protocols that usually consist of plaintext printable ASCII to catch remote x86 exploits.
Port 55850 udp - Possible myserver activity - ref. 010313-1	MyServer is a Trinoo-style Denial of Service tool that usually communicates over port 55850.
TCP SRC and DST outside network	This alert reports that neither the source nor the destination IP addresses are contained within the internal network. While this may be totally harmless, it is anomalous traffic and could indicate packet crafting.
SMB CD...	This event indicates an attempt to circumvent directory access control by trying to change to the ".." directory.
External FTP to HelpDesk my.net.70.50	This alert indicates a FTP connections has been established to the internal HelpDesk, originating form outside the network.
EXPLOIT NTPDX buffer overflow	Thisevent indicates that a buffer overflow exploit was attempted against the ntpd network time daemon. Some versions of ntpd and xntpd are vulnerable to remote root access.
RFB - Possible WinVNC - 010708-1	The VNC protocol is a simple protocol for remote access to graphical user interfaces. It is based on the concept of a *remote framebuffer* or *RFB*. This alert indicates a communication connection under this protocol.
Back Orifice	This event indicates that a remote attacker has sent an information request to a Back Orifice trojan. If the trojan is running on the server, then the server has been compromised.
DDOS shaft client to handler	This event indicates possible control traffic from a Shaft handler to a Shaft zombie. If a zombie is present, your host may be compromised and it may be used to attack other internet sites. Shaft is a distributed denial of service (DDoS) tool.

Traffic from port 53 to port 123	This is only a traffic indication. Port 123 is used by NTP (Network Time Protocol).
SYN-FIN scan!	This event indicates a SYN-FIN scan packet, where the TCP packet had both the SYN and the FIN flag set. This can be used in stealth portscanning.