Global Information Assurance Certification Paper

GCIA

Practical Assignment V3.0

PARLIEMENT HILL CONFERENCE OTTAWA

Prepared by:
Clément Dupuis, CD
CISSP – GCFW – Wannabe GCIA

Last Revised: 02 January 2002

TABLE OF CONTENT

1 INTRODUCTION

This document is to fulfill the requirement of the SANS GCIA practical.

As such I will present in the first part the current situation and an approach to ease the current challenge of establishing correlation, ensuring availability, and maintaining integrity of key files on all of the SNORT sensors deployed within an enterprise. This first section will present a tool named Demarc that is currently in development but already offering a high level of integration and most importantly cater to some of the shortcoming of the SNORT Intrusion Detection Tool.

Part 2 of the document is an analysis of five network detects in the format specified in the practical assignment. The detects will be analyzed to attempt to determine if it was targeted attack, what was the attack, severity, and more details.

The final portion which is part 3, is a security audit performed for a University. They provided 5 days worth of data from a SNORT system that was using a fairly standard rulebase. From the data provided we will analyze what type of attacks took place, what are the most offending IP's, what are the most common IP's being used, sign of compromised or network problems.

THE MEAT

Below in the document, you will read how companies are marketing ID tools as being the key to your problem and a must have for anyone that cares about their security.

These vendors often present ID as a panacea to the shortcoming of the other devices in your infrastructure. However, one will quickly realized that it is a very different view when you are the person that must look after and fine tune the beast after it was deployed and launch into production. The acquisition and deployment phase is the easy part, the tough part comes when you must fine tune your intrusion detection tool until it reach the fine equilibrium between number of attack detected versus false positive versus false negative. This is not an easy task.

Another great challenge that is never mentioned by vendors is the ability and necessity to consolidate data from multiple sensors at a central location to reach a higher level of early warning detection, patterns, or low level of very similar probes.

Most often great tools such as SNORT and even other well-known commercial IDS tools are not very adept at doing this in an efficient manner. They have fantastic detection engine but the management part is not as user friendly as it could be.

Why do we have to go through such trouble you must be asking yourself? It seems that we are not progressing but regressing from what was in place in our security infrastructure a few years ago but do not despair there is help at the horizon.

I strongly believe that it is no longer a matter of choice if you wish to be competitive and stay in business you must have the proper tools in place. The current situation with network security has greatly evolved over the past few years, best practices now dictates that you must have multiple tools deployed as a series of layers to provide defense in depth for your environment. This requirement is driven by enterprises that are facing new challenges such as extranet, remote users, partner's access, and ecommerce. Such a wide range of services requires a wide range of devices to protect your networks. As a minimum you will most likely deploy VPN's, intrusion detection systems, Integrity checker, high availability, firewalls, and virus checker.

Such protection comes with a certain cost. The cost is that administrators have to deal with a series of management tools that are not centralized, that present different interfaces, sometimes on different operating systems, and often there is also a lack of qualified personnel or knowledge to properly administer or use all these tools.

Outsourcing may be an option for some companies but impossible for others that wishes to retain and maintain complete control over their security infrastructure. In the case of the client that I have mentioned above, outsourcing was unfeasible because of the quantity of nominative data on their clients, which are citizen requesting governmental services.

In order to address all of the above problems and ease management, it is highly suitable to have a single or a few centralize administration points and tools where attacks can be reported, integrity violation can be monitored, and last but not least the availability of systems can be monitored as well.

Later on in the document, I will present to you a way that this can be accomplished.

2 CURRENT STATE OF ID

Intrusion Detection (ID) is an area of computer security that is very quickly evolving. There is not a single week where we do not see a new announcement in this field of expertise. We are bombarded with new approach and tools to assist in improving the accuracy and speed at which we can identify, detect, prevent, or minimize attacks against our infrastructure.

As analysts in the field, we must keep a close look on vendors pushing propaganda about their products, they tell you how easy they are, how they can solve all of your problems. Every two to three months we notice the leapfrog effect between the top players in ID. It is sometimes very hard to see clearly through all of this FUD (Fear, Uncertainty, and Doubt) and marketing.

A new trend and very recent change noticed in the above pattern is the fact that we are now seeing open source tools such as SNORT in these benchmarks as well, open source tools are finally given the rewards they deserve. Computer magazine and vendors realize that even thou it is open source, it perform as well or in some case better than some of the expensive commercial IDS.

A recent benchmark had SNORT Version 1.7 placed as third best in Network Intrusion Detection Systems (NIDS) category (http://img.cmpnet.com/nc/1217/graphics/1217f2report_1.pdf) against nine other commercial NIDS. The only reason that it did not reach one of the top two positions is that SNORT lacks the ease of installation, graphical user-friendly configuration, reporting, and proper administration tool. To use SNORT you must be familiar with system administration and it is not always an out of the box solution.

I had the opportunity to work for a large Government Agency that is using SNORT as their main NIDS tool. They are using multiple SNORT sensors and through the use of add-on tools they have been able to greatly enhanced and easy the administration and support challenge of their whole collection of sensors. The tour of their facilities really impressed me and entices me to learn more about the open source ID tools that are available. Below I will introduce one of the tool that they currently use and I will show how it can reduced some of the greatest challenge faced by organizations using open source tools as their main Intrusion Detection Platform.

3 DEMARC

Before we get into the features of the Demarc tool and what problem it can solve, I would like to quickly review some of the basis on which security is built, how these basis relates directly to the ID domain. This brief overview will prepare the reader and give the proper basis to fully understand and appreciate the benefits of a tool such as Demarc.

3.1 The CIA Triad

The CIA triad is one of the well-known security management approaches. It is often represented by a triangle that is a symbol of equality of importance between the bases on which the approach is based on. I added Non-Repudiation on top of the triangle to show that even thou it is not formally part of the CIA triad, non-repudiation is now as important as the three properties of the CIA triad.

Confidentiality: this is the property that ensures that only authorized people are accessing the information and that it cannot be seen by anyone else, a common tool to enforce confidentiality is the use of encryption. An IDS is an important tool in monitoring breach of confidentiality or access by unauthorized person to files they are no supposed to access, this is even more important for files that are not encrypted. The IDS will detect such intrusion and warn you if someone accesses a specific file that is not supposed to be accessed. For example, it could be the detection of someone that attempts to access the list of current salaries on the server of the human resource department.

Integrity: this is the property that ensures there was no change done to the data while in transit or stored. Common tools to ensure the integrity are hashing mechanism such as MD5. Integrity

plays a big role in system security, integrity checkers ensures that web page, executables, log files have not been tempered and it is also used in some case for assistance in the detection of intrusions that have no known signature. For example, if your snort rule files are replace by a bad version that has no signature, then your IDS is rendered useless. A tool such as an integrity checker would quickly detect the changes and report it. Such tool would most certainly help the level of trust you can have in your Intrusion detection tool.

Availability: this is the property that ensures that access to the information is there when it is needed. If one of your remote sites no longer has network connectivity, it will not make much sense to monitor the environment using IDS. Lack of proper availability has the same effect as a Denial of service attack. The services cannot be rendered to the public. Early detection of unavailable system is important and is a part of security.

Non-Repudiation: this is the property that prevents an individual or entity from denying having performed a particular action related to data. A tool such as IDS can contribute to collecting information to demonstrate what has happened from the point of entry up to the point of exit and greatly helped in investigation, audit trails, and forensics evidence.

3.2 Now 'THE' tool

Demarc is a tool used to manage one or multiple SNORT sensors from a single web interface. The tool also allows you to monitor attacks that are being reported by all of your sensors from a central location. The same tool has features that allow you to ensure the integrity of selected files on your snort servers or any other server that you wish to monitor; it can also monitor connectivity of servers and availability of services as well.

All of the above security functions are usually performed by a series of disparate tools that do not share any commonality and forces the user to learn three different types of administrative functions. With Demarc all of these functions are grouped within a single interface that is very intuitive to use and has a common look and feel.

Demarc combines all three services into one powerful client/server program. Not only can you monitor the status of the different machines in your network, but you can also respond to changes in your network all from one centralized location.

3.3 Licensing of Demarc

Demarc is a tool that can be used freely for non-commercial uses. However there is some restrictions that applies to its usage in a commercial environment.

In order to use Demarc in a commercial environment you must first obtain a commercial license from the Demarc organization. However, there are a few exceptions to this license, they are listed below:

- Your company's primary business is as an ISP (Internet Service Provider) and has a customer base of fewer than 1000 users.
- Your company has fewer than 25 employees and is not an ISP.
- You have extenuating circumstances and have received written authorization from DEMARC Organization to use this software free of charge.

3.4 DEMARC Architecture

Demarc is built on a client on a client/server architecture. You only need one central database to receive the data from all of your sensors. The computer that has the database can also be used as a sensor if you wish. The remote sensor only need part of the Demarc installation in order to report to the central location. All reporting is done over SSL for better security.

The Demarc architecture allow you to grant access to the central console to remote users as well. It can be accessed with the simple use of a browser. Demarc access control can be enforce in different ways. You can make use of the mysql permissions along with specific hosts restriction access, you can make use of the Linux firewalling as well. Demarc can be configured to allow your remote sites to view their specific range of IP address or all of the other sensors according to the access rights you wish to assign to them. Access can also be restricted to specific IP address, or you could have specific database for each of the sensor, you can also make use of ipchains or iptables for more granular control.

It is highly recommended that you setup a different account for each sensor, and lock these sensors to the IP address of the host they are connecting from.

Although it is possible to run Demarc without SSL in plain http traffic. The installation includes all of the steps to configure SSL in order to have a more secure access to your central console. If you use it without https, it will give you warning about the lack of security.

If you are really concerned about having a very secure environment, I recommend that you lookup the **stunnel recipe** listed in the references at the end of this document.

3.5 Supported platforms and required software packages

Demarc has support for the following platforms:

- FreeBSD, Linux, OpenBSD, Solaris, and NetBSD.

The following packages must be installed:

- Snort version 1.8 or higher;
- MySQL 3.23 database server;
- Perl with the following Perl modules: CGI, DBI, DBD::mySQL, and Digest::MD5;
- Apache 1.3.22
- OpenSLL 0.9.6b

3.5.1 DEMARC Version 1.5 Stable Installation tips

Demarc installation is a bit tricky as you might have guessed. This tool works in conjunction with Mysql, Perl, SNORT, OpenSSL, and apache. All of these tools must be properly configured and compiled in order for Demarc to properly run.

Below I will give you the recipe for a **fresh** installation of Demarc on a RedHat Linux 7.2 server installation. The instructions below do not include the initial hardening that must be performed initially on the server. It is taken for granted that you have followed the SANS Securing Linux Step-by-Step guide to harden your server. Having contributed to the guide, I can attest that it is a very thorough guide and will provide an adequate level of security.

To get familiar with the Demarc jargon; there is two ways that you can deploy the software. It is either deployed as the main "Demarc Monitoring Console" which is the main client that will host the mysql server (note that the mysql db could also be on a separate host) and the apache server with the Demarc web site. The second type of deployment is as a remote sensor that is called a "Demarc client", such a client runs a copy of snort, there may also be integrity checks, process and log monitoring as well. Such a client reports back to the Demarc Monitoring console.

The installation instructions below assume that all programs will be built from source, which is the preferred way. It is also important to mention that all of the installation steps must be performed while logged as root.

The following packages are necessary for the installation of Demarc:

Name of package	URL to download package
Demarc-1.05-stable.tar.gz	http://www.demarc.com/downloads/demarc-105/demarc-1.05-stable.tar.gz
mysql-3.23.45.tar.gz	ftp://uiarchive.uiuc.edu/mirrors/ftp/ftp.mysql.com/MySQL-3.23
Libpcap-0.6.2.tar.gz	http://www.tcpdump.org/release/libpcap-0.6.2.tar.gz
snort-1.8.3.tar.gz	http://www.snort.org/releases/snort-1.8.3.tar.gz
snortrules.tar.gz	http://www.snort.org/downloads/snortrules.tar.gz
Openssl-0.9.6b.tar.gz	http://www.openssl.org/source
mod_ssl-2.8.5-1.3.22.tar.gz	http://www.modssl.org/source
Apache_1.3.22.tar.gz	http://www.apache.org/dist/httpd/
DBI-1.20.tar.gz	http://www.cpan.org/authors/id/TIMB/DBI-1.20.tar.gz
DBD-mysql-2.1005.tar.gz	http://www.cpan.org/authors/id/JWIED/DBD-mysql-2.1005.tar.gz
Digest-MD5-2.16.gz	http://www.cpan.org/authors/id/GAAS/Digest-MD5-2.16.tar.gz
Stable.tar.gz	http://www.cpan.org/src/stable.tar.gz

FIRST STEP – INSTALLATION OF DEMARC

The first step consist of installing the Demarc files. Use the following commands:

```
# tar fxvz demarc-1.05-stable.tar.gz
# mv demarc-1.05-stable /usr/local/demarc
```

NOTE:
There is a "tmp" directory included under "/usr/local/demarc", it is highly suggested that you make it readable/writable ONLY by the user the web server runs as. Usually this user is called "nobody". By default, this directory has full access permission that allows anyone full access which is a severe security threat. Use the following commands to set the proper permissions:

```
# chown nobody /usr/local/demarc/tmp
# chmod 700 /usr/local/demarc/tmp
```

Make sure that the first line of the following files you just copied has the correct path to the perl interpreter on your system. This usually does NOT need to be changed from the default of "#!/usr/bin/perl". You can use the more command to verify the first line of the following files:

```
/usr/local/demarc/bin/demarcd
/usr/local/demarc/cgi/demark
/usr/local/demarc/install/dm_load_db.pl
/usr/local/demarc/check_pms.pl
```

SECOND STEP – INSTALLATION OF MYSQL

In the directory where your source files are located, logged as root, issue the following 2 commands to uncompress, untar, and change to the directory that contain the source for mysql.

```
# tar zxvf mysql-3.23.45.tar.gz
# cd mysql-3.23.45
#
```

The next step consist of running the configure command with the proper options. The configure command is used to verify which platform you are using, locate some of the binaries that are needed at compile time, and other system variable. For example, below you have two options, the first one --without-server is an option to specify that you do not need the mysql server installed on the host, the --prefix=/usr/local/mysql specifies that the base path for the installation will be /usr/local/mysql.

If you are installing a Demarc client/Snort sensor and the machine will NOT be a Demarc monitoring console, you do not need to build the mysql server on that machine. Use the following syntax:

```
# ./configure - -without-server --prefix=/usr/local/mysql
```

However if the installation WILL be for your main Demarc monitoring console as well as your database server. Use the following syntax:

```
# ./configure  --prefix=/usr/local/mysql
```

After the configuration is completed you can complete the installation of mysql bye issuing the following commands, which will compile and then install the binaries on your system according to the options that you selected above.

```
# make
# make install
#
```

TIP!!

If you are planning to use the mysql client (/usr/local/mysql/bin/mysql) frequently on this machine, you may want to soft link the client to a directory in your $PATH. To identify which directories are in your patch, simply use the echo command followed bye the variable $PATH as demonstrated below:

```
# echo $PATH
/usr/local/sbin:/usr/sbin:/sbin:/bin:/usr/bin:/usr/local/bin:/root/bin
# ln -s /usr/local/mysql/bin/mysql /usr/local/bin/mysql
#
```

If this is a Demarc Monitoring console installation, you must execute the following steps as well. If your installation is for a Demarc Client, you can skip to the next section on installing XXXX.

```
# scripts/mysql_install_db
```

It is now necessary to add a new mysql user and group on your system.

This new user will be called: mysql This user does not need access to a shell, it does not have to log onto the system. The mysql user will be a member of a group called: mysql

Use the following commands:

```
# groupadd -g 3306 mysql
# useradd -d /usr/local/mysql -c Mysql_Server -u 3306 -g 3306 -s /bin/true mysql
```

Now you must create the mysql var directory with the following command:

```
# mkdir /usr/local/mysql/var
# chown -R root /usr/local/mysql
# chown -R mysql /usr/local/mysql/var
# chgrp -R mysql /usr/local/mysql
# cp support-files/my-medium.cnf /etc/my.cnf
```

You have to configure the dynamic linker run-time binding by issuing the commands below:

```
#  echo /usr/local/mysql/lib/mysql >> /etc/ld.so.conf
#  ldconfig
```

You can start mysql with the following command:

```
# /usr/local/mysql/bin/safe_mysqld &
```

This method of starting mysql above is not permanent; upon reboot mysql will not restart. In order to make these changes permanent your must copy the file named mysql.server to /etc/init.d directory and then use the chkconfig command to enable and disable it at the proper run level. Use the following command:

```
# cp /usr/local/mysql/share/mysql/mysql.server  /etc/rc.d/init.d/
# chkconfig – -level 2345 mysql.server on
# chkconfig – -level 01 mysql.server off
# chkconfig –list mysql.server
mysql.server  0:off   1:off   2:on   3:on   4:on   5:on   6:off
#
```

We must now specify a ROOT password for mysql; use one of the lines below, when prompt to enter your current password, simply press enter because the default password is blank. Use the following commands:

```
# /usr/local/mysql/bin/mysqladmin -u root -p password 'new-password'
enter password:
```

Or you may have to use the following command instead:

```
# /usr/local/mysql/bin/mysqladmin -u root -h redhat71 -p password 'new-password'
```

THIRD STEP – INSTALLATION OF SNORT

To install Snort, you will need "lex" and "yacc" or the gnu version "flex" and "bison". Most installation have these packages installed by default. To verify if these utilities are installed on your system, you can use the which command and the patch where it is installed will be presented:

```
#
# which yacc
/usr/bin/yacc
# which lex
/usr/bin/lex
# which flex
/usr/bin/flex
# which bison
/usr/bin/bison
#
```

You will also need libpcap installed as well. To verify if it is installed you can use the find command. If no files are found, it means that it is not installed on your system:

```
# find / -name *libpcap*
#
```

If the libpcap library is not installed, you must install it as follow:

```
# tar zxvf libpcap-0.6.2.tar.gz
# cd libpcap-0.6.2
# ./configure
# make
# make install
# find / -name *libpcap*
/usr/local/lib/libpcap.a
#
```

Now that we have ensured that all prerequisite are met, we can proceed with the installation of SNORT. Follow the steps below to install it:

```
# tar zxvf snort-1.8.3.tar.gz
# cd snort-1.8.3
# ./configure  --with-mysql=/usr/local/mysql/
# make
# make install
```

The next step consist of testing your snort binary to see if there is any errors detected. You should have an output similar to the output below. If you receive error messages, such as the one about the portscan.log file that cannot be found below, move to the section on dealing with error messages below and we will explain how to fix these errors.

```
# snort –T
Log directory = /var/log/snort
Initializing Network Interface eth0
using config file ./snort.conf
Initializing Preprocessors!
Initializing Plug-ins!
Initializating Output Plugins!
Parsing Rules file ./snort.conf
+++++++++++++++++++++++++++++++++++++++++++++++++++
Initializing rule chains...
No arguments to frag2 directive, setting defaults to:
Fragment timeout: 60 seconds
Fragment memory cap: 4194304 bytes
Stream4 config:
Stateful inspection: ACTIVE
Session statistics: INACTIVE
Session timeout: 30 seconds
Session memory cap: 8388608 bytes
State alerts: INACTIVE
Scan alerts: ACTIVE
Log Flushed Streams: INACTIVE
No arguments to stream4_reassemble, setting defaults:
```

Reassemble client: ACTIVE
Reassemble server: INACTIVE
Reassemble ports: 21 23 25 53 80 143 110 111 513
Reassembly alerts: ACTIVE
Back Orifice detection brute force: DISABLED
fopen: No such file or directory
spp_portscan: logfile open error (/var/log/snort/portscan.log)

DEALING WITH ERROR MESSAGES

While testing the SNORT binary there is a few error messages that can be presented.

The first type of error may be:

Snort: error while loading shared libraries: libmysqlclient.so.10: cannot open shared object file: No such file or directory

You can correct this error message by copying the libmysqlclient.so.10 to the /usr/lib directory. Use the following commands:

cp /usr/local/mysql/lib/mysql/libmysqlclient.so.10 /usr/lib
/sbin/ldconfig

The second error message that you may encounter is:

fopen: no such file or directory
spp_portscan: logfile open error (/var/log/snort/portscan.log)

You can fix this error message by creating the snort directory under the /var/log directory. As mentioned above, this error was present on our system, you must ensure that you use the following commands to correct it:

mkdir /var/log/snort

The third error message that you may see is:

ERROR: Unable to open rules files: …

Don't worry about the above message, your binaries are fine but it cannot find your rules files.

Once all error messages are fix, ensure that you test your SNORT library again to see if there is any other errors. If there is no errors you should see a message that says "Snort successfully loaded all rules and checked all rules chains!" this is a good sign. Use the following command:

```
# snort –T
Log directory = /var/log/snort
Initializing Network Interface eth0
……
Part of output that is exactly the same as above snipped for brevity
……
884 Snort rules read...
884 Option Chains linked into 93 Chain Headers
0 Dynamic rules
+++++++++++++++++++++++++++++++++++++++++++++++++++
Rule application order: ->activation->dynamic->alert->pass->log
--== Initializing Snort ==--
Decoding Ethernet on interface eth0
--== Initialization Complete ==--
-*> Snort! <*-
Version 1.8.3 (Build 88)
By Martin Roesch (roesch@sourcefire.com, www.snort.org)
Snort sucessfully loaded all rules and checked all rule chains!
```

FOURTH STEP – INSTALLATION OF APACHE-MODSSL-OPENSSL

The next step of the installation consists of installing apache, openssl, and the mod_ssl packages. From the directory where you have all of the downloaded files, issue the following commands:

```
# tar zxvf openssl-0.9.6b.tar.gz
# tar zxvf mod_ssl-2.8.5-1.3.22.tar.gz
# tar zxvf apache_1.3.22.tar.gz
```

We will start by building openssl with the following commands:

```
# cd openssl-0.9.6b
# sh config
# make
# make test
# make install
```

We will now build mod_ssl with the following commands:

```
# cd ../mod_ssl-2.8.5-1.3.22
# ./configure   -- with-apache=../apache_1.3.22
```

We will now build apache with the following commands:

```
# cd ../apache_1.3.22
# SSL_BASE=../openssl-0.9.6b ./configure  --enable-module=ssl   -- enable-module=so
-- prefix=/usr/local/www/
# make
# make certificate
# make install
# /usr/local/www/bin/apachectl startssl
#
```

The last command above will start your apache server. If you specified a password for your certificate in the steps above, you will be prompted to enter the password in order to start SSL.

FIFTH STEP – INSTALLATION OF PERL AND THE REQUIRED MODULES

Demarc needs the following perl modules to function properly:

DBI	Perl Database independent interface
CGI	Perl common gateway interface module
CGI::Cookie	Interface to Netscape cookies module
DBD::mysql	The perl interface to mysql database
Digest::MD5	The perl interface to the MD5 algorithm

First step is to install perl with the following commands:

```
# tar zxvf stable.tar.gz
# cd perl-5.6.1
# rm –f config.sh Policy.sh
```

```
# sh Configure –de
# make
# make test
# make install
```

After installing the perl package you must install the modules that are needed by Demarc.

```
# tar zxvf DBD-mysql-2.1005.tar.gz
# tar zxvf DBI-1.20.tar.gz
# tar zxvf Digest-MD5-2.16.tar.gz
# cd ../DBI-1.20
# perl Makefile.PL
# make
# make install
# cd ../Digest-MD5-2.16
# perl Makefile.PL
# make
# make install
# cd ../DBD-mysql-2.1005
# perl Makefile.PL
# make
# make install
```

FIFTH STEP – PREPARING THE DATABASE

We are now ready to prepare the DB that Demarc will use to log incoming data from the remote sensors.

We will log on to the mysql database using the user root and then typing the password that we have specified above:

```
# mysql -u root –p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2 to server version: 3.23.45-log
```

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

The next step consists of creating the database for Demarc to use. The database will be called Snort. At the mysql prompt type the command below.

```
mysql> create database Snort;
Query OK, 1 row affected (0.53 sec)

mysql> grant UPDATE,DELETE,INSERT,SELECT on Snort.* to Snort identified by 'b0nhomme';
Query OK, 0 rows affected (0.46 sec)
mysql>
```

We will now ensure that there is no default users left on the system, these default anonymous users can cause problem and should be removed if they are not use by any other users.

```
mysql> use mysql;
Database changed

mysql> DELETE from user WHERE user = '';
Query OK, 2 rows affected (0.48 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.38 sec)

mysql> exit;
Bye
```

Now that the database has been created, it is now time to create the DB Schema. To create the DB schema we will use a script called dm_load_db.pl located under the /usr/local/demarc/install directory.

```
# cd /usr/local/demarc/install
# ./dm_load_db.pl
DB USER? [snort] > root
DB PASSWORD? > (your_password)
DB HOST? [127.0.0.1] > localhost
DB NAME? [snort] > Snort
```

User: root
Password: b0nhomme
Host: localhost
Name: Snort
Is this correct?[y/N] > y

Ensure that there is no message error reported while creating the schema.

SIXTH STEP – CONFIGURING APACHE

At the bottom of /usr/local/www/conf/httpd.conf , you will add the following:

```
Redirect /demarc https://192.168.1.20/dm/demarc
Alias /dm_images "/usr/local/demarc/images"
<Directory "/usr/local/demarc/cgi">
AllowOverride all
DirectoryIndex demarc
</Directory>

PerlModule Apache::Registry
KeepAlive Off
Alias /dm /usr/local/demarc/cgi
SetHandler perl-script
PerlHandler Apache::Registry
Options ExecCGI
allow from all
PerlSendHeader On
DirectoryIndex demarc
```

You must now restart apache to enable the changes above. Use the following commands:

```
# /usr/local/www/bin/apachectl stop
# /usr/local/www/bin/apachectl startssl
```

You can now access Demarc by pointing your browser to :

https://192.168.1.20/demarc

If you encounter any problem, please refer to the Demarc site. There is a good mailing list with feedback provided within hours usually.

3.6 SNORT features of DEMARC

The most important feature of the Demarc tool is the ability to perform correlation of data from multiple SNORT sensors as well as the ability to drill down, search through, and see the details of the events being reported.

Below I will not cover every single configuration screen available in Demarc, this would create a document that would exceed 150 pages by doing so. I will present and explain only the main configuration windows. Every configuration item is done through a graphical interface that is very easy to use. If you wish to have a full view of every single items that Demarc can do, I suggest that you consult the User Guide located at http://www.demarc.com/userguide/ .

3.6.1 Connections (Remote and Local)

The current documentation that comes with Demarc is quite evasive on how things are done internally. I had to contact the people at Demarc to find out how things are working; below you have an explanation from Anthony on how connections are made between the database, the web interface and remote clients. Please also note the suggestions of using Stunnel along with certificates to improve the security of the whole setup, the recipe is in the list of reference at the end of the document.

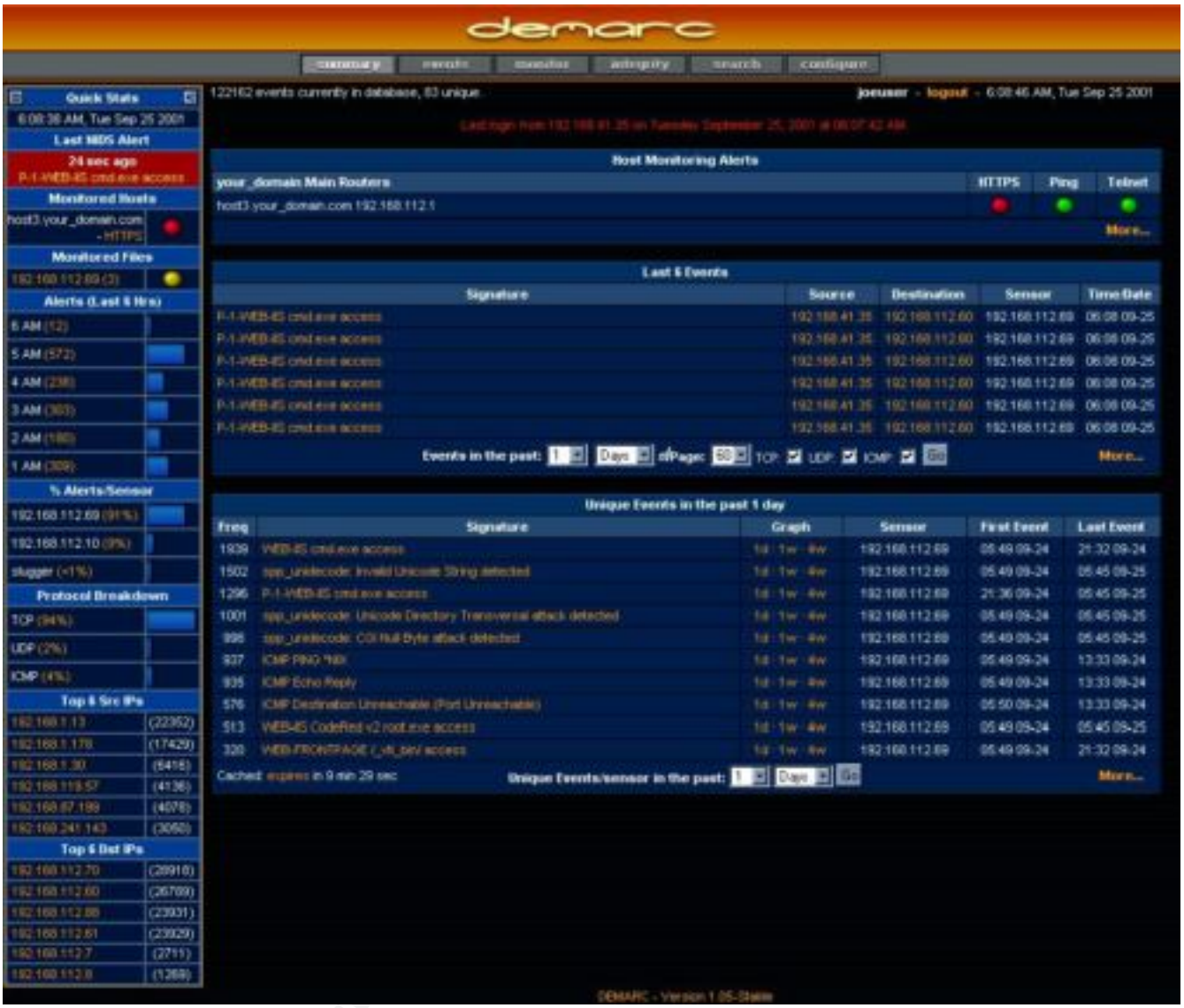
> *The web interface talks to the database via standard db channel (DBI /DBD::mysql perl modules talking on :3306). The clients all talk to the database via the same channel, and request the checks they should perform from the database, and then proceed to check and deposit their results back into the database.*
>
> *This way there is no exploitable daemon listening for connections... We figured that the database daemons have a much better chance of being unexploitable then a home made daemon. If you add stunnel to the mix, then the data, passwords, etc are all sent encrypted, and you don't even have to have a listening database port open on any machines (except binded to localhost lo0 devices of course).*

3.6.2 Demarc Features

Demarc has a whole lot of other features that are very interesting. From the main summary screen you have a composite view of the following (See graphic below):

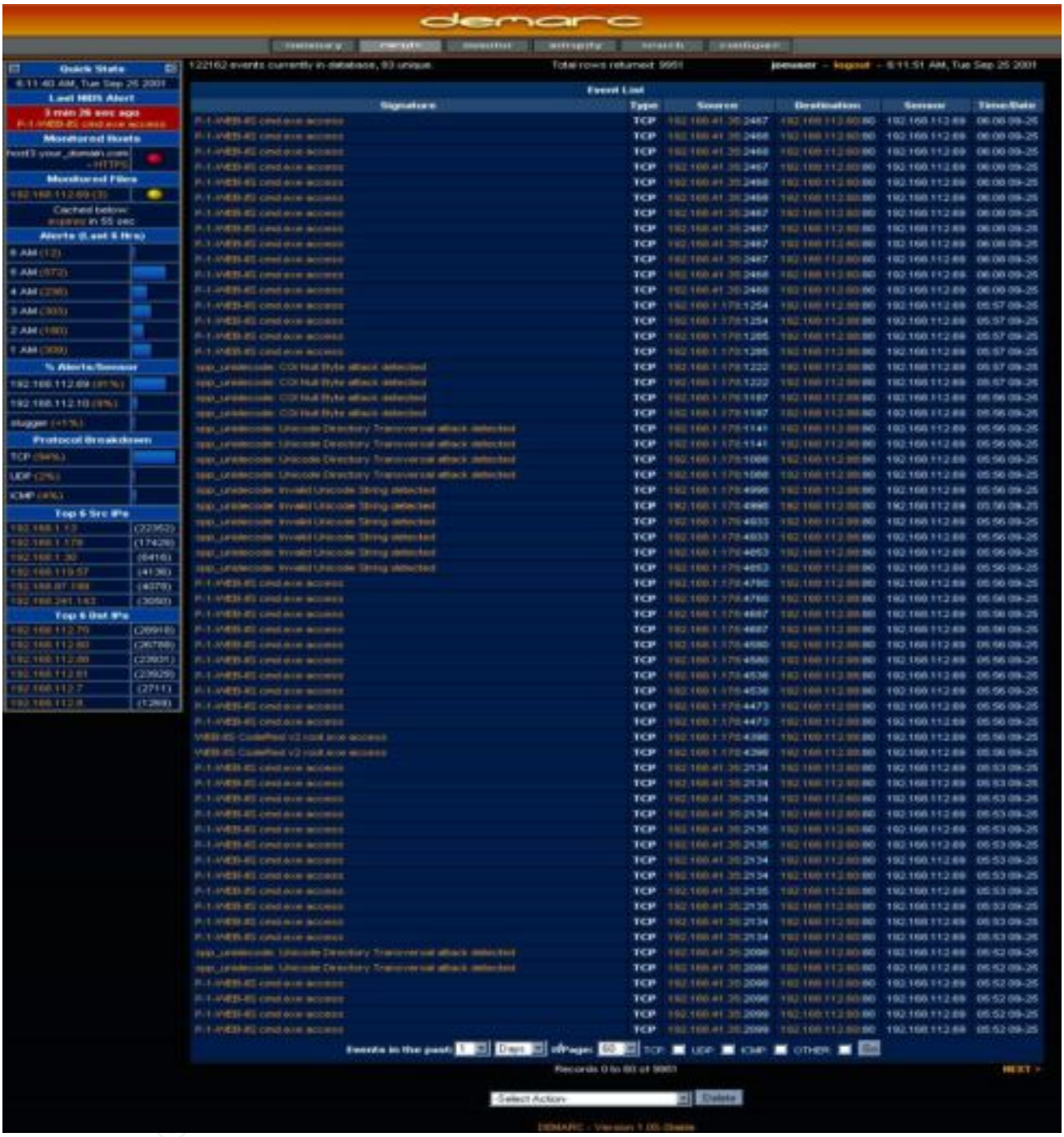
- Timing of last NIDS event;

- Resume of file integrity changes that has been detected;
- Number of alerts per hours over the last six hours;
- Percentage of alerts per specific sensor;
- Protocol percentage breakdown;
- Top six source IP address;
- Top six destination IP address;
- A resume of host monitoring alerts;
- Last 6 events (you can also specify the number of event you wish to see or drill down);
- List of unique events in the past x days. (also configurable to your liking);
- Frequency of specify events;
- Quick drill down on any event or alerts;
- A 'once mouse click' graphing tool for specific type of attack or alert.



3.7 List of events

By a single click of with the mouse on the button marked ‘events’ on the top of the window. It is possible to move from the main screen to the event list screen as depicted below. The event screen has a summary of all events in the db and once again you can drill down by clicking on the event. The event list tells you from which signature it was generated, the type of traffic, the source and destination, from which sensor it came from, and finally the time and date as presented in the graphic below.



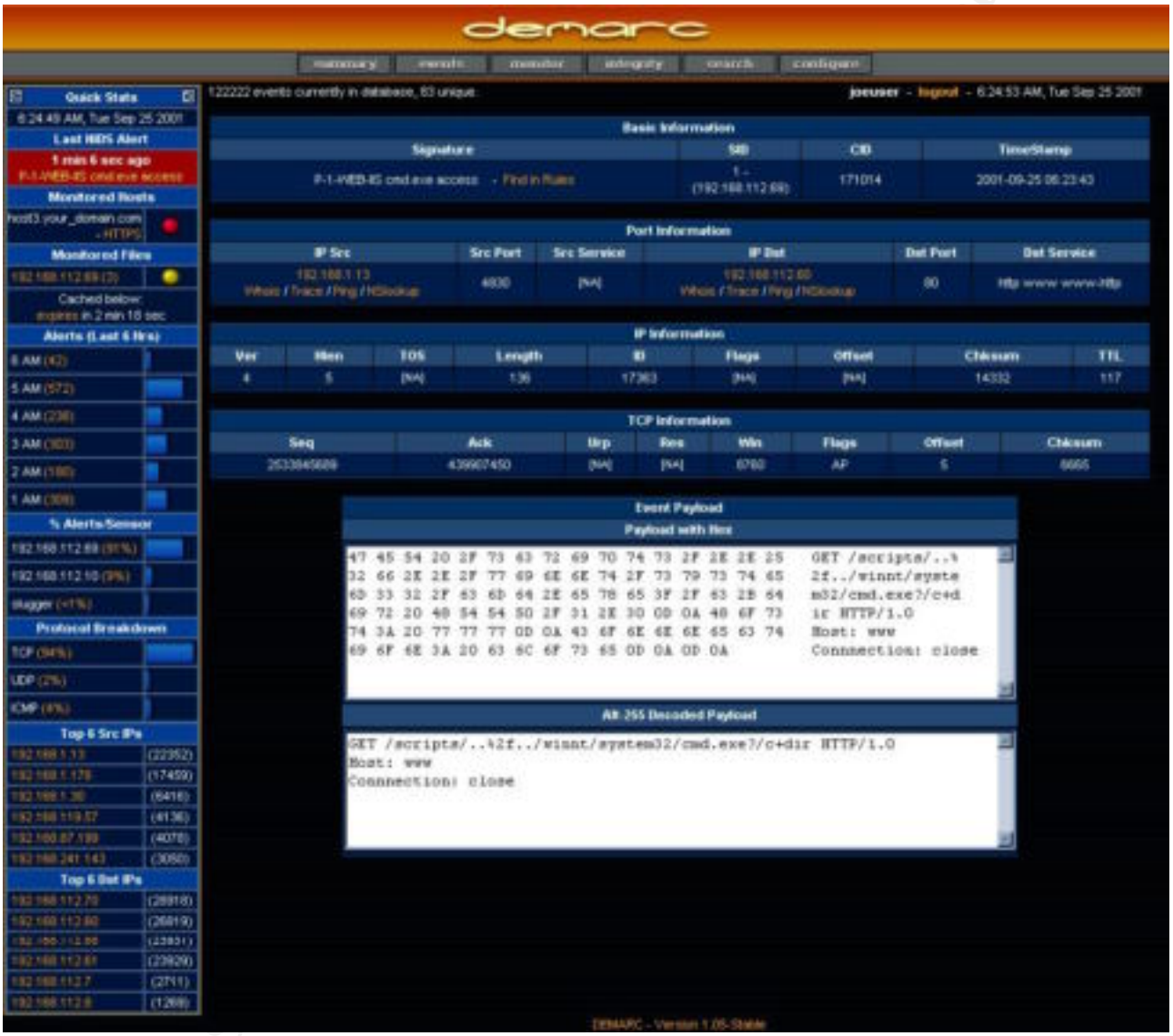
3.8 Event details

By clicking on one of the events presented above, you will get a very detailed view of the event.

The display under the field ‘signature’ tells you on which rule the event was generated, by clicking on find in rules, the rule will be presented.

The packet is also decoded for you. It tells you about flags, length, offset, checksum and other details.

Please take a look at the picture below.



3.9 The availability monitor

As mentioned in the introduction, another very useful feature of Demarc is the ability to monitor processes, services, hosts, or networks. You can configure Demarc to monitor your DNS setup for changes that could indicate a DNS hijacking attack. You can monitor processes, get an alarm when they go down and automatically restart them if desired. You can monitor different

protocols such as ftp, http, SMTP, or custom defines. Another nice feature is the ability to monitor your system load and log files. See picture below for details.

3.10 The integrity monitor

The integrity monitor makes use of MD5 and some characteristics of the files in order to detect unauthorized changes to the files that you are monitoring. It is nearly impossible to modify a file without changing its MD5 Hash value. A MD5 hash is a sort of a cryptographic fingerprint for

files, so even if only one bit has changed in the file, the MD5 hash will no longer match that of the unmodified file. This is the basic concept behind file integrity checks.

By default, the files that you are monitoring are verified every half an hour for any signs of changes. This is a conservative value that can be configured to whatever you wish to use, however you must remember that monitoring a large number of files can be CPU intensive.

As mentioned above, Demarc uses MD5 hash but it will also monitor the following attributes to see if any changes are detected:

- Inode Number
- File Permissions
- User ID of owner
- Group ID of owner
- Size of the file
- Modified timestamp
- Created timestamp

It must be noted that the web integrity checks is a bit different. The following is monitored:

- Size of the webpage downloaded
- MD5 hash of the downloaded webpage

On the integrity screen you can quickly detect if any changes were made to some of your key files. In the example below, you can notice that host 192.168.112.69 has some yellow button that indicates a change to some of the critical files. In this case we can notice that the firewall startup script has been modified, the demarcd file was modified as well, and one of the index.html file which could indicate that one of your web site was compromised or the page was updated on purpose. These alerts will be reported to the main Demarc summary page as well.

Simply by clicking on the 'View Details' link, you will know exactly what was changed on these files. If you know that these changes were authorized changes and you wish to update the

database to ensure that these changes are no longer reported, simply click on the 'Confirm/Update All Changes for All Hosts'.

Below you have a picture of the main integrity monitor screen.

3.11 Conclusion

As you have seen above, the Demarc tool can be of great assistance in maintaining multiple sensors from a central location. The monitoring of multiple facet of your infrastructure could even be delegated to a junior security administrator that can receive the alert and then escalate to a higher level of support if necessary.

This tools addresses most of the CIA triad items. It can greatly improve your overall security while improving your experience dealing with open sources tools such as SNORT.

Every organization with the challenge of monitoring SNORT events and alerts, file integrity, services availability, system load and health, critical system processes, should consider evaluating the Demarc tool.

As mention in the installation portion, it is a bit tricky to get going but once it is up and running, it will make your life so much easier that the time spent on the initial deployment is well worth it.

4 NETWORK DETECTS

Below you will find the analysis of five network detects obtained from the incident.org site at URL http://www.incidents.org/archives/intrusions/date1.html as recommended.

4.1 Network detect 1

4.1.1 Source of trace

http://www.incidents.org/archives/intrusions/msg02778.html

4.1.2 Detect generated by

Snort in fast alert mode

It seems that the rule that triggered this alert is from the Snort WEB-MISC rules distributed at www.snort.org. The Specific rule is listed below, it is a rule that looks from the pattern ../ which indicate a directory traversal attack signature.

alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB-MISC http directory traversal"; flags: A+; content: "../"; reference:arachnids,297; classtype:attempted-recon; sid:1113; rev:1;)

Dec 5 10:09:57 hosthu snort: [1:1113:1] WEB-MISC http directory traversal [Classification: Attempted Information Leak] [Priority:2]: {TCP} 202.4.254.143:2667 -> a.b.c.62:80

Dec 5 10:09:58 hosthu snort: [1:1113:1] WEB-MISC http directory traversal [Classification: Attempted Information Leak] [Priority: 2]: {TCP} 202.4.254.143:2696 -> a.b.c.62:80

Dec 5 10:10:03 hosthu snort: [1:1113:1] WEB-MISC http directory traversal [Classification: Attempted Information Leak] [Priority: 2]: {TCP} 202.4.254.143:2909 -> a.b.c.62:80

Dec 5 10:10:04 hosthu snort: [1:1113:1] WEB-MISC http directory traversal [Classification: Attempted Information Leak] [Priority: 2]: {TCP} 202.4.254.143:2948 -> a.b.c.62:80

Dec 5 10:10:05 hosthu snort: [1:1113:1] WEB-MISC http directory traversal [Classification: Attempted Information Leak] [Priority: 2]: {TCP} 202.4.254.143:2991 -> a.b.c.62:80

Web server log

It looks like the following extract was from a Web Server using the Common Log Format (CLF)

202.4.254.143 - - [05/Dec/2001:10:09:53 -0500] "GET /scripts/root.exe?/c+dir HTTP/1.0" 404 289 "-" "-"

202.4.254.143 - - [05/Dec/2001:10:09:54 -0500] "GET /MSADC/root.exe?/c+dir HTTP/1.0" 404 287 "-" "-"

202.4.254.143 - - [05/Dec/2001:10:09:55 -0500] "GET /c/winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 297 "-" "-

202.4.254.143 - - [05/Dec/2001:10:09:56 -0500] "GET /d/winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 297 "-" "-

202.4.254.143 - - [05/Dec/2001:10:09:57 -0500] "GET /scripts/..%255c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 311 "-" "-"

202.4.254.143 - - [05/Dec/2001:10:09:58 -0500] "GET /_vti_bin/..%255c../..%255c../..%255c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 328 "-" "-"

202.4.254.143 - - [05/Dec/2001:10:09:59 -0500] "GET /_mem_bin/..%255c../..%255c../..%255c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 328 "-" "-"

202.4.254.143 - - [05/Dec/2001:10:10:00 -0500] "GET /msadc/..%255c../..%255c../..%255c/..%c1%1c../..%c1%1c../..%c1%1c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 344 "-" "-"

202.4.254.143 - - [05/Dec/2001:10:10:01 -0500] "GET /scripts/..%c1%1c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 310 "-" "-"

202.4.254.143 - - [05/Dec/2001:10:10:02 -0500] "GET /scripts/..%c0%2f../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 310 "-" "-"

202.4.254.143 - - [05/Dec/2001:10:10:03 -0500] "GET /scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 310 "-" "-"

202.4.254.143 - - [05/Dec/2001:10:10:04 -0500] "GET /scripts/..%c1%9c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 310 "-" "-"

202.4.254.143 - - [05/Dec/2001:10:10:05 -0500] "GET /scripts/..%%35%63../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 400 294 "-" "-"

202.4.254.143 - - [05/Dec/2001:10:10:06 -0500] "GET /scripts/..%%35c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 400 294 "-" "-"

202.4.254.143 - - [05/Dec/2001:10:10:06 -0500] "GET /scripts/..%25%35%63../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 311 "-" "-"

202.4.254.143 - - [05/Dec/2001:10:10:07 -0500] "GET /scripts/..%252f../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 404 311 "-" "-"

4.1.3 Probability the source address was spoofed

The source was probably not spoofed. Based on the attack described below it is unlikely that this type of attack will attempt to spoof the source address. If the address was spoofed then we would not send any reply back to the originator, he would not be able to issue his series of command and it would defeat the goal of the probe. We also notice that all request were made from high ports to low port, which indicates normal traffic as well.

The IP address resolve to a DNS located in Ghana, see whois below.

whois whois.arin.net 202.4.254.143:

Asia Pacific Network Information Center (APNIC2)

These addresses have been further assigned to Asia-Pacific users.

Contact info can be found in the APNIC database,

at WHOIS.APNIC.NET or http://www.apnic.net/

Netname: APNIC-CIDR-BLK

Netblock: 202.0.0.0 - 203.255.255.255

Maintainer: AP

Coordinator:

Administrator, System (SA90-ARIN) [No mailbox]

+61-7-3367-0490

Domain System inverse mapping provided by:

SVC00.APNIC.NET 202.12.28.131

NS.APNIC.NET	203.37.255.97
NS.TELSTRA.NET	203.50.0.137
NS.RIPE.NET	193.0.0.193

4.1.4 Description of the attack

This attack is very likely the 'concept worm' virus or also a high probability of being Nimda or one of its variant attacks. We notice that there is a directory traversal attack being reported by SNORT but this is not the only attack going on after we look closer at the logs provided by the web server. We notice a series of GET statements using different types of encoding which is the typical signature for a Nimda type of attack.

The Nimda signature is very easy to recognized in the web server logs, it will be familiar to the following lines:

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

GET /_mem_bin/..%5c../..%5c../..%5c../winnt/system32/cmd.exe?/c+dir

GET /msadc/..%5c../..%5c../..%5c/..\xc1\x1c../..\xc1\x1c../..\xc1\x1c../winnt/system32/cmd.exe?/c+dir

GET /scripts/..\xc1\x1c../winnt/system32/cmd.exe?/c+dir

GET /scripts/..\xc0/../winnt/system32/cmd.exe?/c+dir

GET /scripts/..\xc0\xaf../winnt/system32/cmd.exe?/c+dir

GET /scripts/..\xc1\x9c../winnt/system32/cmd.exe?/c+dir

GET /scripts/..%35c../winnt/system32/cmd.exe?/c+dir

GET /scripts/..%35c../winnt/system32/cmd.exe?/c+dir

GET /scripts/..%5c../winnt/system32/cmd.exe?/c+dir

GET /scripts/..%2f../winnt/system32/cmd.exe?/c+dir

Note: The first four entries in these sample logs denote attempts to connect to the backdoor left by Code Red II, while the remaining log entries are examples of exploit attempts for the Directory Traversal vulnerability

4.1.5 Attack mechanism

This worm is very smart and introduces the principle of aggregation of vulnerabilities within a single attack. It is even smart enough to use backdoor from other attacks that were performed. It can propagate from client to client via email vulnerabilities, from client to client via open

network shares, from web server to client via browsing of compromised web sites, from client to web server via active scanning for and exploitation of various Microsoft IIS 4.0 / 5.0 directory traversal vulnerabilities, from client to web server via scanning for the back doors left behind by the Code Red II.

This worm was very hard to clean and it was necessary to disable all network access in order to clean the whole network. Below you will find a detailed analysis that was conducted by F-Secure.

Nimda is a complex virus with a mass mailing worm component, which spreads itself in attachments named README.EXE and also under other names. If affects Windows 95, Windows 98, Windows Me, Windows NT 4 and Windows 2000 users.

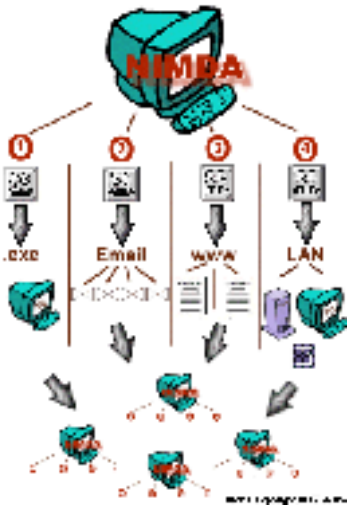
Nimda is the first worm to modify existing web sites to start offering infected files for download. Also it is the first worm to use normal end user machines to scan for vulnerable web sites. This technique enables Nimda to easily reach intranet web sites located behind firewalls - something worms such as Code Red couldn't directly do.

Nimda uses the Unicode exploit to infect IIS web servers. This hole can be closed with a Microsoft patch, downloadable from: http://www.microsoft.com/technet/security/bulletin/ms00-078.asp

The MIME exploit used by the worm can be fixed with this patch: http://www.microsoft.com/technet/security/bulletin/MS01-020.asp

INFECTION PHASES - LIFECYCLE

The actual lifecycle of Nimda can be split to four parts: 1) Infecting files, 2) Mass mailing, 3) Web worm and 4) LAN propagation.



1) File infection

Nimda locates EXE files from the local machine and infects them by putting the file inside its body as a resource, thus 'assimilating' that file. These files then spread the infection when people exchange programs such as games.

2) Mass mailer

Nimda locates e-mail addresses via MAPI from your e-mail client as well as searching local HTML files for additional addresses. Then it sends e-mail to each address. These mails contain an attachment called README.EXE or other name, which might be executed automatically on some systems.

3) Web worm

Nimda starts to scan the Internet, trying to locate www servers. Once a web server is found, the worm tries to infect it by using several known security holes. If this succeeds, the worm will modify random web pages on the site. End result of this modification is that the worm will automatically infect web surfers browsing the site.

4) LAN propagation

The worm will search for file shares in the local network, either from file servers or from end user machines. Once found, it will drop a hidden file called RICHED20.DLL to any directory, which has DOC and EML files. When other users try to open DOC or EML files from these directories, Word, WordPad or Outlook will execute RICHED20.DLL causing an infection of the PC. The worm will also infect remote files if it was started on a server.

TECHNICAL DETAILS

First it should be noted that the worm behaves differently when started from files with different file names and with different command lines.

Starting on a server:

If the name of worm's file is ADMIN.DLL, the worm creates a mutex with 'fsdhqherwqi2001' name, copies itself as MMC.EXE into \Windows\ directory and starts this file with '-qusery9bnow' command line. Usually the worm is started as ADMIN.DLL on infected webservers. In this case the worm starts to scan and infect files on all available drives including removable and network ones. The EXE files (except WINZIP32.EXE) on these drives will get infected with the worm. The infection technique the worm uses is new - the worm puts an infected file inside its body as a resource. When the infected file is run, the worm extracts the embedded original EXE file, runs it and tries to delete it afterwards. If instant deletion is not possible, the worm creates WININIT.INI file that will delete the extracted file on next Windows startup.

The worm also accesses [SOFTWARE\Microsoft\Windows\CurrentVersion\App Paths] key reads subkeys from there and infects all files listed in the subkeys. The worm doesn't infect WinZip32.exe file. Also the worm reads user's personal folders from [Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders] key and infects files in these folders as well.

Then the worm starts to search local hard drives for *.HTML, .ASP, and .HTM files and if such files are found, the worm creates README.EML file (which is the multi-partite message with MIME-encoded worm) in the same directory and adds a small JavaScript code to the end of found files. That JavaScript code would open README.EML file when a web browser loads the infected HTML file. As a result the MIME-encoded worm will get activated because of a security hole and a system will get infected.

The worm's file runs from a minimized window when downloaded from an infected web server. This technique affects users who are browsing the web with Internet Explorer 5.0 or 5.01.

The worm will also put *.EML and *.NWS files in almost all folders of computers it accesses. The RICHED20.DLL file with hidden and system attribute will be put in all folders where DOC or EML files are located. The worm will also try to replace Windows' original RICHED20.DLL file with its own copy.

Starting on a workstation:

If the worm is started from README.EXE file (or a file that has more than 5 symbols in its name and EXE extension), it copies itself to temporary folder with a random name that has 'MEP*.TMP' name and runs itself there with '-dontrunold' command line option.

When started, the worm loads itself as a DLL library, looks for a specific resource there and checks its size. If the resource size is less than 100, the worm unloads itself, otherwise it extracts its resource to a file and launches it. Checking the resource size is done to be able to detect if a worm runs from infected EXE files.

Then the worm gets current time and generates a random number. After performing a few arithmetic operations with this number the worm checks the result. If a result is bigger than worm's counter, the worm starts to search and delete README*.EXE files from temporary folder.

After that the worm prepares its MIME-encoded copy by extrating a pre-defined multi-partite MIME message from its body and appending its MIME-encoded copy to it. The file with a random name is created in a temporary folder.

The worm then looks for EXPLORER process, opens it and assigns its process as remote thread of Explorer. On some platforms the worm fails to run as Explorer's thread. The worm gets API creates a mutex with 'fsdhqherwqi2001' name, startups Winsock services, gets an infected computer (host) info and sleeps for some time. When resumed, the worm checks what platform it is running. If it is running on NT-based system, it compacts its memory blocks to occupy less space in memory and copies itself as LOAD.EXE to Windows system directory. Then it modifies SYSTEM.INI file by adding the following string after SHELL= variable in [Boot] section:

```
explorer.exe load.exe -dontrunold
```

This will start the worm's copy every time Windows starts. The worm also copies itself as RICHED20.DLL file to system folder and sets hidden and system attributes to this file as well as to LOAD.EXE file. Then the worm enumerates shared network resources and starts to recursively scan files on remote systems.

When searching for files on remote systems the worm looks for .DOC and .EML files and then copies its binary image with RICHED20.DLL name to the folders where DOC and EML files are located. The copied DLL file has system and hidden attributes. This is done to increase the

chances of worm activation on remote systems as Windows' original RICHED20.DLL component is used to open OLE files. But instead the worm's RICHED20.DLL file from current directory will be launched.

Also when the worm browsing the remote computers' directories it creates .EML and .NWS (rarely) files that have the names of document or webpage files that the worm could find on a remote system. These .EML and .NWS files are worm's multi-partite messages with a worm MIME-encoded in them. When scanning the worm can also delete the .EML and .NWS files it previously created.

The worm doesn't try to infect local or remote EXE files when started from a workstation.

E-Mail spreading:
The worm searches trough all the '.htm' and '.html' file in the Temporary Internet Files folder for e-mail addresses. It reads trough user's inbox and collects the sender addresses. When the address list is ready it uses it's own SMTP engine to send the infected messages.

IIS spreading:
The worm uses backdoors on IIS servers such as the one CodeRed II installs. It scans random IP addresses for these backdoors. When a host is found to have one the worm instructs the machine to download the worm code (Admin.dll) from the host used for scanning. After this it executes the worm on the target machine this way infecting it.

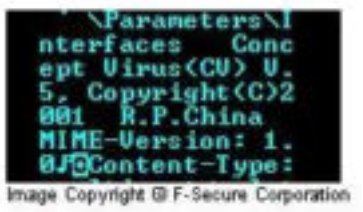
Affecting the security:
The worm adjusts the properties of Windows Explorer, it accesses [Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced] key and adjusts 'Hidden', 'ShowSuperHidden' and 'HideFileExt' keys. This affects Windows' (especially ME and 2000) ability to show hidden files - worm's files will not be seen in Explorer any more.

After that the worm adds a 'guest' account to infected system account list, activates this account, adds it to 'Administrator' and 'Guests' groups and shares C:\ drive with full access priviledges. The worm also deletes all subkeys from [SYSTEM\CurrentControlSet\Services\lanmanserver\Shares\Security] key to disable sharing security.

Additional information:

The worm has a copyright text string that is never displayed:

```
Concept Virus(CV) V.5, Copyright(C)2001  R.P.China
```



4.1.6 Correlations

This attack, some of it's variants, or part of this attack has been reported and well documented by the following organizations as well:

Security Space	**http://www.securityspace.com/smysecure/w32_nmda_amm.html**
CERT	**http://www.cert.org/advisories/CA-2001-26.html**
CERT	**http://www.kb.cert.org/vuls/id/111677**
CERT	**http://www.cert.org/advisories/CA-2001-12.html**
CERT	**http://www.cert.org/incident_notes/IN-2001-09.html**
CERT	**http://www.cert.org/advisories/CA-2001-11.html**
Sarang.net	**http://kang.sarang.net/snort/203/248/27/src203.248.27.38.html**

4.1.7 Evidence of active targeting

This attack is not the result of an individual person sitting at the keyboard and launching a series of commands. If we look at the speed it was performed in the web log, I do not believe that it was manually done. This is most likely a script being run and searching for other targets. So it does not look, as an attack against a specific target for a specific service.

4.1.8 Severity

The severity of this attack is: 3

Criticality	Lethality	System Countermeasures	Network Countermeasures	Severity
4	5	3	3	**3**

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

Scale: 0 = low and 5=high

4.1.9 Substantiation

This attack is already quite old, but still very active. By now most systems SHOULD have been properly patched. Most web servers are playing an important role in either the distribution of corporate information or conduct of commercial activities over the web. This attack is an aggregation of attack and one of the first of this type. It is a very lethal attack, which contributes to the lethality level of 5. Web server are now days used for information distribution and in many case for online business activity, disturbance of these activities could seriously damage the image of the company and even worse you could be portrait as someone attacking other sites. With the number of advisories and patches that followed this attack, it is assumed that a competent administrator would have patched his system already. But considering that the attack made it to the server and was detected by the IDS system, all systems may not have been patched. I do not have enough information to correctly estimate the level of system and network countermeasures in place; this is why I went with a middle of the road rating of 3.

4.1.10 Defensive recommendation

The first defensive step that should be conducted by any conscientious administrator is to ensure that his systems are patched and that they remain patched by following closely his vendor advisories. Other means of minimizing this would be by having outbound firewall rules that would detect unexpected traffic such as your web server surfing other sites all of a sudden. This attack also highlighted that users machines must be maintained as well as they can contribute to the aggravation of the problem by infecting hosts that have been cleaned.

4.1.11 Multiple choice test question based on the trace and analysis above

Question: Why is it important to have granular outbound access rules?

Answer 1: For accounting purposes

Answer 2: To detect which traffic must be encrypted

Answer 3: To detect anomalies in traffic pattern.

Answer 4: It is not important

Correct answer: 3

Details: It is important to have restrictive outbound rules as well as inbound rules. Such rules will often allow you to detect unexpected connection attempts that may indicate unauthorized or unusual activity on your network. For example, outbound telnet or SSH connection from one of your service network servers.

4.1.12 Supplementary comments

This attack was a rude awakening call for a lot of companies. Having an organized reaction plan suddenly became a priority. It was necessary to completely turn off all networking components to prevent further infection.

4.2 Network Detect 2

4.2.1 Source of trace

http://www.incidents.org/archives/intrusions/msg02778.html

4.2.2 Detect generated by

The extracts seems to be from Syslog. It is possible that this computer has TCPWrappers installed.

Dec 5 14:43:30 hostre rpcbind: refused connect from 61.178.15.138 to dump()

Dec 5 14:44:33 hostbe rpcbind: refused connect from 61.178.15.138 to dump()

Dec 5 14:50:51 hostmau portmap[20318]: connect from 61.178.15.138 to dump(): request from unauthorized host

These extracts seems to be from SNORT. It seems that SNORT is logging to Syslog in this case. The extract is generated by a few rules that are distributed in the www.snort.org rules. Below you have the SNORT rule that has triggered the output for each of the lines:

alert tcp $EXTERNAL_NET any -> $HOME_NET 111 (msg:"RPC portmap listing"; flags:A+; content: "|00 01 86 A0 00 00 00 02 00 00 00 04|"; reference:arachnids,429; classtype:rpc-portmap-decode; sid:598; rev:3;)

alert tcp $EXTERNAL_NET any -> $HOME_NET 111 (msg:"RPC portmap request rstatd"; content: "|01 86 A1 00 00|"; reference:arachnids,10; classtype:rpc-portmap-decode; flags:A+; sid:1270; rev:3;)

Dec 5 14:50:51 hostmau snort: [1:598:1] RPC portmap listing [Classification: Attempted Information Leak] [Priority: 3]: {TCP} 61.178.15.138:842 -> z.y.w.12:111
Dec 5 15:23:33 hosty snort: [ID 702911 local0.alert] [1:1270:2] RPC portmap request rstatd [Classification: Decode of an RPC Query] [Priority: 2]: {TCP} 61.178.15.138:672 -> z.y.x.34:111
Dec 5 15:23:35 hostj snort: RPC portmap listing [Classification: Attempted Information Leak Priority: 3]: 61.178.15.138:694 -> z.y.x.66:111
Dec 5 15:23:36 hostmi snort: [ID 702911 auth.alert] [1:1270:2] RPC portmap request rstatd [Classification: Decode of an RPC Query] [Priority: 2]: {TCP} 61.178.15.138:710 -> z.y.x.98:111

4.2.3 Probability the source address was spoofed

The source address is probably not spoofed. The attacker is doing a reconnaissance attempt in order to gather information from RPC services. This Remote Procedure Call (RPC) dump request was performed using TCP, which needed a three-way handshake in the first place. The portmapper dump() that is seen in the logs are when someone is trying to get a list of all running registered daemons. Such a probe can be easily done from the UNIX command line or from scanning tools such as Netscantools. This attacker was probably trying to find out if any RPC services are running by doing a RPC Portmap listing or by issuing the rpcinfo command to see if there is an NFS server running with exportable filesystems. If services such as NFS is running, and there are exportable filesystems, the attacker could remotely attempt to exploit such services. If the address were spoofed this probe would be meaningless because the result would not be returned to the attacker. It must also be noted that there is vulnerable RPC services that can be exploited to gain root access as well. The CVE database is listing quite a few of them.

The IP address is a valid IP; there is no hostname or reserve lookup configured for it.

Here is the whois result for the offending address:

whois whois.arin.net 61.178.15.138:
Asia Pacific Network Information Center (NETBLK-APNIC2)
These addresses have been further assigned to Asia-Pacific users.
Contact info can be found in the APNIC database,

at WHOIS.APNIC.NET or http://www.apnic.net/
Please do not send spam complaints to APNIC.

Netname: APNIC3
Netblock: 61.0.0.0 - 61.255.255.255
Maintainer: AP
Coordinator:
Administrator, System (SA90-ARIN) [No mailbox]
+61-7-3367-0490

Domain System inverse mapping provided by:

NS.APNIC.NET	203.37.255.97
NS.RIPE.NET	193.0.0.193
RS1.ARIN.NET	192.149.252.21

Regional Internet Registry for the Asia-Pacific Region.

4.2.4 Description of the attack

In the logs we notice that SNORT is reporting a RPC Portmap listing, this event indicates that a query was sent to the Portmapper daemon, requesting port information for RPC services (see evidence of targeting section below for more details). We also notice that all of the requests are directed to port 111, which is the RPC port. This is most likely a series of alerts that were triggered when an RPC dump request was issued from the remote host to three local hosts. CERT regularly receives report of exploitations involving three RPC vulnerabilities: rpc.cmsd, ttdbserverd, and statd/automountd. These exploitations can lead to root compromise on systems that implement vulnerable RPC services.

4.2.5 Attack mechanism

Attacks performed using the RPC services are numerous. Some of the attack allows the gathering of information on some of the RPC services and in some case vulnerabilities are identified, which can then be used to gain administrator access through these vulnerable services or it may also be a case of misconfiguration of services such as NFS where unauthorized users may access remote filesystems.

A good example of a vulnerability associated with NFS would be the Linux nfsd Remote Buffer Overflow Vulnerability. In this case the length of the string holding the directory name which was to be removed was not checked and the buffer holding it could be overflowed, allowing execution of arbitrary code on the NFS server as root. A consequence of this being exploited is remote root compromise. As you can see, this is a very serious vulnerability.

RPC is associated to a lot of programs/daemons, see partial list below.

Portmapper	100000	portmap	sunrpc	
rstatd	100001	rstat	rup	perfmeter
rusersd	100002	rusers		
nfs	100003	nfsprog		
ypserv	100004	ypprog		
mountd	100005	mount	showmount	
ypbind	100007			
walld	100008	rwall	shutdown	
yppasswdd	100009	yppasswd		
etherstatd	100010	etherstat		
rquotad	100011	rquotaprog	quota	rquota

4.2.6 Correlations

The following addresses have reported information about these type of vulnerabilities or reported having seen probing activities related to RPC vulnerabilities.

Security Focus,
http://www.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=782
Sans GIAC, http://www.sans.org/y2k/020500-1215.htm
Sans GIAC, http://www.sans.org/y2k/010400-1115.htm
Cisco Security DB http://www.opensystems.com/support/docs/6332/expsig_6113.html
Cisco Security DB http://www.opensystems.com/support/docs/6332/expsig_6112.html
NIPC Warning http://www.nipc.gov/warnings/alerts/2001/01-010.htm

4.2.7 Evidence of active targeting

This seems like active targeting being accomplished by a human being. We see that there is gap between each of the command and that it is not on a wide scale. The person comes and finds that there is some RPC services running, he then comes back about 10 minutes later with a few more probes. The only strange part in the packet is the fact that it is from a low port number to a low port number. This can happen considering that RPC servers do not use reserved ports (as opposed to those services listed in the */etc/services* file); when they start they just use any port that they can find available. When a client program wants to request a particular RPC service, it has no way of knowing on which port the program number it wants to access is provided. This is where the portmapper daemon solves this problem. It maps RPC program numbers to the TCP/IP ports on which their servers are listening. When an RPC server starts, it picks an available port, and then registers that port and what RPC program numbers it will serve with the

portmapper daemon. When a client program needs to access a service, it first queries the portmapper on the RPC server's host which reports the TCP and UDP port on which the server is listening, and then it contacts that port to request its service.

4.2.8 Severity

The severity of this attack is: 1

Criticality	Lethality	System Countermeasures	Network Countermeasures	Severity
4	5	4	4	1

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

Scale: 0 = low and 5=high

4.2.9 SUBSTANTIATION

The RPC based attacks are common against environment that uses UNIX based computers and more specifically NFS based file sharing. As mentioned in by SANS and the FBI in the top twenty threats, all form of file sharing such as NFS have vulnerabilities associated with them. The criticality of the attack could be serious for environment using such tools, often we have databases that have a shared storage area running under NFS. The lethality if also quite high, in most case the attacks are in the form of buffer overflow and could give root access. Our system and network are maintained in a proper state, it seems that TCPWrappers are installed to help in securing the insecure RPC services. The fact that this probe was detected and reported is also a good sign that the network is well protected.

4.2.10 Defensive recommendation

The first step is to disable any RPC services if they are not being used. If they are needed in your environment, I would suggest that you follow the following steps from the SANS top 20 vulnerabilities.

- Wherever possible, turn off and/or remove these services on machines directly accessible from the Internet.

- Where you must run them, install the latest patches
- Block the RPC port (port 111) at the border router or firewall.
- Block the RPC "loopback" ports, 32770-32789 (TCP and UDP)

4.2.11 Multiple choice test question based on the trace and analysis above

Question: What does the abbreviation RPC means?

Answer 1: Remote Processor Cycles

Answer 2: Remote Procedure Call

Answer 3: Remote Processor Call

Answer 4: Remote Procedure Cycles

Correct answer: 2

Details: RPC (Remote Procedure Call) is a mechanism for client-server applications, which is in the public domain and used by most flavors of Unix. Applications such as NFS, the Network File System, and NIS, the Network Information System are based on RPC.

An RPC server provides a group of procedures, which a client can call by sending a request to the server. The server then invokes the procedure on behalf of the client, returning a value as necessary. For example, you could request the list of available file systems under NFS.

4.2.12 Supplementary comments

All forms of file sharing are mentioned in the SANS/FBI top 20 risks. Unless absolutely necessary it should be disabled. If you must use such services, then refer to the SANS recommendations on how to secure your RPC based services. Please consult: http://www.sans.org/top20.htm

4.3 Network Detect 3

4.3.1 Source of trace

http://www.incidents.org/archives/intrusions/msg02778.html

4.3.2 Detect generated by

The logs provided seems to be from Syslog, some of the entries were produced by the TCP Wrappers alerts and others were from Snort. Some matches were from the following Snort rule:

alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SCAN SYN FIN";flags:SF; reference:arachnids,198; classtype:attempted-recon; sid:624; rev:1;)

Dec 5 19:42:03 hostbe sshd1[2503]: refused connect from 202.105.52.139

Dec 5 19:42:03 hostbe sshd1[2504]: refused connect from 202.105.52.139

Dec 5 19:42:03 hostre sshd1[27411]: refused connect from 202.105.52.139

Dec 5 19:42:03 hostre sshd1[27412]: refused connect from 202.105.52.139

Dec 5 19:44:07 hostci sshd[7779]: [ID 947420 auth.warning] refused connect from root@202.105.52.139

Dec 5 19:44:07 hostci sshd[7780]: [ID 947420 auth.warning] refused connect from root@202.105.52.139

Dec 5 19:44:07 hostdr sshd[14537]: [ID 947420 auth.warning] refused connect from root@202.105.52.139

Dec 5 19:44:07 hostdr sshd[14538]: [ID 947420 auth.warning] refused connect from root@202.105.52.139

Dec 5 19:44:07 hostl sshd[23550]: [ID 947420 auth.warning] refused connect from root@202.105.52.139

Dec 5 19:44:07 hostl sshd[23551]: [ID 947420 auth.warning] refused connect from root@202.105.52.139

Dec 5 19:44:07 hostst sshd[12911]: [ID 947420 auth.warning] refused connect from root@202.105.52.139

Dec 5 19:44:07 hostst sshd[12912]: [ID 947420 auth.warning] refused connect from root@202.105.52.139

Dec 5 19:44:08 hostro sshd1[1215]: connect from 202.105.52.139

Dec 5 19:44:08 hostro sshd1[1216]: connect from 202.105.52.139

Dec 5 19:44:42 202.105.52.139:22 -> z.y.w.12:22 SYNFIN ******SF

Dec 5 19:44:42 202.105.52.139:4843 -> z.y.w.12:22 SYN ******S*

Dec 5 19:44:42 hostmau snort: [1:624:1] SCAN SYN FIN [Classification: Attempted Information Leak] [Priority: 3]: {TCP} 202.105.52.139:22 -> z.y.w.12:22

Dec 5 19:44:43 hostmau sshd[20690]: refused connect from 202.105.52.139 (202.105.52.139)

Dec 5 19:44:43 hostmau sshd[20691]: refused connect from 202.105.52.139 (202.105.52.139)

Dec 5 19:44:59 hostca sshd[23810]: Denied connection from 202.105.52.139 by tcp wrappers.

Dec 5 19:44:59 hostca sshd[23811]: Denied connection from 202.105.52.139 by tcp wrappers.

Dec 5 19:44:59 hostca sshd[23812]: Denied connection from 202.105.52.139 by tcp wrappers.

Dec 5 19:44:59 hostca sshd[23813]: Denied connection from 202.105.52.139 by tcp wrappers.

Dec 5 19:44:59 hostca sshd[23814]: Denied connection from 202.105.52.139 by tcp wrappers.

Dec 5 19:44:59 hostca sshd[23815]: Denied connection from 202.105.52.139 by tcp wrappers.

Dec 5 19:44:59 hostca sshd[23816]: Denied connection from 202.105.52.139 by tcp wrappers.

Dec 5 19:44:59 hostca sshd[23817]: Denied connection from 202.105.52.139 by tcp wrappers.

Dec 5 19:45:40 hoste sshd[98569]: twist 202.105.52.139 to /bin/echo "You are not welcome to use sshd from 202.105.52.139."

Dec 5 19:45:40 hoste sshd[98570]: twist 202.105.52.139 to /bin/echo "You are not welcome to use sshd from 202.105.52.139."

Dec 5 19:50:43 202.105.52.139:22 -> z.y.x.34:22 SYNFIN ******SF

Dec 5 19:50:43 hosty snort: [ID 702911 local0.alert] [111:13:1] spp_stream4: STEALTH ACTIVITY (SYN FIN scan) detection {TCP} 202.105.52.139:22 -> z.y.x.34:22

Dec 5 19:50:44 202.105.52.139:2928 -> z.y.x.34:22 SYN ******S*

Dec 5 19:50:44 hostj snort: SCAN SYN FIN [Classification: Attempted Information Leak Priority: 3]: 202.105.52.139:22 -> z.y.x.66:22

Dec 5 19:50:44 hostmi snort: [ID 702911 auth.alert] [111:13:1] spp_stream4: STEALTH ACTIVITY (SYN FIN scan) detection {TCP} 202.105.52.139:22 -> z.y.x.98:22

Dec 5 19:50:45 hostj sshd1[20445]: refused connect from 202.105.52.139

Dec 5 19:50:45 hostj sshd1[20446]: refused connect from 202.105.52.139

Dec 5 19:50:45 hosty sshd[24372]: [ID 947420 auth.warning] refused connect from root@202.105.52.139

Dec 5 19:50:45 hosty sshd[24373]: [ID 947420 auth.warning] refused connect from root@202.105.52.139

Dec 5 19:50:47 hostmi sshd[17473]: [ID 947420 auth.warning] refused connect from root@202.105.52.139

Dec 5 19:50:47 hostmi sshd[17474]: [ID 947420 auth.warning] refused connect from root@202.105.52.139

4.3.3 Probability the source address was spoofed

The source was probably not spoofed. This looks like a reconnaissance attacks against multiple of our hosts. This is using TCP and a three-way handshake must be established. It seems that the IP address in not responding and once again it belong to an assignment in ASIA-Pacific. I also love these administrator entries with 'No mailbox'.

whois whois.arin.net 202.105.52.139:
Asia Pacific Network Information Center (APNIC2)
These addresses have been further assigned to Asia-Pacific users.
Contact info can be found in the APNIC database,
at WHOIS.APNIC.NET or http://www.apnic.net/
Please do not send spam complaints to APNIC.

Netname: APNIC-CIDR-BLK

Netblock: 202.0.0.0 - 203.255.255.255

Maintainer: AP

Coordinator:
Administrator, System (SA90-ARIN) [No mailbox]
+61-7-3367-0490
Domain System inverse mapping provided by:
SVC00.APNIC.NET 202.12.28.131
NS.APNIC.NET 203.37.255.97
NS.TELSTRA.NET 203.50.0.137
NS.RIPE.NET 193.0.0.193
Regional Internet Registry for the Asia-Pacific Region.

4.3.4 Description of the attack

The attack took place over an 8 minutes time frame. This attacker was targeting only port 22 which is the SSH port. It seems that the attacker is trying to find out if SSH is running on any of the hosts. He was trying to find any version of SSH. The attacker used SYN FIN scans as well probably to identify our OS in order to be able to see if the specific platform is vulnerable to any of the known SSH attacks. However in real life as you will see in correlation below it could also be that sshd is misconfigured or librairies are not adequate and valid connection attempts are being done. In this case with the amount of evidence and the numerous stealth scan, we can safely and easily deduct it was malicious. It must be noted that some of the flag are absolutely abnormal flags and should not be seen unless it is crafted packets. With the amount of information presented I cannot tell for sure what vulnerability or specific exploit the attacker was probing for.

4.3.5 Attack mechanism

There are currently multiple vulnerabilities that have been released against SSH. Some of them are very theorical and only a very specific setup would allow exploitation but other are quite simple and exploits have been produced already.

SSH version 3.0 for UNIX had a potential remote root exploit that was discovered. This root exploit was concerning accounts with password field entries of two characters or less. Unauthorized users could potentially log in to these accounts using any password, including an empty password. This is due to a problem with password authentication to the sshd2 (described below).

It must also be noted that if you use a form of authentication other than password, AND password authentication is disabled, you are **NOT VULNERABLE** to this potential flaw.

This seems like it is not very deadly but please take a minute to read this paragraph carefully. Under UNIX the encrypted password is stored in /etc/shadow or /etc/password. This SSH vulnerability is a very serious problem under the Solaris operating system because the password field uses NP to indicate locked administrative accounts such as 'lp', 'adm', 'bin' etc. All of these administrative accounts can be accessed without a password if a vulnerable SSH daemon is installed on the machine. Some Linux machines have !! in that field and they may be vulnerable as well. The !! is used by xfs and gdm.

The technical problem is described as follow by ssh.com:

The bug is in the code that compares the result of calling crypt(pw, salt) with the value of the encrypted password in the /etc/shadow (or /etc/password) file. SSH Secure Shell Server 3.0.0 or the 3.0.0 daemon does a bounded string compare bounded to the length of the value stored in aforementioned file (2 characters in this case) against the return value of crypt(). The return value of crypt() is 13 characters, with the first two characters being the salt value itself. The salt value used is the first two characters of the encrypted password in /etc/shadow (or /etc/password). A 2 character string comparison between the 2 character encrypted password in /etc/shadow, and the 13 character crypt() return value, whose first two characters ARE the 2 characters from the password in /etc/shadow. The strings match, and the 3.0.0 daemon then accepts the password, no matter what is input.

4.3.6 Correlations

SecurityFocus Incident Mailing List
http://www.securityfocus.com/cgi-bin/archive.pl?id=75&start=2001-12-29&end=2002-01-04&threads=0&mid=Pine.GSO.3.96.1011209120600.16706E-100000@crypto
SecurityFocus Incident Mailing List, All messages dealing with SSHD scanning
http://www.securityfocus.com/cgi-bin/archive.pl?id=75&start=2001-12-29&end=2002-01-04&threads=1&tid=244571
Refused SSH Connect http://www.freebsddiary.org/ssh_refused.php
www.incident.org http://www.incidents.org/archives/intrusions/msg01710.html
www.incident.org http://www.incidents.org/archives/intrusions/msg02719.html
Abnormal TCP Flag
http://www.whitehats.ca/main/members/Seeker/seeker_tcp_header/seeker_tcp_header.htm

4.3.7 Evidence of active targeting

It seems that this was a reconnaissance only. I do not believe that a specific host was targeted as we see that the responses are from multiple hosts. It seems that some type of script or tool was used if we look at the number of probes within 1 second and the SIN-FIN scans probably for the purpose of fingerprinting.

4.3.8 Severity

The severity of this attack is: 2

Criticality	Lethality	System Countermeasures	Network Countermeasures	Severity
5	5	5	3	**2**

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

Scale: 0 = low and 5=high

4.3.9 Substantiation

In this attack we had some hosts that were targeted for over 8 minutes. Servers that are protected by SSH and do not allow other insecure form of access such as telnet are usually important servers such as firewalls. SSH is definitively a critical service as administrator use it to perform remote administration when there is problem with a host. They are usually a critical part of the business as such I have assigned a score of 5. As explained above, some of the SSH vulnerabilities are very severe and could lead to a complete system compromise, I have assigned 5 for criticality. In this case I am giving a high rating to this admin for system countermeasures because he was wise enough to install TCP Wrappers and to filter access based on IP addresses. However I gave a lower rating for the network countermeasures because there is no need for such traffic to be allowed through if it is not needed from external address. It could have been blocked directly at the firewall or the router.

4.3.10 Defensive recommendation

I would recommend the following steps to improve SSH deployment:

Ensure that you have a version of SSH that is not vulnerable by verifying with your vendor as well as verifying with leading advisory sites.

If there is no need to have access from external source, then ensure that this service is not allowed through your screening device.

Use a form of authentication other than password. SSH supports PKI and hardware token as well.

4.3.11 Multiple choice test question based on the trace and analysis above

Question: What is the main advantage of using TCP Wrappers?

Answer 1: It allows faster SSH connections.

Answer 2: It allows more granular access control.

Answer 3: It allows automatic rotation of log files.

Answer 4: It is a nice way of securing traffic in transit.

Correct answer: 2

Details: As mentioned by Stacy M. Arruda, in his paper at http://www.sans.org/infosecFAQ/unix/TCP_wrappers2.htm

TCP Wrappers acts much like a soldier at a checkpoint, verifying a host's clearance prior to entry. Simply put TCP Wrappers capitalizes on the client/server relationship necessary for most TCP/IP applications. TCP Wrappers inserts itself into the middle of the relationship and acts as the server until the client/host is authenticated. TCP Wrappers utilizes its access control feature to authenticate hosts. TCP Wrappers does all of this with no overhead to the system and best of all it is free. (TCP Wrappers is available at ftp.cert.org/pub/tools/tcp_wrappers_7.6.tar.gz and ftp.porcupine.org/pub/security/).

4.3.12 Supplementary comments

TCP Wrappers does more than strictly providing better access control and logging. I invite you to read Mr. Arruda paper mentioned above. It is a nice resume of the TCP Wrappers features.

4.4 Network Detect 4

4.4.1 Source of trace

http://www.incidents.org/archives/intrusions/msg02755.html

4.4.2 Detect generated by

The logs provided seems to be from Syslog, some of the entries were produced by Snort and other by BIND itself. Some matches were from the following Snort rule:

alert udp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"DNS named version attempt"; content:"|07|version"; offset:12; content:"|04|bind"; nocase; offset: 12; reference:arachnids,278; classtype:attempted-recon; sid:257; rev:1;)

Dec 3 13:05:28 hosty named[6612]: [ID 295310 daemon.notice] security: notice: denied query from [211.251.146.129].3236 for "version.bind" CHAOS

Dec 3 13:05:28 hosty snort: [ID 702911 auth.alert] [1:257:1] DNS named version attempt [Classification: Attempted Information Leak] [Priority: 3]: {UDP} 211.251.146.129:3236 -> z.y.x.34:53

Dec 3 13:05:32 hostj named[506]: security: notice: denied query from [211.251.146.129].3236 for "version.bind"

Dec 3 13:05:32 hostj snort: DNS named version attempt [Classification: Attempted Information Leak Priority: 3]: 211.251.146.129:3236 -> z.y.x.66:53

Dec 3 13:05:32 hostmi named[287]: [ID 295310 daemon.notice] security: notice: denied query from [211.251.146.129].3236 for "version.bind" CHAOS

Dec 3 13:05:33 hostmi snort: [ID 702911 auth.alert] [1:257:1] DNS named version attempt [Classification: Attempted Information Leak] [Priority: 2]: {UDP} 211.251.146.129:3236 -> z.y.x.98:53

4.4.3 Probability the source address was spoofed

The IP address was probably not spoofed. This seems to be a reconnaissance attempt; it would be futile and useless to attempt to find out the version of bind by using a spoof address, obviously the version number would never be returned to the originating source if the IP was spoofed. It would be like talking to someone on a phone that has a broken earpiece.

The IP address does exist but there is no hostname or reverse lookup associated with the IP. Once again we are dealing with an IP address from the Asia Pacific region. See the whois result below.

whois whois.arin.net 211.251.146.129:

Asia Pacific Network Information Center (NETBLK-APNIC-CIDR-BLK)
These addresses have been further assigned to Asia-Pacific users.
Contact info can be found in the APNIC database,
at WHOIS.APNIC.NET or http://www.apnic.net/
Please do not send spam complaints to APNIC.

Netname: APNIC-CIDR-BLK2
Netblock: 210.0.0.0 - 211.255.255.255
Coordinator:
Administrator, System (SA90-ARIN) [No mailbox]
+61-7-3367-0490
Domain System inverse mapping provided by:

NS.APNIC.NET	203.37.255.97
SVC00.APNIC.NET	202.12.28.131
NS.TELSTRA.NET	203.50.0.137
NS.RIPE.NET	193.0.0.193

4.4.4 Description of the attack

In the logs that are presented, we see that a DNS query is being refused. Then we see that the attacker was attempting to do a 'bind.version' query. Such a query is usually part of a reconnaissance to detect the specific version of bind that is installed and once the result is received, it is very easy to see if the specific version of BIND is vulnerable. Once again we have a case of reconnaissance that have failed due to having proper configuration of bind and restrictions on accessing the bind service.

4.4.5 Attack mechanism

BIND is one of the most exploited services that exist in the history of computer. This service is known to have buffer overflow in some of its version that could allow an intruder to get root access on the machine.

The attack consists of finding the bind.version that is currently in use. Once this information is gathered it will be matched against a list of known vulnerabilities for the specific bind version. Incidentally the a nice list is at http://www.isc.org/products/BIND/bind-security.html

The SANS Institute top 20 vulnerabilities have BIND listed in third position amongst the most vulnerable UNIX services. They describe its vulnerability as follow:

> The Berkeley Internet Name Domain (BIND) package is the most widely used implementation of Domain Name Service (DNS) -- the critical means by which we all locate systems on the Internet by name (e.g., www.sans.org) without

having to know specific IP addresses -- and this makes it a favorite target for attack. Sadly, according to a mid-1999 survey, as many as 50% of all DNS servers connected to the Internet are running vulnerable versions of BIND. In a typical example of a BIND attack, intruders erased the system logs and installed tools to gain administrative access. They then compiled and installed IRC utilities and network scanning tools, which they used to scan more than a dozen class-B networks in their search for additional systems running vulnerable versions of BIND. In a matter of minutes, they had used the compromised system to attack hundreds of remote systems, resulting in many additional successful compromises. This example illustrates the chaos that can result from a single vulnerability in the software for ubiquitous Internet services such as DNS. Outdated versions of Bind also include buffer overflow exploits that attackers can use to get unauthorized access.

A good example of a severe on BIND attack would be using the NXT vulnerability. As mentioned at the official BIND site from isc.org (http://www.isc.org/products/BIND/bind-security.html) a bug in the processing of NXT records can theoretically allow an attacker to gain access to the system running the DNS server at whatever privilege level the DNS server runs at.

This exploit is clearly explained on the CIAC site. The exploit requires two systems to be successful. The first is a DNS server that will have an altered DNS table. The second machine is where the attack will take place.

Intruders alter a valid DNS server's (we will call this box [SERVER 1]) lookup table to point toward their computer [HACKER.COM] as the Authoritative Name Server for that domain. Intruders then prompt your DNS server to resolve [HACKER.COM]. [SERVER 1] passes the information back to your DNS server for the Authoritative Name Server for . Your DNS server then goes to [HACKER.COM] looking to complete the query. Once your DNS server queries [HACKER.COM] for resolution, BIND runs and the buffer overflow condition occurs.

Once the buffer overflow is executed, the following command is executed in the source code obtained by CIAC: cd /; uname -a; pwd; id;. The named service will crash as a result of the buffer overflow.

4.4.6 Correlations

Intrusion Signatures and Analysis, New Riders, Stephen Northcutt, Mark Cooper, Matt Fearnow, Karen Frederick, ISBN 0-7357-1063-5, Page 42-46

Hackers Beware, New Riders, Eric Cole, ISBN 0-7357-1009-0, Pages 606-612
www.incident.org http://www.incidents.org/archives/intrusions/msg02715.html
www.incident.org http://www.incidents.org/archives/intrusions/msg00087.html

4.4.7 Evidence of active targeting

This attack could have been active targeting. It is a few packets, a very specific attack. However if we look at the timing, we see that the probes were done extremely fast, it was probably done though the use of a tool such as Sam Spade (http://www.samspade.org/ssw/), DIG, or a script that calls nslookup with the proper parameter.

For example, if you use DIG you can use the following command line:
dig @yourdomain.com version.bind chaos txt

The result will include a line that shows the exact version in use for the DNS server. The output will be similar to the following two lines where we can see that the version is 8.2.2-P5:
;; ANSWER SECTION:
VERSION.BIND. 0S CHAOS TXT "8.2.2-P5"

4.4.8 Severity

The severity of this attack is: 0

Criticality	Lethality	System Countermeasures	Network Countermeasures	Severity
4	5	5	4	**0**

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

Scale: 0 = low and 5=high

4.4.9 Substantiation

DNS is a critical service for any organization or company. YES, you can still communicate without using the DNS service and strictly IP but who in their right mind would want to do so. The whole internet today rely on the Domain Name service to allow easy communication

between hosts, do you know by heart the IP address of your preferred search engine for example. In this case we only had a reconnaissance attempt, most likely they will not come back because they did not succeed in getting the specific bind version as it was denied. Considering the role that DNS plays I assigned a rating of 4 for criticality. Most of the DNS attacks are fairly lethal, they either disable access to the services or in some case could be used to compromise whole systems. They are easy to identify and easy to see if they are vulnerable. Considering that about 50% of the DNS systems are still vulnerable today (according to SANS), I think that this is a very lethal attack. I gave a rating of 5. Good system configuration prevented the attacker from having a successful reconnaissance of this service; I gave 5 for this category. As far as Network I gave a rating of 4 because it was monitored and detected by the IDS and it shows that the admin is worried about these types of threats.

4.4.10 Defensive recommendation

As it is very well explained in the SANS/FBI top 20 vulnerabilities, available at http://www.sans.org/top20.htm . The following steps must be taken to secure bind or if you do not need the service ensure that it is not running.

The following steps should be taken to defend against the BIND vulnerabilities:

1. Disable the BIND name daemon (called "named") on all systems that are not authorized to be DNS servers. Some experts recommend you also remove the DNS software.
2. On machines that are authorized DNS servers, update to the latest version and patch level. Use the guidance contained in the following advisories:
3. For the NXT vulnerability: http://www.cert.org/advisories/CA-99-14-bind.html For the QINV (Inverse Query) and NAMED vulnerabilities: http://www.cert.org/advisories/CA-98.05.bind_problems.html http://www.cert.org/summaries/CS-98.04.html
4. Run BIND as a non-privileged user for protection in the event of future remote-compromise attacks. (However, only processes running as root can be configured to use ports below 1024 – a requirement for DNS. Therefore you must configure BIND to change the user-id after binding to the port.)
5. Run BIND in a chroot()ed directory structure for protection in the event of future remote-compromise attacks.
6. Disable zone transfers except from authorized hosts.
7. Disable recursion and glue fetching, to defend against DNS cache poisoning.
8. Hide your version string.

4.4.11 Multiple choice test question based on the trace and analysis above

Question: Why is it important to hide your bind version from attackers?

Answer 1: It would reveal if you have a vulnerable version of bind.

Answer 2: It would reveal if you have a secondary name server.

Answer 3: It would reveal your assigned range of IP Addresses.

Answer 4: It would not matter because it is an open source program.

Correct answer: 1

Details: Insures that your DNS is configured not to reveal the current version of bind being installed. This will prevent successful reconnaissance that would reveal if your version of bind is vulnerable to known attacks.

4.4.12 Supplementary comments

Bind is one of the services that have been known to present vulnerabilities for years. It is one of the most commonly exploited services on UNIX computers. If you need this service you must ensure that you follow the security steps recommended above.

4.5 Network Detect 5

4.5.1 Source of trace

http://www.incidents.org/archives/intrusions/msg02725.html

4.5.2 Detect generated by

The logs below seems to be generated by a portscan detected through Snort trace.

```
Dec  2 07:52:30 207.61.229.195:1526 -> a.b.c.182:515 SYN ******S*
Dec  2 07:52:30 207.61.229.195:1527 -> a.b.c.183:515 SYN ******S*
Dec  2 07:52:30 207.61.229.195:1539 -> a.b.c.195:515 SYN ******S*
Dec  2 07:52:33 207.61.229.195:1445 -> a.b.c.101:515 SYN ******S*
Dec  2 07:52:33 207.61.229.195:1671 -> a.b.d.72:515 SYN ******S*
```

Dec 2 07:52:34 207.61.229.195:1797 -> a.b.d.198:515 SYN ******S*

Dec 2 07:52:34 207.61.229.195:1814 -> a.b.d.215:515 SYN ******S*

Dec 2 07:52:34 207.61.229.195:1844 -> a.b.d.245:515 SYN ******S*

Dec 2 07:52:34 207.61.229.195:1954 -> a.b.e.100:515 SYN ******S*

Dec 2 07:52:34 207.61.229.195:2141 -> a.b.f.32:515 SYN ******S*

Dec 2 07:52:37 207.61.229.195:1850 -> a.b.d.251:515 SYN ******S*

Dec 2 07:52:37 207.61.229.195:2030 -> a.b.e.176:515 SYN ******S*

Dec 2 07:52:37 207.61.229.195:2033 -> a.b.e.179:515 SYN ******S*

Dec 2 07:52:37 207.61.229.195:2038 -> a.b.e.184:515 SYN ******S*

Dec 2 07:52:37 207.61.229.195:2131 -> a.b.f.22:515 SYN ******S*

Dec 2 07:52:37 207.61.229.195:2141 -> a.b.f.32:515 SYN ******S*

Dec 2 07:52:37 207.61.229.195:2254 -> a.b.f.145:515 SYN ******S*

4.5.3 Probability the source address was spoofed

The source was probably not spoofed. It looks like a reconnaissance for some LPR vulnerabilities. This one is a SYN scan to see if the port is active. There has been a large increase in this type of reconnaissance lately. If the source was spoofed then I do not see whey you would do such attack because the result would not get back to you. Traffic is from high port to low port, which seems normal.

The IP address 207.61.229.195 is responding to ping packets from the Internet. It is some of my Canadian fellows from Ontario somewhere. It is the York College of Information technology. By contacting the college IT administrator it would be fairly easy to confirm the source and verify if they had any compromise or abuse from some of their users.

IP address: 207.61.229.195
Host name: mail.ycit.on.ca

Organization	York College of Information Technologies
Registrar	Webnames.ca (UBC Research Enterprises Inc.)
Renewal	2002/07/0
Date Approved	2000/11/22
Last Changed	2001/10/12
Description	
Registrar Number	70
Registrant Number	154494
Domain Number	154494

DNS1
ns2.bellglobal.com
DNS2
ns3.bellglobal.com
DNS3
ns4.bellglobal.com
Administrative Contact

Name	Bret D. Snider
Job Title	President & CEO
Postal Address	44 Victoria street Suite 18th Floor Toronto ON M5C1Y2 Canada
Phone	1 416 861 1808
Fax	1 416 861 9723
Email	bsd@ycit.on.ca

4.5.4 Description of the attack

The attack consist of doing a scan of a range of IP addresses to detect vulnerable service related to LPR which usually run on port 515. Once you have found active ports, you can further proble to see what is the platform being used and what is the specific version of LPR in use.

4.5.5 Attack mechanism

The LPRng port, versions prior to 3.6.24, contains a potential vulnerability, which may allow root compromise from both local and remote systems. The vulnerability is due to incorrect usage of the syslog function. Local and remote users can send string-formatting operators to the printer daemon to corrupt the daemon's execution, potentially gaining root access.

As explained on the SecurityFocus website, LPRng contains a function, use_syslog(), that returns user input to a string in LPRng that is passed to syslog() as the format string. As a result, it is possible to corrupt the program's flow of execution by entering malicious format specifiers. In testing this has been exploited to remotely elevate privileges.

4.5.6 Correlations

Wade Dauphinee Practical
http://www.giac.org/practical/Wade_Dauphinee_GCIA.doc
SANS GIAC

http://www.sans.org/y2k/040901-1200.htm
SANS GIAC
http://www.sans.org/y2k/111000-1200.htm
SANS Alert
http://www.sans.org/newlook/alerts/port515.htm
Security Focus
http://www.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=1712
CERT Advisory
http://www.cert.org/advisories/CA-2000-22.html

4.5.7 Evidence of active targeting

I do not believe that this is active targeting. This seems to be a reconnaissance. I have noted that not all hosts are being reported as being scanned in the class of address that we have, this is normal in most reconnaissance activities, the first stage usually ping to detect hosts that are alive and responding, then only the hosts responding are further scanned.

4.5.8 Severity

The severity of this attack is: 1

Criticality	Lethality	System Countermeasures	Network Countermeasures	Severity
3	5	3	4	**1**

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

Scale: 0 = low and 5=high

4.5.9 Substantiation

Print services are not the most critical element in any enterprise, unless you are a printing service of course. Although it is a pain when you can't print, you could still continue to work without such service. I have assigned a level of 3. The lethality of the attack is quite high; this is an attack that could lead to a root compromised if successful, exploits are commonly available on the net, I am giving a rating of 5 in this case. It is fairly hard to evaluate the system

countermeasure just by looking at the portscan above, so I will give a middle of the road rating of 3 due to lack of info to properly assign a value. Network countermeasure seems present as shown by the detected extract and I have assigned a value of 4.

4.5.10 Defensive recommendation

Do not run this service if it is not needed.
Apply a patch from your vendor.
Filter TCP port 515 on your screening device.
Always monitor for access to unauthorized services.

4.5.11 Multiple choice test question based on the trace and analysis above

Question: What is one of the counter measure to prevent LPRng attacks?

Answer 1: Filter incoming TCP 514 traffic.

Answer 2: Filter incoming TCP 515 traffic.

Answer 3: Filter incoming TCP 516 traffic.

Answer 4: It cannot be minimized because it uses UDP.

Correct answer: 2

Details: An effective means of minimizing the change of success of this type of attack is by denying all TCP 515 traffic from external addresses. However, you must remember that you internal users would still be able to abuse of this service.

4.5.12 Supplementary comments

The port 515 traffic was very popular following the announcement of a vulnerability for the LPRng service. Web site such as http://www.dshield.org are good place to visit to find out what are the most commonly probed ports or to find out information on the latest vulnerabilities in the wild.

5 ANALYSE THIS

For this assignment I have used the following files from http://www.research.umbc.edu/~andy/

ALERTS	OOS	SCANS
Alert.011120	Oos_Nov.20.2001	Scans.011120
Alert.011121	Oos_Nov.21.2001	Scans.011121
Alert.011122	Oos_Nov.22.2001	Scans.011122
Alert.011123	Oos_Nov.23.2001	Scans.011123
Alert.011124	Oos_Nov.24.2001	Scans.011124

5.1 Executive Summary

In this portion of the assignment, a local university that has been having numerous problems with the stability of their systems and would like to have a quick answer to what is detected in their environment, to help us they provided 5 days of data to us. The data is in the form of SNORT alert, Out of spec, and Scans files.

Even thou we have limited information on how the info was gathered, what was the environment, we will attempt to find as much details as we can from the files that have been extracted. I guess this is part of the challenge imposed upon us by this university.

I must say thanks to the author of snort_stats and snort_sort as they have helped me greatly in my dissecting the hundreds of Megs of data. Simple Linux tools such as Grep must also be mentioned because they are fast, and they can drill down faster than any human will ever be able to do. It was possible to search through a 100 megs log file with regular expression and the result was presented in about 10 seconds. (Not bad for a Linux box running on a Celeron 400Mhz). I would definitively not try this on a windows box.

It was noted that there is quite a large amount of GNUTELLA or peer to peer sharing going on over the networks. Most of the GNUTELLA traffic is from MY.NET going outbound but there is also a large portion of the scans being launched against our networks to find out if we have any hosts running GNUTELLA type of programs. The more Peer to Peer we have on the network, the more attractive we become. See the section on TCP SRC and DST outside network for more details.

Another very strange pattern of alerts detected is the "MISC traceroute". A total of 47788 alerts was detected. Most of the probes were directed to MY.NET.140.9 and a few were directed to MY.NET.70.148 and MY.NET.115.115. Please see below section 5.3 for more details on this traffic. I do believe that host MY.NET.140.9 should be look at, to see whey it is being targeted so actively by traceroutes.

There was some indication that you may have some compromised hosts on your internal networks. It was noted that some very severy portscan were taking place between internal host. The source of the scans was from MY.NET.5.75 and MY.NET.5.76. A first look tend to present these machine as doing DHCP/BOOTP queries to other hosts but a deeper look revealed that there was also other activities going on from one of the two host. MY.NET.5.76 was performing a scan that was well hidden amongst the 800,000 plus alerts of DHCP/BOOTP were mainly used as a smoke screen to hide a port scan going on at the same time. Please see the section below called: "Port Scan activities from internal hosts" for logs extract that shows this pattern. I strongly suggest that these two hosts be looked at soonest for any signs of compromised.

There is a possibility that one of your internal host MY.NET.6.44 is infected by the MyRomeo worm. I strongly recommended that you look at this host right away to verify if it is infected or not. Snort has reported some MyRomeo activities from this host to six other external hosts.

There is also another one of your host that may be infected by the Backdoor NetMetro. This backdoor use a very specific port of 5031 and 5032 and it seems that MY.NET.163.111 may be infected. Take a look at "Possible Backdoor on Internal system" below.

Host MY.NET.60.40 is using Telnet to connect to external hosts; I would recommend that you make use of a more secure remote access program. Telnet can easily be replace by SSH which offer encrypted session and you login name and password will no longer travel in clear over the network.

Most of the above vulnerabilities could have been avoided by having a screening device that allow only traffic that is required into the internal networks. If no screening device is in place or if you have a screening device that has rules that are too permissive, you should seriously consider reviewing it's policies in order to provide a greater level of protection.

5.2 TCP and DST outside network

Section 5.5 which is the statistics has details of all the top 10 hosts that triggered alerts with TCP and DST outside network. In this portion we will take the top two and analyze them further.

117	172.139.43.16	207.50.81.10	TCP SRC and DST outside network
43	172.139.43.16	204.216.217.66	TCP SRC and DST outside network

The destination address 207.50.81.10 is part of a range of address that belong to an ISP in texas. The destination port number seems to be random. The second IP address is also from an ISP provider located in San Diego. One very interesting point is the fact that the source IP 172.139.43.16 is using port 6346 as a source port. This is an indication that this machine may be using GNUTELLA Peer to Peer file sharing program or one of it's variant.

This type of traffic seems very common it is reported quite often on www.incident.org, it was correlated in Clifford Yago practical at www.sans.org/y2k/practical/Clifford_Yago_GCIA.doc.

A further look through the scan files revealed that this type of traffic was seen over 2754 times. The scan files also revealed that there is quite a few of the internal hosts that are doing SYN scan to find hosts with services on port 6346. I am not sure if this is actual SYN scan or this is simply the GNUTELLA client trying to connect to a large number of peers all at once which triggers the SYN alert.

It was also noted that the alert files contains a lot of notice on Outbound and Inbound GNUTella connections being accepted. The Outbound connections were from random IP address towards multiples of our internal hosts. The inbound connections attempts were as well from multiple external IP addresses to multiple internal hosts.

There was also very strong evidence of active targeting from one external host toward IP address MY.NET.181.180 using all kind of crafted packets. Note the fix source port used for all the scans. There is a small interval in between the scans so the attacker may have been using some tool and had to reconfigure the tool between scans. See the extract below:

```
Nov 23 12:49:49 217.226.37.31:3633 -> MY.NET.181.180:6346 NOACK ****P*SF
Nov 23 12:49:49 217.226.37.31:3670 -> MY.NET.181.180:6346 SYN ******S*
Nov 23 12:50:29 217.226.37.31:3670 -> MY.NET.181.180:6346 SPAU **UAP*S*
Nov 23 12:50:32 217.226.37.31:3670 -> MY.NET.181.180:6346 SPAU **UAP*S*
Nov 23 12:50:32 217.226.37.31:3670 -> MY.NET.181.180:47 SPAU **UAP*S*
```

Nov 23 12:50:48 217.226.37.31:3670 -> MY.NET.181.180:6346 SYN 1*****S* RESERVEDBITS

Nov 23 12:51:27 217.226.37.31:3670 -> MY.NET.181.180:0 UNKNOWN *2*A**S* RESERVEDBITS

Nov 23 12:51:33 217.226.37.31:3670 -> MY.NET.181.180:6346 SPAU **UAP*S*

Nov 23 12:51:52 217.226.37.31:3692 -> MY.NET.181.180:6346 UNKNOWN *2*APR** RESERVEDBITS

Nov 23 12:51:58 217.226.37.31:3692 -> MY.NET.181.180:1 UNKNOWN *2***R** RESERVEDBITS

Nov 23 12:51:58 217.226.37.31:3692 -> MY.NET.181.180:6346 UNKNOWN *2***R** RESERVEDBITS

Nov 23 12:52:02 217.226.37.31:3692 -> MY.NET.181.180:205 UNKNOWN *2***R** RESERVEDBITS

Nov 23 12:52:02 217.226.37.31:3692 -> MY.NET.181.180:1 UNKNOWN *2***R** RESERVEDBITS

Nov 23 12:52:33 217.226.37.31:3692 -> MY.NET.181.180:6346 UNKNOWN *2***R** RESERVEDBITS

Nov 23 12:53:12 217.226.37.31:3692 -> MY.NET.181.180:0 UNKNOWN *2*APR** RESERVEDBITS

Nov 23 12:53:12 217.226.37.31:3692 -> MY.NET.181.180:6346 UNKNOWN *2***R** RESERVEDBITS

Nov 23 12:53:30 217.226.37.31:3692 -> MY.NET.181.180:0 UNKNOWN *2*APR** RESERVEDBITS

Nov 23 12:53:33 217.226.37.31:3692 -> MY.NET.181.180:128 UNKNOWN *2***R** RESERVEDBITS

Nov 23 12:53:41 217.226.37.31:3692 -> MY.NET.181.180:6346 UNKNOWN *2***R** RESERVEDBITS

Nov 23 12:53:47 217.226.37.31:3717 -> MY.NET.181.180:6346 SYN ******S*

Nov 23 12:54:02 217.226.37.31:3719 -> MY.NET.181.180:0 NOACK 12U*PR*F RESERVEDBITS

Nov 23 12:54:02 217.226.37.31:3719 -> MY.NET.181.180:6346 NOACK 1*U**R*F RESERVEDBITS

Nov 23 12:54:38 217.226.37.31:3735 -> MY.NET.181.180:0 NOACK 12***RS* RESERVEDBITS

Nov 23 12:54:57 217.226.37.31:3735 -> MY.NET.181.180:6346 NOACK 12***RS* RESERVEDBITS

Nov 23 12:54:58 217.226.37.31:3745 -> MY.NET.181.180:6346 SYN ******S*

Nov 23 12:55:01 217.226.37.31:3747 -> MY.NET.181.180:6346 INVALIDACK 12UAPRS* RESERVEDBITS

Nov 23 12:55:18 217.226.37.31:3747 -> MY.NET.181.180:0 INVALIDACK 1**APRS* RESERVEDBITS

Nov 23 12:55:24 217.226.37.31:3747 -> MY.NET.181.180:157 NOACK 12U**RS* RESERVEDBITS

Nov 23 12:55:36 217.226.37.31:3747 -> MY.NET.181.180:0 NOACK **U*PRS*

Nov 23 12:55:37 217.226.37.31:3747 -> MY.NET.181.180:6346 NOACK 12**PRS* RESERVEDBITS

Nov 23 12:55:37 217.226.37.31:3747 -> MY.NET.181.180:6346 INVALIDACK 12UAPRS* RESERVEDBITS

Nov 23 12:55:42 217.226.37.31:3756 -> MY.NET.181.180:0 INVALIDACK 12UAPR*F RESERVEDBITS

Nov 23 12:55:45 217.226.37.31:3756 -> MY.NET.181.180:6346 NOACK 12U**R*F RESERVEDBITS

Nov 23 12:57:39 217.226.37.31:3756 -> MY.NET.181.180:6346 INVALIDACK 12UAPR*F RESERVEDBITS

Nov 23 12:58:45 217.226.37.31:3756 -> MY.NET.181.180:0 INVALIDACK 1**APR*F RESERVEDBITS

Nov 23 12:59:31 217.226.37.31:3756 -> MY.NET.181.180:6346 INVALIDACK 12UAPR*F RESERVEDBITS

Nov 23 13:03:44 217.226.37.31:3756 -> MY.NET.181.180:194 INVALIDACK 1**APR*F RESERVEDBITS

It was also noted that about a dozen of external address have been trying to scan internal hosts for port 6346. Most of them used crafted packets with all king of combination of flags.

Seeing GNUTELLA traffic is not abnormal into an environment such as a University. However it should be looked at more closely to see if this is against the university policy to allow user such service or is this permitted at large. This could consume quite a bit of bandwidth and there is also some risks associated with sharing files from untrusted sources that end up directly on internal hosts.

5.3 MISC Traceroute

A further analysis of the alert files has revealed a very intense pattern of MISC traceroute. The pattern is spread over the five days. It seems that there is a few attempts per minutes. The packets are all from High port numbers to High port numbers. The destination ports range being used are in the range of aproximatively 33422 to 33482. The source ports range are approximatively from 35775 to 60000.

Host MY.NET.140.9 is the principal target with about 98% of all probes directed toward this host. There are also hosts MY.NET.70.148 and MY.NET.115.115 that are targeted as well.

It is also noticed that one in a while there is an ICMP Destination Unreachable being sent from remote hosts to MY.NET.140.9

I have tried to find correlation for this type of attack but I failed to find anything on all the sources that I know. This is however one host that I would keep a very close eye on to ensure that all is normal.

11/20-00:32:00.372149 [**] MISC traceroute [**] 141.161.61.81:39157 -> MY.NET.140.9:33450

11/20-00:32:08.144746 [**] MISC traceroute [**] 128.192.234.130:37337 -> MY.NET.140.9:33465

11/20-00:32:13.148230 [**] MISC traceroute [**] 128.192.234.130:37337 -> MY.NET.140.9:33466

11/20-00:32:13.155462 [**] MISC traceroute [**] 132.198.101.254:56386 -> MY.NET.140.9:33466

11/20-00:32:18.175968 [**] MISC traceroute [**] 132.198.101.254:56386 -> MY.NET.140.9:33467

11/20-00:32:19.630142 [**] MISC traceroute [**] 152.1.14.3:43610 -> MY.NET.140.9:33463

11/20-00:32:25.841036 [**] MISC traceroute [**] 138.26.220.46:34832 -> MY.NET.140.9:33459

11/20-00:32:34.867250 [**] ICMP Destination Unreachable (Host Unreachable) [**] 160.36.56.17 -> MY.NET.140.9

11/20-00:32:39.216175 [**] MISC traceroute [**] 205.253.57.100:51689 -> MY.NET.140.9:33450

11/20-00:32:39.744339 [**] MISC traceroute [**] 128.32.0.29:35805 -> MY.NET.140.9:33482

11/20-00:32:44.048495 [**] ICMP Destination Unreachable (Host Unreachable) [**] 198.124.254.166 -> MY.NET.140.9

11/20-00:32:45.698876 [**] MISC traceroute [**] 128.206.119.187:52293 -> MY.NET.140.9:33479

11/20-00:32:46.507744 [**] MISC traceroute [**] 153.90.170.67:44301 -> MY.NET.140.9:33480

11/20-00:33:03.390418 [**] MISC traceroute [**] 130.132.252.244:43178 -> MY.NET.140.9:33462

11/20-00:33:05.280716 [**] MISC traceroute [**] 128.3.7.27:38679 -> MY.NET.140.9:33467

11/20-00:33:08.801273 [**] MISC traceroute [**] 198.32.163.66:46345 -> MY.NET.140.9:33473

11/20-00:33:13.634794 [**] MISC traceroute [**] 128.197.160.253:48785 -> MY.NET.140.9:33463

11/20-00:33:16.036697 [**] MISC traceroute [**] 140.142.16.229:43088 -> MY.NET.140.9:33474

11/20-00:33:16.042922 [**] MISC traceroute [**] 128.103.209.74:49307 -> MY.NET.140.9:33462

11/20-00:33:18.417073 [**] MISC traceroute [**] 160.36.56.49:49616 -> MY.NET.140.9:33466

11/20-00:33:20.408251 [**] MISC traceroute [**] 129.170.18.58:41962 -> MY.NET.140.9:33462

5.4 Port Scan activities from internal hosts

While doing the initial analysis and sorting of the alerts, scans, and oss files. There is two IP address that came out as being the IP's that generated the largest number of alerts in the alerts file. They are listed below, along with the total number of alerts.

RANKING	IP ADDRESS	NUMBER OF ALERTS
1	MY.NET.5.75	839480
2	MY.NET.5.76	422397

A further look through all of the files has revealed some very interesting facts. First we will start with host MY.NET.5.75. This host is constantly doing DHCP/BOOTP request to hosts on the internal networks. This seems at first glance like some misconfigured box or some port scanner trying to find the entire DHCP/BOOTP servers in the enterprise. For the sake of brevity, I have included only a few lines from the logs and I am not going to list all 800,000 lines.

11/20-00:38:14.840501 [**] spp_portscan: portscan status from MY.NET.5.75: 12 connections across 12 hosts: TCP(0), UDP(12) [**]

11/20-00:38:16.766388 [**] spp_portscan: portscan status from MY.NET.5.75: 22 connections across 22 hosts: TCP(0), UDP(22) [**]

11/20-00:38:18.729015 [**] spp_portscan: portscan status from MY.NET.5.75: 14 connections across 14 hosts:

A further look into the scans files revealed that the activities above have trigged some scan rules as well. We can now see that the portscan is from port 67 towards port 68. It is targeting different subnets and hosts. At this point I was about to call my analysis completed and simply deduct that it was some a DHCP port scan coming from an internal host, which raised some

question about it's intentions. Even thou this seems fairly obvious I decided to take a deeper look at the second IP address as well.

Nov 20 00:06:43 MY.NET.5.75:67 -> MY.NET.236.198:68 UDP

Nov 20 00:06:43 MY.NET.5.75:67 -> MY.NET.225.134:68 UDP

Nov 20 00:06:43 MY.NET.5.75:67 -> MY.NET.242.98:68 UDP

Nov 20 00:06:43 MY.NET.5.75:67 -> MY.NET.229.246:68 UDP

Nov 20 00:06:44 MY.NET.5.75:67 -> MY.NET.224.130:68 UDP

Nov 20 00:06:44 MY.NET.5.75:67 -> MY.NET.222.94:68 UDP

Nov 20 00:06:44 MY.NET.5.75:67 -> MY.NET.229.234:68 UDP

The host MY.NET.5.76 had very similar pattern in the alert file. It also seemed like a scan for DHCP/BOOTP servers going to different subnets and hosts.

11/20-06:03:14.452546 [**] spp_portscan: portscan status from MY.NET.5.76: 10 connections across 10 hosts: TCP(0), UDP(10) [**]

11/20-06:03:15.946056 [**] spp_portscan: portscan status from MY.NET.5.76: 17 connections across 17 hosts: TCP(0), UDP(17) [**]

11/20-06:03:17.635185 [**] spp_portscan: portscan status from MY.NET.5.76: 15 connections across 15 hosts: TCP(0), UDP(15) [**]

The interesting part came from the scan files. When I did a lookup through the scan files to see if there was any activities reported from host MY.NET.5.76, I was pleasantly surprised to see that this host was also doing scan from port 67 to port 68 but embedded into these hundreds of alerts there was a slow SYN scan going on as well. It seems that it was trying to find hosts that had a Telnet server installed. Obviously this is not normal traffic. This indicates packets that were crafted. This host and the MY.NET.5.75 host has to be looked at soonest.

Nov 20 00:02:22 MY.NET.5.76:67 -> MY.NET.203.70:68 UDP

Nov 20 00:02:22 MY.NET.5.76:62283 -> MY.NET.200.26:23 SYN ******S*

Nov 20 00:02:23 MY.NET.5.76:67 -> MY.NET.206.22:68 UDP

……Series of line deleted for brevity

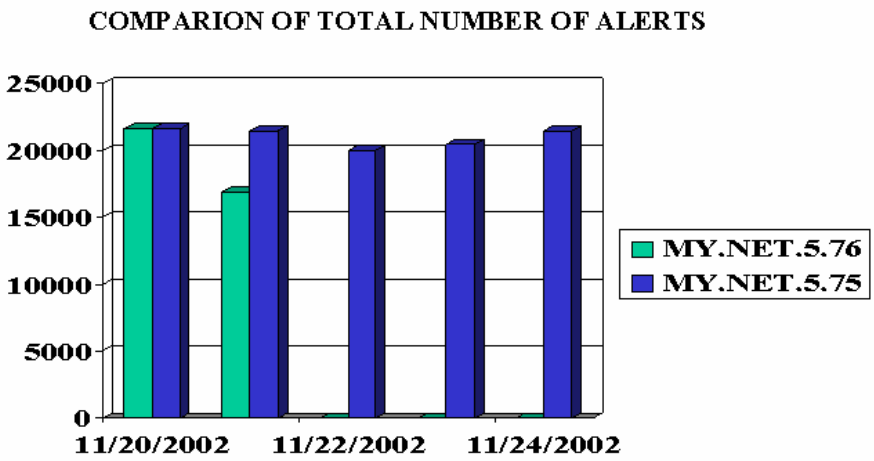
Nov 20 00:02:30 MY.NET.5.76:67 -> MY.NET.204.82:68 UDP

Nov 20 00:02:30 MY.NET.5.76:62284 -> MY.NET.200.2:23 SYN ******S*

Nov 20 00:02:30 MY.NET.5.76:67 -> MY.NET.204.14:68 UDP

Nov 20 00:02:30 MY.NET.5.76:67 -> MY.NET.203.50:68 UDP

Below you have a graph prepared in Powerpoint that shows the distribution of alerts per day for the two hosts mentioned above. We can clearly see that MY.NET.5.76 was very activate at the beginning of the period and the MY.NET.5.76 remained active throughout the five days.



5.5 Possible worm from internal host

There is one of our host that may be infrected with the MyRomeo Worm. Host MY.NET.6.44 should be looked at soonest for any signs of infection.

This worm arrives with one of several different subject lines and has two attachments named Myjuliet.chm and Myromeo.exe. Once you read the message, the two attachments are automatically saved and launched. When launched, this worm attempts to send itself out to all names in the Microsoft Outlook address book using one of several Internet mail servers located in Poland. Otherwise this worm does no harm to the infected system.

This activity was detected by one of the MyRomeo worm rules listed below.

alert tcp any 110 -> any any (msg:"Virus - Possible MyRomeo Worm"; content: "myromeo.exe"; nocase; sid:723; classtype:misc-activity; rev:3;)

alert tcp any 110 -> any any (msg:"Virus - Possible MyRomeo Worm"; content: "myjuliet.chm"; nocase; sid:724; classtype:misc-activity; rev:3;)

alert tcp any 110 -> any any (msg:"Virus - Possible MyRomeo Worm"; content: "ble bla"; nocase; sid:725; classtype:misc-activity; rev:3;)

alert tcp any 110 -> any any (msg:"Virus - Possible MyRomeo Worm"; content: "I Love You"; sid:726; classtype:misc-activity; rev:3;)

alert tcp any 110 -> any any (msg:"Virus - Possible MyRomeo Worm"; content: "Sorry... Hey you !"; sid:727; classtype:misc-activity; rev:3;)

alert tcp any 110 -> any any (msg:"Virus - Possible MyRomeo Worm"; content: "my picture from shake-beer"; sid:728; classtype:misc-activity; rev:3;)

alert tcp any 110 -> any any (msg:"Virus - Possible MyRomeo Worm"; content: "Matrix has you..."; sid:735; classtype:misc-activity; rev:3;)

11/20-07:50:56.092298 [**] Virus - Possible MyRomeo Worm [**] MY.NET.6.44:110 -> 141.157.98.233:49152
11/20-09:40:05.354923 [**] Virus - Possible MyRomeo Worm [**] MY.NET.6.44:110 -> 18.251.1.21:1152
11/20-09:49:53.872185 [**] Virus - Possible scr Worm [**] MY.NET.6.44:110 -> 24.198.4.57:1075
11/20-17:34:14.350251 [**] Virus - Possible scr Worm [**] MY.NET.6.44:110 -> MY.NET.207.246:1102
11/20-23:02:31.243239 [**] Virus - Possible MyRomeo Worm [**] MY.NET.6.44:110 -> 209.20.225.88:3859
11/23-22:04:08.497619 [**] Virus - Possible MyRomeo Worm [**] MY.NET.6.44:110 -> 216.136.226.90:1115

5.6 Possible backdoor on internal system

The alert file has revealed some indication that one of the internal host MY.NET.163.111 may be infected by the NetMetro backdoor. The Net Metropolitain program is a program similar to Back Orifice, where an attacker can remotely control and monitor a host. This is very deadly because it will give an intruder an open door directly into the internal network. It seems that the logs entries below were generated by one of these two Snort rules. Please not the fix source port of 5031 (which cannot be changed in Net Metropolitan) and the fix destination port of 1296.

The MY.NET.163.111 host should be verified soonest to know if the host is infected or not. It is easy to detect infection because a file named NMS.exe will be present on the infected host.

alert tcp $HOME_NET any -> $EXTERNAL_NET 5032 (msg:"BACKDOOR NetMetro File List"; flags: A+; content:"|2D 2D|"; reference:arachnids,79; sid:159; classtype:misc-activity; rev:3;)

alert tcp $EXTERNAL_NET 5031 -> $HOME_NET !53:80 (msg:"BACKDOOR NetMetro Incoming Traffic"; flags: A+; reference:arachnids,79; sid:160; rev:1;)

11/20-13:01:33.051631 [**] BACKDOOR NetMetro Incoming Traffic [**] 204.178.125.65:5031 -> MY.NET.163.111:1296

11/20-13:01:36.701036 [**] BACKDOOR NetMetro Incoming Traffic [**] 204.178.125.65:5031 -> MY.NET.163.111:1296

11/20-13:01:36.801056 [**] BACKDOOR NetMetro Incoming Traffic [**] 204.178.125.65:5031 -> MY.NET.163.111:1296

11/20-13:01:37.323920 [**] BACKDOOR NetMetro Incoming Traffic [**] 204.178.125.65:5031 -> MY.NET.163.111:1296

11/20-13:01:48.400531 [**] BACKDOOR NetMetro Incoming Traffic [**] 204.178.125.65:5031 -> MY.NET.163.111:1296

11/20-13:01:48.521576 [**] BACKDOOR NetMetro Incoming Traffic [**] 204.178.125.65:5031 -> MY.NET.163.111:1296

11/20-13:01:50.645031 [**] BACKDOOR NetMetro Incoming Traffic [**] 204.178.125.65:5031 -> MY.NET.163.111:1296

11/20-13:01:59.224099 [**] BACKDOOR NetMetro Incoming Traffic [**] 204.178.125.65:5031 -> MY.NET.163.111:1296

5.7 Possible Trojan on Internal hosts

It seems that there is some internal hosts connecting to external that may be compromise by the Subseven Trojan. There are also connections being established from external hosts to internal hosts that may be compromised by the subseven Trojan.

The subseven Trojan is another backdoor program that allow an attacker to control a computer remotely as if the attacker was sitting directly at the host. This means that a compromised internal host is allowing a remote attacker to have full access to the internal network.

As mentioned above, ensure that you verify hosts for the presence of this Trojan. Please refer to http://www.sans.org/newlook/resources/IDFAQ/subseven.htm which is a very thorough analysis of this Trojan. It has very detailed instruction on detecting subseven and step to remove it if you are infected.

The following snort rules are used to detect this Trojan:

alert tcp $HOME_NET 1243 -> !$HOME_NET any (msg:" TROJAN ACTIVITY-Possible Subseven"; flags:SA;)

alert tcp any any -> any any (msg:"TROJAN ACTIVITY-Possible SubSeven access"; content:"connected. time/date"; flags:PA;)

alert tcp !$HOME_NET any -> $HOME_NET 6776 (msg:"TROJAN ATTEMPT- SubSeven access"; flags:S;)

alert tcp !$HOME_NET any -> $HOME_NET 6711 (msg:"TROJAN ATTEMPT- Deep Throat/SubSeven"; flags:S;)

alert tcp !$HOME_NET any -> $HOME_NET 1243 (msg:"TROJAN ATTEMPT- Subseven"; flags:S;)

Nov 21 10:47:52 205.188.233.185:27374 -> MY.NET.145.197:6970 UDP

Nov 21 10:47:56 205.188.233.185:27374 -> MY.NET.145.197:6970 UDP

Nov 21 10:48:00 205.188.233.185:27374 -> MY.NET.145.197:6970 UDP

Nov 21 10:48:04 205.188.233.185:27374 -> MY.NET.145.197:6970 UDP

Nov 21 10:48:08 205.188.233.185:27374 -> MY.NET.145.197:6970 UDP

Nov 21 10:48:12 205.188.233.185:27374 -> MY.NET.145.197:6970 UDP

Nov 21 10:48:14 205.188.233.185:27374 -> MY.NET.145.197:6970 UDP

Nov 21 10:48:20 205.188.233.185:27374 -> MY.NET.145.197:6970 UDP

Nov 21 10:48:24 205.188.233.185:27374 -> MY.NET.145.197:6970 UDP

Nov 21 10:48:28 205.188.233.185:27374 -> MY.NET.145.197:6970 UDP

11/22-17:37:32.820928 [**] Possible trojan server activity [**] MY.NET.11.4:80 -> MY.NET.16.42:27374

11/22-17:37:32.820997 [**] Possible trojan server activity [**] MY.NET.16.42:27374 -> MY.NET.11.4:80

11/22-17:37:32.821279 [**] Possible trojan server activity [**] MY.NET.16.42:27374 -> MY.NET.11.4:80

11/22-17:37:32.974184 [**] Possible trojan server activity [**] MY.NET.11.4:80 -> MY.NET.16.42:27374

11/22-18:57:05.146139 [**] Possible trojan server activity [**] MY.NET.60.14:80 -> 24.4.252.249:27374

11/22-18:57:05.146205 [**] Possible trojan server activity [**] MY.NET.60.14:80 -> 24.4.252.249:27374

11/22-18:57:05.203149 [**] Possible trojan server activity [**] MY.NET.60.14:80 -> 24.4.252.249:27374

11/23-05:46:58.120768 [**] Possible trojan server activity [**] MY.NET.54.3:80 -> 216.73.25.3:27374

11/23-05:46:58.686201 [**] Possible trojan server activity [**] MY.NET.54.3:80 -> 216.73.25.3:27374

11/23-08:00:57.218777 [**] Possible trojan server activity [**] MY.NET.253.114:80 -> 24.3.0.35:27374

11/23-08:00:57.255889 [**] Possible trojan server activity [**] MY.NET.253.114:80 -> 24.3.0.35:27374

11/23-13:13:39.987648 [**] Possible trojan server activity [**] 208.170.69.12:4575 -> MY.NET.190.194:27374

11/23-13:13:45.998420 [**] Possible trojan server activity [**] 208.170.69.12:4575 -> MY.NET.190.194:27374

11/23-14:14:52.607323 [**] Possible trojan server activity [**] 24.4.252.178:27374 -> MY.NET.253.114:80

11/23-14:14:52.622265 [**] Possible trojan server activity [**] MY.NET.253.114:80 -> 24.4.252.178:27374

5.8 Cleaning up and pasting together alerts, oos, and scans files.

This section explain how the files were cleaned up from comments and then concatenated together into one big file for easier analysis.

The first step that I have taken is to download all of the needed files to my Linux computer (Red Hat 7.2). I prefer to use Linux as a platform for the initial analysis as we have more powerful query and parsing tools such as grep and awk available. Notepad is not very adept at dealing with 70 megs files but VI will happily let you do so.

Once all of the files were in place, I used VI to removed header at the beginning of the files. These headers are nice to indicate what date the data was collected but creates problems once we start analyzing our data.

Once all headers were removed, I used the 'cat' command to concatenate all of the alert files into a single big file that will be used for analysis. This can be very easily accomplished with the following command (replace the filename below with the proper filename):

cat alertfile1.txt alertfile2.txt alertfile3.txt alertfile4.txt alertfile5.txt >> onfile.txt

I did the same with the OOS and the Scan files.

Now I have 3 files called oss.txt, scans.txt, and alert.txt In a few seconds I can look through all these files for traffic relates to a specific IP address. For example, if I wish to lookup all traffic related to IP 192.168.0.1, I would simply issue the command **grep 192.168.0.1 *.txt** and all related data will is presented on the screen. A simple redirect allows you to send the result to a file or to add the result to an existing file for perusal at a later time.

5.9 Sorting out alert data

I have started my analysis by using the snort_sort.pl script. This script takes for input the snort alert log file and generates a nicely formatted web page that shows what are the attacks reported in the alert files and you also have the ability to click on IP addresses to find further information through whois if you choose to have the output formatted as HTML.

The snort_sort.pl script does not give you any statistics. It simply tells you all of the attacks that have been performed and list for each category the alerts that belong to this category. So if you have a 60 Megs file to begin with, you will end up with a 60 Megs file of sorted data. It is very easy from its output to further analyze and find how many of each type of attack was performed but it must be done manually or you could also send the output to a file as plain text and import it into your favorite spreadsheet for further analysis of specific number of attacks per IP, top source, top destination, etc... Below you have a resume of all the attack types that were detected. It must be noted that the snort_sort.pl script does not sort in alphabetical order, so I had to cut and past the list into excel to properly sort it as below.

There was a total of 143 different attacks detected in the alert files.

Attempted Sun RPC high port access
BACKDOOR NetMetro File List
BACKDOOR NetMetro Incoming Traffic
beetle.ucs
connect to 515 from inside
connect to 515 from outside
CS WEBSERVER - external ftp traffic
CS WEBSERVER - external ssh traffic
CS WEBSERVER - external web traffic
DDOS mstream client to handler
DDOS shaft client to handler
DNS named iquery attempt
DNS zone transfer
EXPLOIT NTPDX buffer overflow
EXPLOIT x86 NOOP
EXPLOIT x86 setgid 0
EXPLOIT x86 setuid 0
EXPLOIT x86 stealth noop
External FTP to HelpDesk MY.NET.70.50
External FTP to HelpDesk MY.NET.83.197
External RPC call
FINGER redirection
FTP CWD - possible warez site
FTP DoS ftpd globbing
FTP MKD - possible warez site
FTP MKD . - possible warez site
HelpDesk MY.NET.53.29 to External FTP
HelpDesk MY.NET.70.50 to External FTP
High port 65535 tcp - possible Red Worm - traffic
High port 65535 udp - possible Red Worm - traffic
ICMP Address Mask Request (Undefined Code!)
ICMP Destination Unreachable (Communication Administratively Prohibited)

ICMP Destination Unreachable (Fragmentation Needed and DF bit was set)
ICMP Destination Unreachable (Host Unreachable)
ICMP Destination Unreachable (Network Unreachable)
ICMP Destination Unreachable (Protocol Unreachable)
ICMP Echo Request Broadscan Smurf Scanner
ICMP Echo Request BSDtype
ICMP Echo Request CyberKit 2.2 Windows
ICMP Echo Request Delphi-Piette Windows
ICMP Echo Request L3retriever Ping
ICMP Echo Request Nmap or HPING2
ICMP Echo Request Sun Solaris
ICMP Echo Request webtrends scanner
ICMP Echo Request Windows
ICMP Fragment Reassembly Time Exceeded
ICMP Parameter Problem (Unspecified Error)
ICMP Photuris (Undefined Code!)
ICMP redirect (Host)
ICMP Redirect (Network)
ICMP SKIP (Undefined Code!
ICMP Source Quench
ICMP Source Quench (Undefined Code!)
ICMP Timestamp Request (Undefined Code!)
ICMP traceroute
IDS475/web-iis_web-webdav-propfind
IDS50/trojan_trojan-active-subseven
Incomplete Packet Fragments Discarded
INFO - Possible Squid Scan
INFO - Web Dir listing
INFO FTP anonymous FTP
INFO Inbound GNUTella Connect accept
INFO Inbound GNUTella Connect request
INFO MSN IM Chat data
INFO Napster Client Data
INFO napster login
INFO napster new user login
INFO Outbound GNUTella Connect accept
INFO Possible IRC Access
INFO VNC Active on Network
MISC Cisco Catalyst Remote Access
MISC Large ICMP Packet
MISC Large UDP Packet
MISC PCAnywhere Startup
MISC Source Port 20 to <1024
MISC source port 53 to <1024
MISC traceroute
NMAP TCP ping!

Null scan!
Port 55850 tcp - Possible myserver activity - ref. 010313-1
Possible trojan server activity
Queso fingerprint
RFB - Possible WinVNC - 010708-1
RPC tcp traffic contains bin_sh
SCAN - wayboard request - allows reading of arbitrary files as http service
SCAN FIN
SCAN Proxy attempt
SCAN Synscan Portscan ID 19104
SCAN XMAS
SITE EXEC - Possible wu-ftpd exploit - GIAC000623
SMB Name Wildcard
SMTP chameleon overflow
SMTP relaying denied
SNMP public access
spp_http_decode: CGI Null Byte attack detected
spp_http_decode: IIS Unicode attack detected
SUNRPC highport access!
TCP SRC and DST outside network
TELNET access
TELNET login incorrect
TFTP - Internal TCP connection to external tftp server
TFTP - Internal UDP connection to external tftp server
Tiny Fragments - Possible Hostile Activity
Traffic from port 53 to port 123
Virus - Possible MyRomeo Worm
Virus - Possible NAIL Worm
Virus - Possible pif Worm
Virus - Possible scr Worm
Virus - SnowWhite Trojan Incoming
Watchlist 000220 IL-ISDNNET-990517
Watchlist 000222 NET-NCFC
WEB-CGI csh access
WEB-CGI files.pl access
WEB-CGI finger access
WEB-CGI formmail access
WEB-CGI ksh access
WEB-CGI redirect access
WEB-CGI rsh access
WEB-CGI scriptalias access
WEB-CGI survey.cgi access
WEB-CGI tsch access
WEB-CGI w3-msql access
WEB-CGI webgais access
WEB-FRONTPAGE _vti_rpc access

WEB-FRONTPAGE access.cnf access
WEB-FRONTPAGE fpcount.exe access
WEB-FRONTPAGE service.cnf access
WEB-FRONTPAGE shtml.exe
WEB-IIS .cnf access
WEB-IIS _vti_inf access
WEB-IIS asp-dot attempt
WEB-IIS Unauthorized IP Access Attempt
WEB-IIS view source via translate header
WEB-MISC .htaccess access
WEB-MISC /etc/passwd
WEB-MISC 403 Forbidden
WEB-MISC Attempt to execute cmd
WEB-MISC compaq nsight directory traversal
WEB-MISC count.cgi access
WEB-MISC guestbook.cgi access
WEB-MISC http directory traversal
WEB-MISC Invalid URL
WEB-MISC L3retriever HTTP Probe
WEB-MISC Lotus Domino directory traversal
WEB-MISC prefix-get //
X11 outgoing
x86 NOOP - unicode BUFFER OVERFLOW ATTACK

5.10 Statistics

In this part of the practical, I have used the Perl script snort_stat.pl. This script gives us a whole lot of details about the alerts that were generated by SNORT.

5.10.1 List of IP and number of alerts (Top 20)

Below you have a list of IP that have generated the largest number of alerts. It is very impressive to see that a single IP could generate 839480 alerts.

RANKING	IP ADDRESS	NUMBER OF ALERTS
1	MY.NET.5.75	839480
2	MY.NET.5.76	422397
3	MY.NET.87.50	242420
4	205.188.233.153	23812
5	205.188.233.185	21366
6	205.188.246.121	18358

7	205.188.244.57	16886
8	205.188.233.121	16509
9	MY.NET.97.16	13488
10	MY.NET.150.220	10606
11	205.188.244.121	8653
12	217.136.7.67	8635
13	MY.NET.150.246	7543
14	MY.NET.253.10	6658
15	209.235.8.118	6544
16	MY.NET.97.173	5981
17	MY.NET.97.212	5923
18	MY.NET.97.169	5853
19	217.57.186.124	5778
20	140.109.73.142	5064

5.10.2 Greatest number of alerts with TCP SRC and DST outside network

In the resume below you have statistics on which hosts has generated the greatest number of alerts with TCP SRC and DST outside network. For the sake of brevity, I have kept only the top 10 offending hosts. You notice that 172.139.43.16 has generated the greatest number of alerts and that they were directed at 207.50.81.10.

# of attacks	From	To	With
117	172.139.43.16	207.50.81.10	TCP SRC and DST outside network
43	172.139.43.16	204.216.217.66	TCP SRC and DST outside network
7	205.188.248.57	192.168.0.27	TCP SRC and DST outside network
7	172.173.78.112	216.201.211.177	TCP SRC and DST outside network
6	209.97.68.137	192.168.0.2	TCP SRC and DST outside network
6	24.180.202.54	206.248.228.40	TCP SRC and DST outside network
5	6.0.0.0	6.0.0.0	TCP SRC and DST outside network
5	169.254.101.152	205.188.52.251	TCP SRC and DST outside network
4	169.254.81.17	172.25.1.5	TCP SRC and DST outside network
4	152.163.226.153	169.254.233.165	TCP SRC and DST outside network

5.10.3 Percentage and number of attacks from a host to a destination

The next table below presents a slightly different view. It shows a percentage and number of alerts from a single host to a single destination. As you notice most of the IP listed above are listed here, it does not change much for the top 10 but it did changed as we went down the list.

%	# of attacks	From	To
33.62	117	172.139.43.16	207.50.81.10
12.36	43	172.139.43.16	204.216.217.66
2.01	7	172.173.78.112	216.201.211.177
2.01	7	205.188.248.57	192.168.0.27
1.72	6	24.180.202.54	206.248.228.40
1.72	6	209.97.68.137	192.168.0.2
1.44	5	169.254.101.152	205.188.52.251
1.44	5	6.0.0.0	6.0.0.0
1.15	4	152.163.226.153	169.254.233.165
1.15	4	152.163.226.57	169.254.233.165

5.10.4 Percentage and number of attacks from one host to any with same method

The table below presents a summary that is once again a bit different. This time it is from a single host to any other host. We notice that there is some new IP range being introduced that we have not seen so far.

%	# of attacks	From	Type
45.98	160	172.139.43.16	TCP SRC and DST outside network
10.06	35	192.168.1.106	TCP SRC and DST outside network
9.48	33	192.168.1.100	TCP SRC and DST outside network
4.31	15	134.192.130.90	TCP SRC and DST outside network
3.16	11	172.173.78.112	TCP SRC and DST outside network
2.59	9	24.180.202.54	TCP SRC and DST outside network
2.30	8	169.254.101.152	TCP SRC and DST outside network
2.01	7	205.188.248.57	TCP SRC and DST outside network
1.72	6	209.97.68.137	TCP SRC and DST outside network

1.44	5	6.0.0.0	TCP SRC and DST outside network

5.10.5 Percentage and number of attacks to one certain host

The table below presents the number of alerts against a host. Some of the numbers should jumpt to your eyes right away, such as number 117, it is the same number that we have seen above in the first two tables that we introduce, it is the number of attack between IP 172.139.43.16 and 207.50.81.10, so we can conclude that the greatest number of attack against a single host was 117 attacks originating from 172.139.43.16.

%	# of attacks	To	Type
33.62	117	207.50.81.10	TCP SRC and DST outside network
12.36	43	204.216.217.66	TCP SRC and DST outside network
3.45	12	169.254.233.165	TCP SRC and DST outside network
2.30	8	172.25.1.5	TCP SRC and DST outside network
2.01	7	216.201.211.177	TCP SRC and DST outside network
2.01	7	192.168.0.27	TCP SRC and DST outside network
1.72	6	206.248.228.40	TCP SRC and DST outside network
1.72	6	192.168.0.2	TCP SRC and DST outside network
1.44	5	6.0.0.0	TCP SRC and DST outside network
1.44	5	205.188.52.251	TCP SRC and DST outside network

5.10.6 Portscans performed to/from HOME_NET

The table below is a summary of the number of scans attempts from a single IP address.

Scan Attempts	Source Address
263	199.183.24.194
121	65.14.160.211
83	204.152.184.75
36	217.136.7.67
33	210.73.44.199
25	66.114.106.22
21	212.51.220.121

20	217.57.186.124
18	217.226.37.31
16	24.101.109.213

5.11 Five addressees and their registration information

Below you will find five addressees that I have selected and their registration information. The first IP I have retained is 199.183.24.194 because this is the top scanner. I have also retained IP address 172.139.43.16 because this is the host with the most attack against a single IP address and also the host with the most attack against more than one machine. I have also retained IP address 169.254.101.152 because this is a host that has been targeting one of our IP very severely with port 137 SMB attacks. The fourth IP that I have retained is 207.50.81.10, which is the preferred target; it would be interesting why this is an interesting target. The fifth IP that I have retained is 152.163.226.153, which belong to the 52.163.x.x network, this network is detected throughout the files and I would like to find out more about them.

5.11.1 199.183.24.194

This one is really interesting. Red Hat scanning our environment to dead. Very bizarre indeed. It would highly surprise me that it is in fact from Red Hat, it seems that the person spent too much time going through a class B network.

whois whois.arin.net 199.183.24.194:

ICG NetAhead, Inc. (NET-ICG-BLK-BLK4-C) ICG-BLK-BLK4-C 199.183.16.0 - 199.183.143.255

Red Hat Software (NET-REDHAT) REDHAT 199.183.24.0 - 199.183.24.255

5.11.2 172.139.43.16

This one was from America Online. This reassures me. This is very likely that a good old script kiddie would launch a series of attack from there. He! I am on a dialup link they cannot trace it back to me, have you heard of abuse@aol.com

America Online, Inc. (NETBLK-AOL-172BLK)
12100 Sunrise Valley Drive
Reston, VA 20191
US

Netname: AOL-172BLK
Netblock: 172.128.0.0 - 172.191.255.255
Maintainer: AOL

Coordinator:
America Online, Inc. (AOL-NOC-ARIN) domains@AOL.NET
703-265-4670
Domain System inverse mapping provided by:
DAHA-01.NS.AOL.COM 152.163.159.233
DAHA-02.NS.AOL.COM 205.188.157.233

5.11.3 169.254.101.152

Not a whole lot of success with this one. I have tried to dig deeper onto LINKLOCAL but no chance. Search engine have a linklocal.com listed but the links are dead.

IANA (NETBLK-LINKLOCAL)
Internet Assigned Numbers Authority
4676 Admiralty Way, Suite 330
Marina del Rey, CA 90292-6695
US

Netname: LINKLOCAL
Netblock: 169.254.0.0 - 169.254.255.255

Coordinator:
Internet Corporation for Assigned Names and Numbers (IANA-ARIN) res-ip@iana.org
(310) 823-9358

Domain System inverse mapping provided by:

BLACKHOLE-1.IANA.ORG 192.0.32.18
BLACKHOLE-2.IANA.ORG 192.0.32.19

5.11.4 207.50.81.10

This one does not require much comment. Another cable modem user, with tons of bandwidth and some time on his hands to try out the latest tools.

Cable & Wireless USA (NETBLK-CW-09BLK) CW-09BLK 207.48.0.0 - 207.51.255.255
TCA Internet - Tyler (NETBLK-CW-207-50-80) CW-207-50-80
207.50.80.0 - 207.50.95.255

5.11.5 152.163.226.153

Once again, our friends from AOL coming at us. Cable modem, xDSL provider, AOL and other large providers are all common source of attacks.

America Online (NET-ANS-BNET8)
12100 Sunrise Valley Drive
Reston, VA 20191
US

Netname: AOL-BNET
Netblock: 152.163.0.0 - 152.163.255.255
Coordinator:
America Online, Inc. (AOL-NOC-ARIN) domains@AOL.NET
703-265-4670

Domain System inverse mapping provided by:

DNS-01.NS.AOL.COM 152.163.159.232
DNS-02.NS.AOL.COM 205.188.157.232

6 REFERENCES

The following references were used for the first portion of the assignment:

- Security Essentials for the Home Network, Mike Burden,May 2, 2001, http://www.sans.org/infosecFAQ/homeoffice/home_net.htm
- Info on Demarc and screen captures obtained from: http://www.demarc.org
- The Demarc user guide http://www.demarc.com/userguide/
- Info on the internals of Demarc, provided by Anthony from Demarc through a series of email messages
- Guidelines for cryptography policies http://www.cybercrime.gov/oeguide.htm
- Securing Demarc with stunnel http://www.mindspring.com/~joelmoses/demarc_stunnel.html
- MySql, http://www.mysql.org
- Apache web server, http://www.apache.org
- Perl, http://www.cpan.org
- Snort IDS, http://www.snort.org
- OpenSSL, http://www.openssl.org

The following references were used for the reminder of the assignment:

- Practical Assignment of Faud Khan http://www.sans.org/y2k/practical/Faud_Khan_GCIA.doc
- SANS/FBI Top 20 vulnerabilities http://www.sans.org/top20.htm
- Dshield web site, http://www.dshield.org/
- Faud Khan Practical, http://www.sans.org/y2k/practical/Faud_Khan_GCIA.doc Tips on using snortsort.pl and snortstat.pl

- Win Miller Practical, http://www.sans.org/y2k/practical/Win_Miller_GCIA.doc
  Extraction of his AWK reduction script
- SamSpade IP whois, http://www.samspade.org/
- Network tools online, http://www.network-tools.com
- Remote procedure call and portmapper daemon
  http://www.uwsg.iu.edu/usail/network/services/portmapper.html
- Analysis of NIMDA, http://www.f-secure.com/v-descs/nimda.shtml
- Web Server Folder Traversal,
  http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms00-078.asp
- Netscantools Standard Edition, http://www.netscantools.com/nstpro_rpcinfo.html
- SSH Version 3.0.0Vulnerability, http://www.ssh.com/products/ssh/exploit.cfm
- What version of Bind, http://www.freebsddiary.org/bind-version.php
- SecurityFocus LPRng vulnerability,
  http://www.securityfocus.com/cgi-bin/vulns-item.pl?section=discussion&id=1712
- Shell sorting Script, http://www.sans.org/y2k/practical/Jacomo_Piccolini_GCIA.doc
- MyRomeo Worm, http://www.sarc.com/avcenter/venc/data/w32.blebla.worm.html
- NetMetro Backdoor, http://www.dark-e.com/archive/trojans/NetMetro/100/index.shtml
- SubSeven Trojan, http://www.symantec.com/avcenter/venc/data/backdoor.subseven.html
- SANS ID FAQ, http://www.sans.org/newlook/resources/IDFAQ/oddports.htm