GIAC Certified Intrusion Analyst (GCIA) Practical Assignment

Version 3.3

(Revised August 19, 2002)

R. D. Smith

22 December 2002

Table of Contents

List of Figures

List of Tables

1 Part 1 - How to setup Snort, MySQL and Acid on Mac OS X (10.2)

1.1 Purpose of document

This document will guide a user through the installation of Snort, MySQL, and ACID on Mac OS X (10.2), also known as Jaguar, it's Apple code name. It will also guide the user through some of the processes of securing the machine. The intention is to give users that are new to any of the software the opportunity to build an enterprise-class system.

The following list of software will be installed on the machine:

- Mac OS X (v10.2.2) with the Developer Tools
- Snort (v1.90, build 209)
- MySQL (v3.23.52)
- PHP (v4.2.3)
- ADOdb (v2.43)
- PHPlot (v4.4.6)
- libjpeg.a (v0.6b)
- libpng.a (v1.2.5)
- GD (v1.8.4)
- ACID (v0.96b21)

The FreeBSD installation guide written by Keith Tokash inspired this paper.

1.1.1 Assumptions

The user has at least a little (a few months) experience with a Mac OS X and it's Unix-like underpinnings. This machine will be a dedicated Snort box.

1.1.2 Conventions

Command line input will be in the Arial font: `command line`. For commands that are entered by an administrator, a prompt of "`%`" will preface the command (but should not be typed as part of the input). For the root user, a "`#`" will be used as the prompt.

Responses at the command line will use the italicized Arial font: *response*

Directory and files in the text will be in the Courier New font: `/Directory/file`

1.2 Mac OS X Installation

1.2.1 Hardware

This installation can be done on any Macintosh platform that will run Jaguar. I used a flat-panel iMac for a test bed. I would recommend use a machine that has an available PCI slot for a second Ethernet card. The second card will allow the machine to connected through a "receive-only" Ethernet cable to the monitored network while using the other for monitoring the Snort output remotely.

I used the directions from Sam Ng (http://www.geocities.com/samngms/sniffing_cable/) to build my cable. It involves inserting a capacitor in the transmit lines from the Snort machine. The capacitor creates a high-pass filter that should prevent the passing of data over the wire by removing the step transition from the transmitted waveform. The CRC error rate of the resulting waveform should be high and prevent the flow of data but not prevent the hub at the other end of the cable from seeing the Snort machine. Different directions for creating a receive-only cable are available from ironcomet.com (http://www.ironcomet.com/sniffer.shtml). These directions have you tie the transmit lines back to the receive lines to prevent the data flow.

The machine shouldn't need a large hard drive for the operating system. A 20 GB hard drive should be acceptable for the installing the system and necessary software. Additional storage space may be needed to archive and store alert files. This storage could be on internal SCSI or IDE drives or external FireWire drives. For permanent storage of old alert files, the files can be burned to CDs or DVDs.

1.2.2 Mac OS X Installation

Disconnect the machine form the network before beginning. Boot from the Jaguar installation CD by inserting the first CD and then selecting "Reboot" from the Apple menu. Hold down the C key until you see the gray Apple logo.

Mac OS X should be reinstalled and the hard drive should be reformatted to UFS. The goal is to remove any unnecessary software including Mac OS 9. The hard drive can be reformatted by either using the Disk Utility or as part of the Jaguar installation. The difference between the options is Disk Utility allows you to repartition the hard drive and OS installer will not.

To use Disk Utility, start it by selecting "Open Disk Utility…" from the Installer menu once you see the Jaguar installation screen.

To use the installation to reformat, begin the installation by following the on-screen instructions. On the Select Destination screen, click on the "Options…" button, and then select the Erase and Install option. Select "Unix File System" as the "Format disk as" option.

Next we will minimize the amount of software that needs to be installed while enduring that the necessary packages are installed. At the Installation Type screen, click on the Customize button in the lower left portion of the window. This will allow you to prevent the installation of unnecessary software. Ensure that the BSD Subsystem is selected. Remove the check marks from the following items (unless you need them for a specific purpose)

- Additional Applications.
- Any unnecessary printer drivers (all of them if possible)
- Additional Asian Fonts
- Localized files.

Finish the installation of the software normally. The machine will prompt you to remove and insert the second installation CD if it is required.

1.2.3 On the Initial Boot and Post-installation

The installation program will begin configuring the OS and request information for registration, create the initial administrator account, and set up the network configuration. The installation program will then try to connect the Apple Computer via the network to send in the registration. It will not be able to since the network is disconnected. Continue on with the post installation setup.

Once the post-installation program is finished you will see the Desktop. Open the System Preferences. While we have this open we will do some initial security related setup.

1. Disable auto-login by selecting Accounts and remove the check mark from the box next to "Log in automatically as…" near the bottom of the panel (See Figure 1-1)

Figure 1-1, Automatic Log In Checkbox

2. Start the built-in firewall by selecting the Sharing preferences. Go to the Firewall tab and click on the Start button. Mac OS X has ipfw built-in. There are several GUI applications that can be used to configure the firewall but the built-in interface is sufficient for initial use. The default settings of the OS follow the "deny all, allow by exception" philosophy.

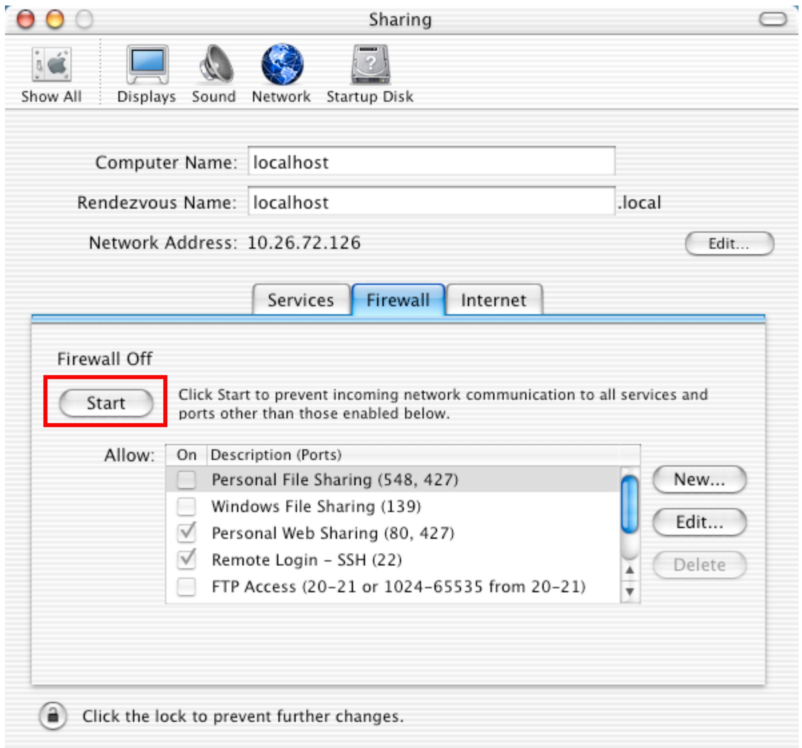


Figure 1-2, Apple Firewall Preference Pane

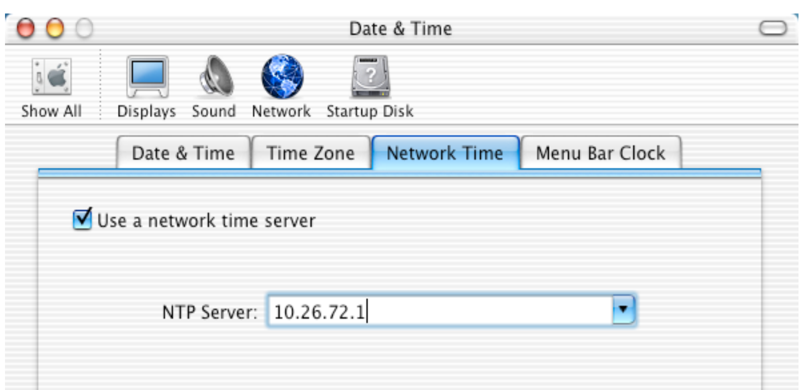


Figure 1-3, Network Time Preference Pane

3. Open the Date and Time panel. Go to the Network Time tab. Put a checkmark in the box next to "Use a network time server." Click on the NTP Server box and delete the contents and enter a

NTP server that the machine can reach as you can see in Figure 1-3. This step in not necessary for machine with only one Ethernet card since the machine will not be able to transmit to the network.

4. Enable remote administration via OpenSSH by selecting the Sharing preferences. Click on the Services tab. Place a checkmark next to Remote Login.

Now install an antivirus package (e.g., Virex, Norton Antivirus, etc.).

The next step is to update the OS to the latest OS upgrades and security patches. Connect the machine to the network. Open the System Preferences. Go the Software Update preferences, and click on the Update Now button.

The Software Update application will connect to the update server at Apple Computers. Software Update will present a list of updates to various software packages along with the OS updates and security patches.

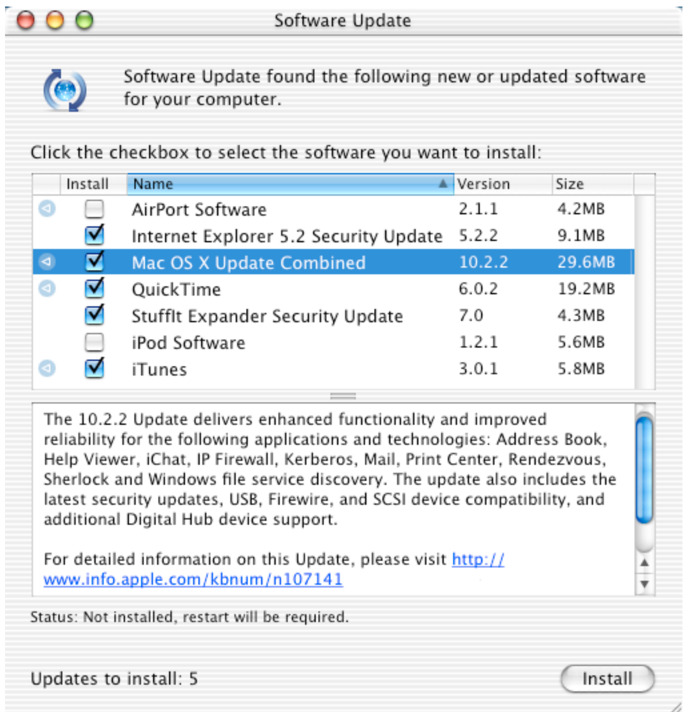


Figure 1-4, Software Update

If some of the updates (e.g., Airport Software) do not need to be installed, they may be hidden in the GUI. To hide any updates, select the updates that need to be hidden and select "Make Inactive" from the Update menu.

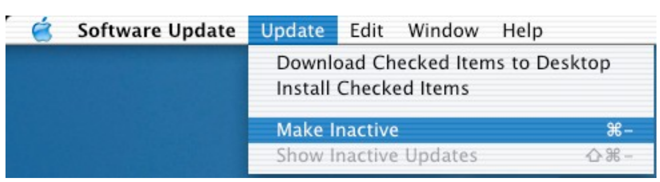


Figure 1-5, Inactivating software Updates

This is also a good time to get the latest anti-virus signatures and get any updates to the antivirus program.

If the machine has a single Ethernet card, then the maintenance of the operating system software, antivirus signatures and upgrades to the installed software must be brought to the machine on removable media (e.g., CDRW, Zip disk, etc.).

1.2.4 Post-Installation Housekeeping

1. We need to create a directory to hold the source code for software packages that we download and install.

   a. Open a Terminal window and enter

```
% sudo mkdir /usr/local/src
Password: <administrator passord>
```

   When prompted for a password, enter your password.

   b. Take ownership of the new directory.

```
% sudo chown smithrd:admin /usr/local/src
```

   where `smithrd` should be replace by your administrator account, which should be in the `admin` group.

2. Change the message that is displayed when a Terminal window is opened or a user logs in via OpenSSH to say something to discourage unauthorized access.

   a. Copy the existing file

```
% sudo cp /etc/motd /etc/motd.default
```

   b. Edit `/etc/motd` with your favorite text editor. I like using BBEdit 7.0 since it automatically understands the difference between Macintosh line ending and Unix line endings and doesn't munge them like TextEdit. To use BBEdit from the command line:

```
% sudo bbedit /etc/motd
```

   This will start BBEdit with `motd` in a window. The file can also be edited using the vi editor that is installed with the operating system.

```
% sudo vi /etc/motd
```

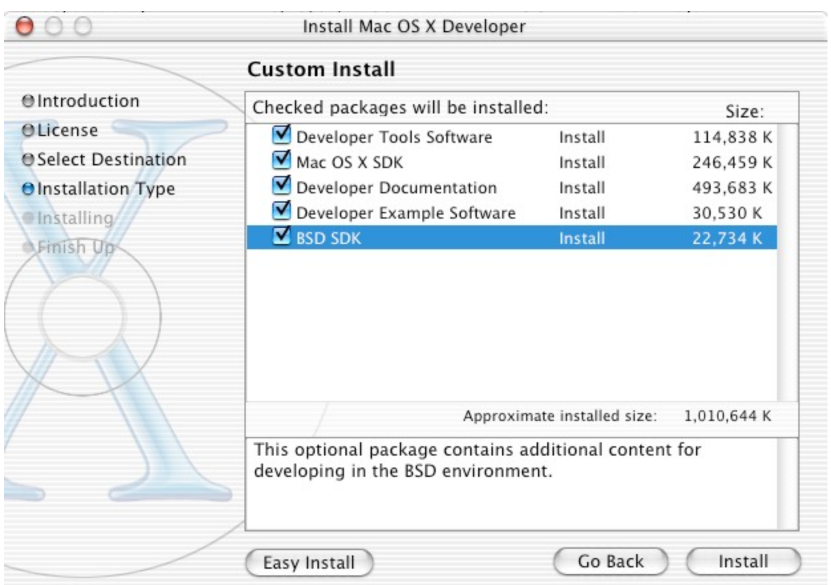


Figure 1-6, Developer Tools Installation

3. Install the Apple's July 2002 Mac OS X Developers Tools for Mac OS X (also known as DevTools) and the August 2002 and October 2002 Developers Tools updates. You will need to do a custom installation. This will give you the `gcc` v3.0 compiler that will be needed for compiling and installing some of the open source software including Snort.

   The Developers Tools should have come on a CD with Jaguar. If you did not get the DevTools CD, then you can get a disk image from Apple as a free download at http://www.apple.com/developer/. The Apple Developers Connection requires a free registration. You will also need to down load the DevTools updates from the same site. The installation of DevTools and the updates is a simple package install. When you install the DevTools, do a custom installation to get the BSD SDK.

1.3 Installing and basic configuration of the necessary software

1.3.1 Apache

Mac OS X installs Apache 1.3.26 as part of the default installation. It is controlled by the Sharing Preference panel and the `/etc/httpd/httpd.conf` file.

Open the System preferences and go to the Sharing icon to bring up the Sharing panel. Click on the Services tab. Click on the Lock Icon in the bottom left corner and authenticate if necessary to unlock the panel. Put a check in the check box next to Personal Web Sharing. This will start the web server. Test the web server by opening your browser and pointing it to `http://localhost`. It should now display the default Apache test web page similar to Figure 1-7.

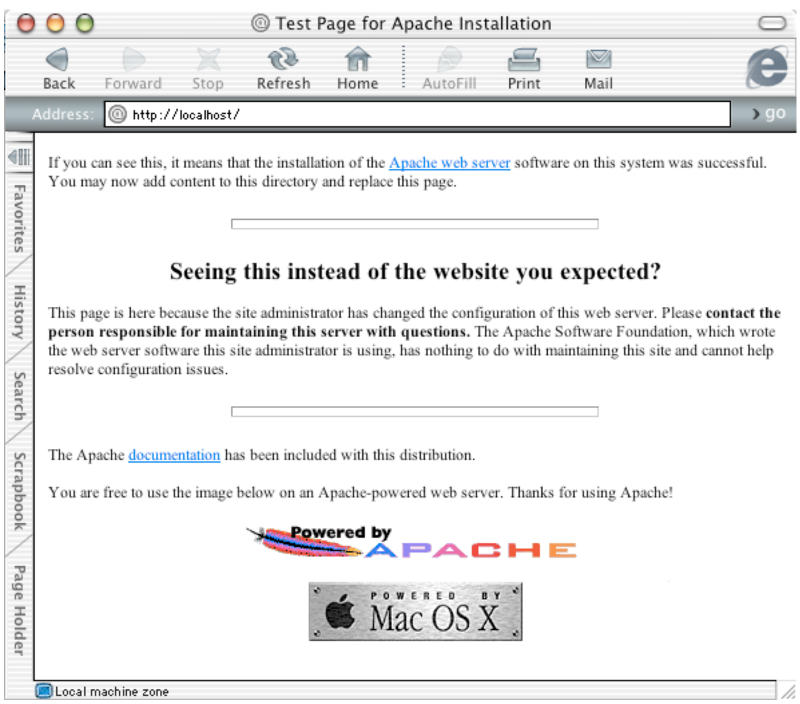


Figure 1-7, Apache Test Page

1.3.2 PHP

The Apple-provided Apache web server is built with support for PHP. However, the PHP (version 4.1.2) that comes with Jaguar is not built with GD support. We need support for GD to use with ACID. Thus, we will need to install a custom PHP library. The library could be compiled on the machine and used but it is quicker and easier to use the PHP library compiled by Marc Liyanage. His library is based on the latest version, 4.2.3, of PHP and includes support for GD and numerous other packages.

1.3.2.1 Basic Installation

1. Download it from http://www.entropy.com/software/macox/php. Normally the browser will place downloads to the Desktop. We need to copy it to the `src` directory and uncompress it.

   ```
   % cd /usr/local/src
   % cp ~/Desktop/libphp4.so.gz /usr/local/src
   % gunzip libphp4.so.gz
   ```

2. Save the existing PHP library

   ```
   % sudo mv /usr/libexec/httpd/libphp4.so /usr/libexec/httpd/libphp4.so.apple
   ```

3. Copy the new PHP library in to the correct location.

   ```
   % sudo cp libphp4.so /usr/libexec/httpd
   ```

1.3.2.2 Enable PHP Support in Apache

Now we need to enable the PHP support in the web server. Using your favorite editor, open the Appache configuration file, `/etc/httpd/httpd.conf` file.

`% sudo bbedit /etc/httpd/httpd.conf` or `% sudo vi /etc/httpd/httpd.conf`

Now locate the line that looks like

```
#LoadModule php4_module        libexec/httpd/libphp4.so
```

and delete the `#` at the beginning of the line. This enables Apache to load the PHP module when it starts. Next remove the `#` from the line that looks like

```
#AddModule mod_php4.c
```

Next, Apache needs to know to send files with the .php extension to the PHP modules that were just enabled. To do this we need to add the following lines to the `Document types` section of `httpd.conf` just after the
"`AddType application/x-tar .tgz` line."

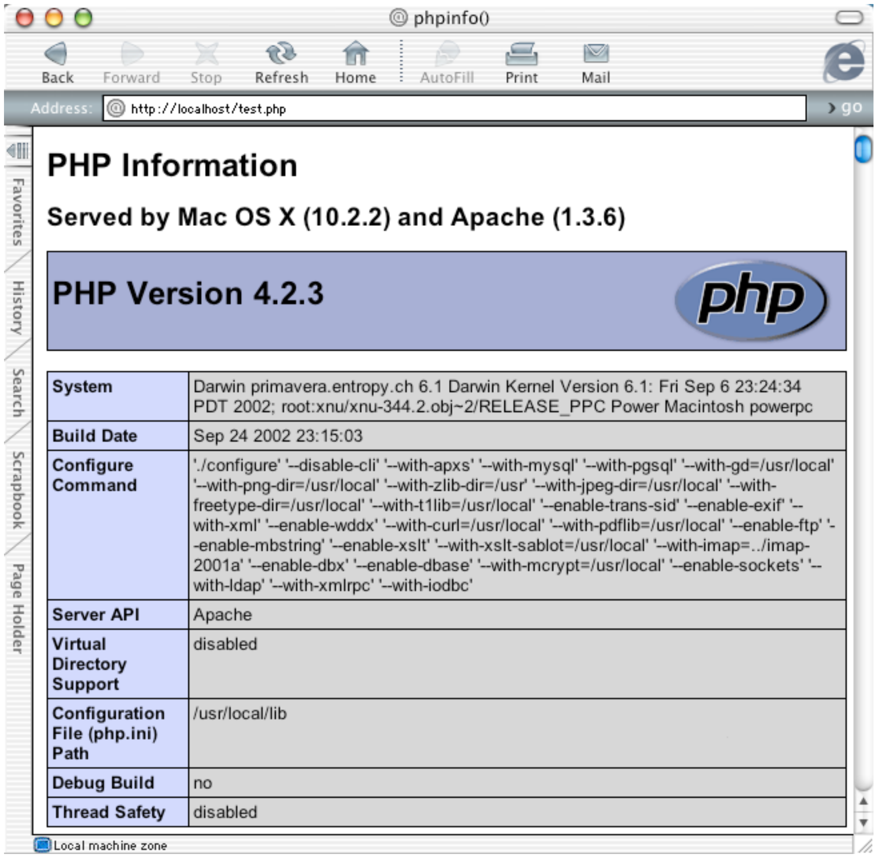
```
#
# Use PHP 4.x
#
AddType application/x-httpd-php       .php
AddType application/x-httpd-php-source .phps
```

This should configure the web server to serve the PHP correctly. To get the web server to read the new configuration stop and start it using the Sharing preference panel.

Test the web server by creating `test.php` in `/Library/WebServer/Documents`. Put the following in `test.php`:

```
<html><body>
<h1> PHP Information</h1>
<h2>Served By Mac OS X (Jaguar) and Apache (1.3.26) </h2>
<? phpinfo() ?>
</body></html>
```

Point your browser to http://localhost/test.php. You should see a page of information similar to Figure 1-8 that lists the version of PHP installed and plethora of other information on the web server configuration. Note that this file should be removed from the web server root directory before placing the machine in service since it provides a wealth of useful information to an attacker.



PHP Information

Served by Mac OS X (10.2.2) and Apache (1.3.6)

PHP Version 4.2.3

System	Darwin primavera.entropy.ch 6.1 Darwin Kernel Version 6.1: Fri Sep 6 23:24:34 PDT 2002; root:xnu/xnu-344.2.obj~2/RELEASE_PPC Power Macintosh powerpc
Build Date	Sep 24 2002 23:15:03
Configure Command	'./configure' '--disable-cli' '--with-apxs' '--with-mysql' '--with-pgsql' '--with-gd=/usr/local' '--with-png-dir=/usr/local' '--with-zlib-dir=/usr' '--with-jpeg-dir=/usr/local' '--with-freetype-dir=/usr/local' '--with-t1lib=/usr/local' '--enable-trans-sid' '--enable-exif' '--with-xml' '--enable-wddx' '--with-curl=/usr/local' '--with-pdflib=/usr/local' '--enable-ftp' '--enable-mbstring' '--enable-xslt' '--with-xslt-sablot=/usr/local' '--with-imap=../imap-2001a' '--enable-dbx' '--enable-dbase' '--with-mcrypt=/usr/local' '--enable-sockets' '--with-ldap' '--with-xmlrpc' '--with-iodbc'
Server API	Apache
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/usr/local/lib
Debug Build	no
Thread Safety	disabled

Figure 1-8, PHP Test: phpinfo()

1.3.3 ADOdb:

This is a PHP database extraction library used by ACID. The tarball can be found at http://php.weblogs.com/adodb. The ACID requires a version greater than 1.2 and the latest is version is 2.43.

1. Start by copying to the build directory and untar.

```
% cp ~/Desktop/adodb243.tar.gz /usr/local/src
% tar -xzf adodb243.tar.gz
```

2. Copy the ADOdb files to the web server.

```
% sudo mkdir /Library/WebServer/PHP
% sudo cp -R adodb /Library/WebServer/PHP/ADOdb
```

Note that the ADOdb/`doc` and ADOdb/`example` directories and their contents should be removed prior to placing the Snort box on line.

1.3.4 PHPlot

This is an optional package. PHPlot is a PHP graphics library used by ACID to create graphs. The PHPlot v 4.4.6 tarball can be obtained http://www/phplot.com.

1. Start by copying the tarball to the build directory and untar it.

```
% cp ~/Desktop/phplot-4.4.6.tar.gz /usr/local/src
% tar -xzf phplot-4.4.6.tar.gz
```

2. Copy the PHPlot files to the web server.

```
% cp -R phplot-4.4.6 /Library/WebServer/PHP/phplot
```

1.3.5 GD

This library is also an optional library needed for ACID graphics. GD provides JPEG and PNG graphic manipulation support to PHP and Perl. GD requires two libraries that are not installed with the OS: `libjpeg.a` and `libpng.a`. It also requires `zlib` but it is included in Mac OS X.

1.3.5.1 Install libjpeg.a

1. Download the source tarball, `jpegsrc.v6b.tar.gz`, from http://www.ijg.org. And copy it to the build directory.

```
% cp ~/Desktop/jpegsrc.v6b.tar.gz /usr/local/src
% tar -xzf jpegsrc.v6b.tar.gz
```

2. Change directory and configure the makefile. We also need to specify that the man pages go to the correct location when the library is installed.

```
% pushd jpeg-6b/
% ./configure
% make
% sudo make install mandir=/usr/local/share/man
% sudo make install-lib
% sudo ranlib /usr/local/lib/libjpeg.a
% popd
```

1.3.5.2 Install libpng.a

1. Download the source tarball, `libpng-1.2.5.tar.gz`, from http://www.libpng.org/. And copy it to the build directory.

```
% cp ~/Desktop/libpng-1.2.5.tar.gz /usr/local/src
% tar -xzf libpng-1.2.5.tar.gz
```

2. Change directory and configure the `makefile`.

```
% pushd libpng-1.2.5/
% cp scripts/makefile.macosx ./Makefile
% perl -i.pre -p -e `s/-current_version\$\(PNGVER\)//g` Makefile
```

3. We also need to specify where the `zlib` files are located.

```
% make ZLIBINC="/usr/lib" ZLIBLIB="/usr/lib"
% sudo make install
% sudo ranlib /usr/local/lib/libpng.a
% popd
```

1.3.5.3 Install GD

Now that all of the libraries are installed, we need to install the GD applications. We will use a slightly older version of GD that compiles easily on Jaguar.

1. Download the source tarball, `gd-1.8.4.tar.gz`, from http://www.boutell.com/gd/. Copy it to the build directory.

```
% cp ~/Desktop/gd-1.8.4.tar.gz /usr/local/src
% tar -xzf gd-1.8.4.tar.gz
```

2. Change to the `gd-1.8.4` directory.

```
% pushd gd-1.8.4/
```

3. The `Makefile` will need to be modified since we don't have the X Window system and freetype installed. Edit the Makefile by changing

```
INCLUDEDIRS=-I. -I/usr/include/freetype2 -I/usr/include/X11
    -I/usr/X11R6/include/X11 -I/usr/local/include
```

to

```
INCLUDEDIRS=-I. -I/usr/local/include
```

and

```
LIBDIRS=-L. -L/usr/local/lib -L/usr/lib/X11 -L/usr/X11R6/lib
```

to

```
LIBDIRS=-L. -L/usr/local/lib
```

4. Compile specifying the compiler and install.

```
make
sudo make install
sudo ranlib /usr/local/lob/libgd.a
```

5. You will get a message back that says

```
ranlib: file: /usr/local/lib/libgd.a(gdcache.o) has no symbols
```

but the error can be ignored.

1.3.6 MySQL

MySQL will be the database server used to store the Snort alerts. The quickest way to get MySQL up and running is to use a Mac OS X package installation. I will use Marc Liyanage's package that is based on based on MySQL version 3.23.52.

1.3.6.1 Installation

Download the package from http://www2.entropy.ch/download/mysql-3.23.52-jaguar.pkg.tar.gz and also download http://www2.entropy.ch/download/mysql-startupitem.pkg.tar.gz while you are there. The second package will be used to start MySQL server automatically.

The browser and Stuffit Expander should unstuff the .pkg file the directory where downloads are sent. This is usually your Desktop. Double-click on the `mysql-3.23.52.pkg` file to install MySQL server.

Jaguar creates a hidden "mysql" user account when the system is installed so that is one step we will not need to perform. Now we need to configure the database server.

1.3.6.2 Database Server configuration

1. Start the Terminal application to begin the configuring of the MySQL server. Change to the mysql directory.

```
% cd /usr/local/mysql
```

2. Install and initialize the database.

```
% sudo ./scripts/mysql_install_db
Password: <admin password>
Preparing db table
Preparing host table
Preparing user table
Preparing func table
Preparing tables_priv table
Preparing columns_priv table
Installing all prepared tables
021104 20:40:19  ./bin/mysqld: Shutdown Complete


To start mysqld at boot time you have to copy support-files/mysql.server
to the right place for your system


PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !
This is done with:
./bin/mysqladmin -u root  password 'new-password'
./bin/mysqladmin -u root -h LOCALHOST  password 'new-password'
See the manual for more instructions.


You can start the MySQL daemon with:
cd . ; ./bin/safe_mysqld &


You can test the MySQL daemon with the benchmarks in the 'sql-bench' directory:
```

```
cd sql-bench ; run-all-tests

Please report any problems with the ./bin/mysqlbug script!

The latest information about MySQL is available on the web at
http://www.mysql.com
Support MySQL by buying support/licenses at https://order.mysql.com
```

3. Change ownership of the mysql directory to the “mysql” user.

```
% sudo chown -R mysql /usr/local/mysql/*
```

4. Starting the server in the background.

```
% sudo ./bin/safe_mysqld --user=mysql &
[1] 751
Starting mysqld daemon with databases from /usr/local/mysql/data
```

5. Test the installation of MySQL.

```
% /usr/local/bin/mysql test
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 1 to server version: 3.23.52-entropy.ch

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

6. Exit from the server.

```
mysql> exit
Bye
```

7. Change the root password of the server.

```
% ./bin/mysqladmin -u root password 'admin-mysql'
```

See section 4.3 of the MySQL Reference Manual for more information on securing MySQL.

MySQL needs to start up automatically so we need to install the second package, `startupitem.pkg`. This will put a StartupItem named “MySQL” into `/Library/StartupItems`.

1.3.7 Snort:

Snort is the heart of this system. We will compile it with support for the output module for MySQL. Once Snort is built we will configure it to log the MySQL database and start automatically upon system boot. Download the latest stable version of Snort from http://www.snort.org/dl/. I will use `snort-1.9.0.tag.gz`.

1.3.7.1 Installation:

1. Open a Terminal window and copy the tarball.

```
% cp ~/Desktop/snort-1.9.0.tar.gz /usr/local/src
```

2. Untar and gunzip the snort source.

```
% cd /usr/local/src
% tar -xzf snort-1.9.0.tar.gz
```

3. Create a symbolic link to the directory created by un-taring the tarball and then change to that directory.

```
% sudo cd ln -s snort-1.9.0 /usr/local/snort
% cd snort
```

4. Configure the Makefile for Snort to use MySQL.

```
./configure -with-mysql=/usr/local/mysql
```

5. Compile and install by typing:

```
% make
% sudo make install
```

6. This will put the snort binary in `/usr/local/bin`

```
% which snort
/usr/local/bin/snort
```

7. Create the log file directory and change ownership.

```
% sudo mkdir /var/log/snort
% sudo chown snortman:snortman /var/log/snort
```

1.3.7.2 Adding a user for Snort

Snort has the ability to run as a normal user, so if there is ever an exploit and someone uses Snort to take over your box, they won't own the entire system. This also means that your database's root password isn't sitting in a clear-text file (`snort.conf`). We are going to add a hidden user, one that doesn't show up in the list of users on the logon screen.

To add a hidden user, open NetInfo Manger from the Utilities folder. Authenticate as an administrator by clicking on the lock at the bottom left-hand corner of the window. You will then be required to enter your password in the Authenticate window.

Now, we will add the hidden snort user. Click on the users entry in the middle column in the upper half of the window. This will put a list of users in the Third column. Click on the mysql user. Duplicate the mysql user entry by using the Duplicate command from the Edit menu. This will create a user account named "mysql copy." Now rename the copy by double clicking on the name, e.g., "mysql copy," in the lower half of the window. Enter snortman for the new name and hit enter this should take you to the entry below name. Use the following information, see Figure 1-9, to update the snortman user account:

name:	snortman
realname:	Snort User
shell:	/dev/null
home:	/dev/null
uid:	76
gid:	76
_writers_passwd:	0

Don't change the password entry. When you are done, save the changes by selecting Save from the file menu. You will be asked to confirm you want to save the changes.

Next we need to make the snortman group. Using the same procedure only with the mysql group.

name:	snortman
gid:	76

Again, don't change the password entry. When you are done, save the changes by selecting Save from the file menu. You will be asked to confirm you want to save the changes. Figure 1-10 shows the snotrman group final setup. It's a good idea to use this user exclusively to run Snort so it doesn't need a shell.

Figure 1-9, Snort User Information

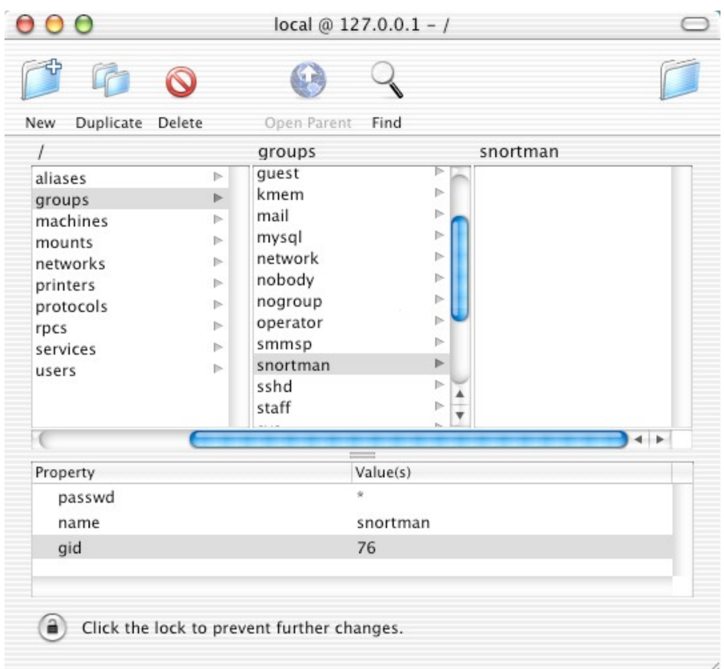


Figure 1-10, Snort User Group Information

1.3.7.3 Configure Snort

1. Restrict the permissions on the configuration file.

```
chmod 644 /usr.local.snort/etc/snort.conf
```

2. Modify the Snort configuration file. Edit `/usr/local/snort/etc/snort.conf` in a text editor. (**Note:** This is a basic installation guide and not a tutorial on the configuration of Snort. A much better place to get that information is the Snort documentation (http://www.snort.org/docs/SnortUsersManual.pdf).)
    a. Check that the RULE_PATH variable is set to

    ```
    var RULE_PATH ../rules
    ```

    If not, change it to point to the correct directory (either relative or complete path).

    ```
    var RULE_PATH /usr/local/snort/rules.
    ```

    b. Change the following in the database section:

    ```
    output database: log, mysql, user=snortman password=snortman
        dbname=snort host=localhost
    ```

3. Create the snort database and configure MySQL for use with Snort and ACID.
    a. Log in to MySQL and create the snort database

    ```
    % mysql -u root -p
    Enter password:
    Welcome to the MySQL monitor. Commands end with ; or \g.
    Your MySQL connection id is 4 to server version: 3.23.52-entropy.ch

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

    % CREATE DATABASE snort;
    ```

    b. Set permissions on the snort database for the user accounts that will be created for Snort and ACID.

    ```
    mysql> grant INSERT,SELECT on snort.* to snortman@localhost;
    Query OK, 0 rows affected (0.00 sec)
    ```

    c. Set the required permissions for the ACID user account.

    ```
    mysql> grant CREATE,INSERT,SELECT,UPDATE,DELETE on snort.* to acid@localhost;
    Query OK, 0 rows affected (0.00 sec)
    ```

    d. Set the password for the Snort and ACID user accounts.

    ```
    mysql> SET PASSWORD FOR snortman@localhost=PASSWORD('snortman');
    Query OK, 0 rows affected (0.00 sec)

    mysql> SET PASSWORD FOR acid@localhost=PASSWORD('acidman');
    Query OK, 0 rows affected (0.00 sec)
    ```

    e. Exit MySQL.

    ```
    mysql> exit
    Bye
    ```

    f. Create the tables in the snort database using the script supplied with Snort.

    ```
    % mysql -u root -p < /usr/local/snort/contrib/create_mysql snort
    Enter password:
    ```

    g. Log back into MySQL and check that the tables were created correctly.

    ```
    % mysql -u root -p snort
    Enter password:
    Welcome to the MySQL monitor. Commands end with ; or \g.
    Your MySQL connection id is 11 to server version: 3.23.52-entropy.ch

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

    Database changed
    mysql> show tables;
    +------------------+
    ```

```
| Tables_in_snort  |
+------------------+
| data             |
| detail           |
| encoding         |
| event            |
| icmphdr          |
| iphdr            |
| opt              |
| reference        |
| reference_system |
| schema           |
| sensor           |
| sig_class        |
| sig_reference    |
| signature        |
| tcphdr           |
| udphdr           |
+------------------+
16 rows in set (0.00 sec)

mysql> exit
Bye
```

4. Setup Snort to start up when the machine boots. We create a Snort StartupItem by copying the MySQL StartupItem and modifying it.

   a. Duplicate the `MySQL` StartupItem.

   ```
   % sudo cp -R /Library/StartupItems/MySql /Library/StartupItems/Snort
   ```

   b. Change to the `Snort` StartupItem Directory and rename the main file.

   ```
   % cd /Library/StartupItems/Snort/
   % sudo mv MySQL Snort
   ```

   c. Replace the contents of `Snort`, the main file in the StartupItem with the following

   ```
   #!/bin/sh
   . /etc/rc.common

   ##
   # Start up the Snort Intrusion Detection System
   #   on Mac OS X /Darwin
   #
   # History
   # -------
   #
   # 2002-12-18   RD Smith <rdsmith@mac.com>
   #              First Version. Adapted from the MySQL
   #              StartupItem created by Marc Liyanage
   #              <liyanange@access.ch>
   #

   if [ -x /usr/local/bin/snort ]; then

       ConsoleMessage "Starting Snort"

       /usr/local/bin/snort -c /usr/local/snort/etc/snort.conf \
         -i en0 -u snortman -g snortman -D > /dev/null &

   fi
   #
   ```

   d. Remove some of the unnecessary localization files.

   ```
   % cd Resources/

   % rm -R Dutch.lproj/ French.lproj/ German.lproj/ Italian.lproj/ Japanese.lproj/
   Spanish.lproj/
   ```

e. Edit the remaining English localization strings. Replace the contents of Snort/Resources/`English.lproj/Localizable.strings` with

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist SYSTEM
"file://localhost/System/Library/DTDs/PropertyList.dtd">
<plist version="0.9">
<dict>
        <key>Starting Snort IDS</key>
        <string>Starting Snort IDS</string>
</dict>
</plist>
```

f. Replace the contents of the `Snort/StartParameters.plist` with

```
{
  Description   = "Snort Network Intrusion Detection System";
  Provides      = ("Snort");
  Requires      = ("Resolver");
  Preference    = "Late";
  Messages =
  {
    start = "Starting Snort IDS";
    stop  = "Stopping Snort IDS";
  };
}
```

1.3.8 ACID

ACID is the Analysis Console for Intrusion Databases. It works with Snort with the Snort output module. The ACID tarball is downloaded as part of the Snort tarball. The tarball is located in the `contrib` directory. Now that we have installed all of the support packages, we will now install ACID itself.

1.3.8.1 Installation

1. Start by copying the tarball to the `build` directory and untar it.

```
% cp ../snort/contrib/ACID-0.9.6b21.tar.gz /usr/local/src
% tar -xzf ACID-0.9.6b21.tar.gz
```

2. Copy the files to the web server. I will copy the files to the PHP directory and then create a symbolic link in the Documents directory.

```
% cp -R -p acid /Library/WebServer/PHP/acid
% cd /Library/WebServer/Documents
% ln -s ../PHP/acid acid
```

3. Configuring ACID for the local setup. Open `/Library/WebServer/PHP/acid_conf.php` with a text editor.

   a. Change the path of the database interface to the full path of ADOdb,

   ```
   $DBLib_path = /Library/WebServer/PHP/ADOdb.
   ```

   b. Change the Alert DB connection parameters:

   ```
   $alert_dbname =”snort”;
   $alert_host = “localhost”;
   $alert_port = “3306”;
   $alert_user = “acid”;
   $alert_password = “acidman”;
   ```

   c. Change the path of the charting package to the full path of PHPlot,

   ```
   $ChartLib_path /Library/WebServer/PHP/phplot
   ```

   d. Add the path of the spp_portscan logs

   ```
   $portscan_file = "/var/log/snort/portscan.log"
   ```

e. Save `acid_conf.php`.

4. Change the permissions on the ACID configuration file.

```
% chmod 644 /Library/WebServer/PHP/acid/acid_conf.php
```

5. Open `acid_db_setup.php` in a browser window and follow the directions there to finish configuring database support. This will add four tables to the snort database.

1.3.8.2 Adding a user for ACID

We will use the same procedure that was used to create the hidden snortman user. To add the hidden "acid" user, open NetInfo Manger from the Utilities folder. Authenticate as an administrator by clicking on the lock at the bottom left-hand corner of the window. You will then be required to enter your password in the Authenticate window.

Now, we will add the hidden ACID user. Click on the users entry in the middle column in the upper half of the window. This will put a list of users in the Third column. Click on the mysql user. Duplicate the mysql user entry by suing the Duplicate command form the Edit menu. This will create a user account named "mysql copy". Now rename the copy by double clicking on the name, e.g., "mysql copy," in the lower half of the window. Enter "acid" for the new name and hit enter this should take you to the entry below name. Use the following infoformation to update the acid user account:

name:	acid
realname:	ACID User
shell:	/dev/null
home:	/dev/null
uid:	77
gid:	77
_writers_passwd:	0

Don't change the password entry. When you are done, save the changes by selecting Save from the file menu. You will be asked to confirm you want to save the changes. Next we need to make the acid group. Using the same procedure only with the mysql group.

name:	acid
gid:	77

Again, don't change the password entry. When you are done, save the changes by selecting Save from the file menu. You will be asked to confirm you want to save the changes. It's a good idea to use this user exclusively to run ACID so it doesn't need a shell.

1.3.8.3 ACID Test

Now if your web browser is pointed to http://localhost/acid/acid_main.php you should see something like Figure 1-11.

1.3.8.4 Hide the ACID display.

We need to change the default behavior of the Apache Web server. The goal is to prevent unauthorized access to ACID. The problem is Jaguar's built-in firewall GUI doesn't allow us to block inbound connections to port 80 when Personal Web Sharing is turned on. To get around the problem, we will modify the port and IP address that Apache binds to.

1. Using your favorite editor, open the Apache configuration file, `/etc/httpd/httpd.conf` file.

   `% sudo bbedit /etc/httpd/httpd.conf` or `% sudo vi /etc/httpd/httpd.conf`

2. In Section 1, find the Listen section and the line that looks like

```
#Listen 12.34.56.78:80
```

3. Remove the # from the beginning of the line and change the IP address to 127.0.0.1. Change the port to something above 1024. It should look like:

   ```
   Listen 127.0.0.1:14380
   ```

4. Save `httpd.conf`. In the System Preferences, stop and restart Personal Web Server.

Now if your web browser is pointed to http://localhost/acid/acid_main.php you should get an error connecting to the server. If you go to http://127.0.0.1:14380/acid/acid_main.php you should see ACID again.

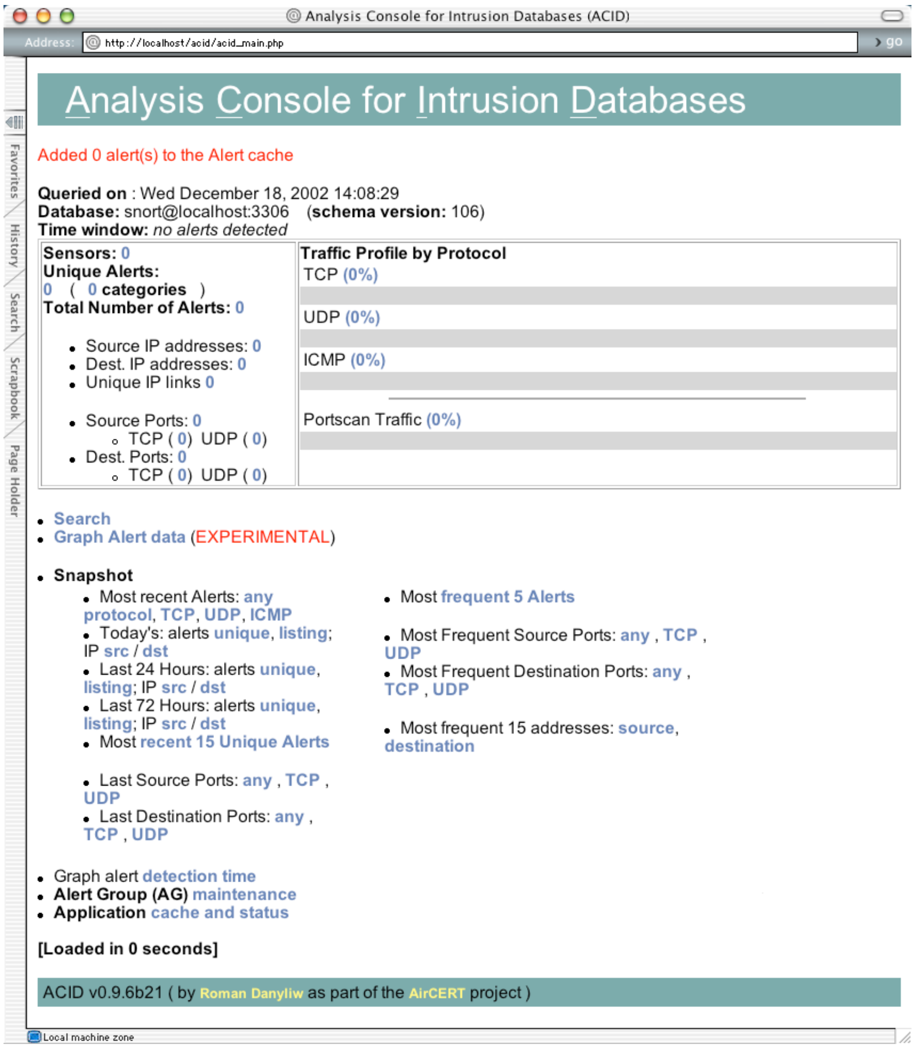


Figure 1-11, ACID Initial Screen Check

Another option to hide the display is to manually reconfigure the sniffing Ethernet card so that it doesn't transmit. The initial configuration of the Ethernet card can be checked with the following command.

```
% ifconfig en0
```

The resulting out put show status the en0 interface as

```
en0: flags=8963<UP,BROADCAST,SMART,RUNNING, SIMPLEX,MULTICAST> mtu 1500
        inet6 fe80::203:93ff:fe68:d08%en0 prefixlen 64 scopeid 0x4
        inet 10.26.75.32 netmask 0xfffff800 broadcast 10.26.79.255
        ether 00:03:93:de:ad:01
        media: autoselect (100baseTX <full-duplex>) status: active
        supported media: none autoselect 10baseT/UTP <half-duplex> 10baseT/UTP <half-
duplex,hw-loopback> 10baseT/UTP <full-duplex> 10baseT/UTP <full-duplex,hw-loopback>
```

```
100baseTX <half-duplex> 100baseTX <half-duplex,hw-loopback> 100baseTX <full-duplex>
100baseTX <full-duplex,hw-loopback>
```

Reconfigure the Ethernet card to “receive only” with the following command.

```
% sudo ipconfig set en0 NONE
```

This may take a few seconds. Check that the Ethernet card was reconfigured.

```
% ifconfig en0

en0: flags=8963<UP,BROADCAST,SMART,RUNNING, SIMPLEX,MULTICAST> mtu 1500
        tunnel inet  -->
        ether 00:03:93:de:ad:01
        media: autoselect (100baseTX <full-duplex>) status: active
        supported media: none autoselect 10baseT/UTP <half-duplex> 10baseT/UTP <half-
duplex,hw-loopback> 10baseT/UTP <full-duplex> 10baseT/UTP <full-duplex,hw-loopback>
100baseTX <half-duplex> 100baseTX <half-duplex,hw-loopback> 100baseTX <full-duplex>
100baseTX <full-duplex,hw-loopback>
```

The reconfiguration will only last until the next reboot. This could be added to the Snort StartupItem script to put the Ethernet card in “receive only” mode upon reboot.

1.4 Testing your box

Reboot the machine. As it reboots, hold down the command and V keys until you see a black screen with white text. Now the machine will boot in verbose mode and show all of the console messages as it boots. Watch the messages as they scroll up the screen. You should see these messages:

“Starting Snort”
“Snort Started”
“Starting MySQL”

Log in as your administrative user. Open a Terminal Window and type

```
% ps -caux | grep mysqld

mysql    362   0.0  1.0    12308   1324  ??  S   8:15PM   0:00.10 mysqld

% ps -caux | grep snort

snortman 312   0.0  4.7    51844   6172  ??  S   8:14PM   0:00.48 snort

% ps -caux | grep httpd

root     406   0.0  2.3    23160   2960  ??  Ss  8:15PM   0:00.45 httpd
www      411   0.0  0.2    23160    280  ??  S   8:15PM   0:00.00 httpd
```

In the process lists generated, you should be able to find MySQL, Snort and httpd (Apache) running.

If the Ethernet card was not put into receive-only mode, then you can check the status of the Ethernet card by typing

```
% ifconfig en0
```

The resulting output shows status of the en0 interface in promiscuous mode as shown below.

```
en0: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
        inet6 fe80::203:93ff:fe68:d08%en0 prefixlen 64 scopeid 0x4
        inet 10.26.75.32 netmask 0xfffff800 broadcast 10.26.79.255
        ether 00:03:93:de:ad:01
        media: autoselect (100baseTX <full-duplex>) status: active
        supported media: none autoselect 10baseT/UTP <half-duplex> 10baseT/UTP <half-
duplex,hw-loopback> 10baseT/UTP <full-duplex> 10baseT/UTP <full-duplex,hw-loopback>
100baseTX <half-duplex> 100baseTX <half-duplex,hw-loopback> 100baseTX <full-duplex>
100baseTX <full-duplex,hw-loopback>
```

You can also check the open ports by using the `netstat` command. This will list the status all the open TCP/IP ports on the machine. You should see the ports for your web server, 14380 (http), and MySQL, 3306, listed as LISTEN.

```
% netstat -an -f inet
Active Internet connections (including servers)
Proto   Recv-Q Send-Q    Local Address         Foreign Address       (state)
tcp4         0      0    127.0.0.1.1033        127.0.0.1.1011        ESTABLISHED
tcp4         0      0    127.0.0.1.1011        127.0.0.1.1033        ESTABLISHED
```

```
tcp4    0    0    127.0.0.1.14380    *.*                LISTEN
tcp4    0    0    *.*                *.*                CLOSED
tcp4    0    0    *.*                *.*                CLOSED
tcp4    0    0    127.0.0.1.1033     127.0.0.1.1019     ESTABLISHED
tcp4    0    0    127.0.0.1.1019     127.0.0.1.1033     ESTABLISHED
tcp4    0    0    127.0.0.1.631      *.*                LISTEN
tcp4    0    0    *.3306             *.*                LISTEN
tcp4    0    0    *.22               *.*                LISTEN
tcp4    0    0    127.0.0.1.1033     *.*                LISTEN
udp4    0    0    *.5353             *.*
udp4    0    0    *.53               *.*
udp4    0    0    *.*                *.*
udp4    0    0    *.*                *.*
udp4    0    0    *.631              *.*
udp4    0    0    127.0.0.1.49156    127.0.0.1.1023
udp4    0    0    127.0.0.1.49155    127.0.0.1.1023
udp4    0    0    *.1023             *.*
udp4    0    0    10.26.75.32.123    *.*
udp4    0    0    127.0.0.1.123      *.*
udp4    0    0    *.123              *.*
udp4    0    0    127.0.0.1.1033     *.*
udp4    0    0    *.514              *.*
udp4    0    0    *.68               *.*
```

Some of the other open ports are 22/tcp (sshd), 631/udp (cupsd), 514/udp (syslog), 123/udp (ntp), and 53 (dns).

The final test should be to scan the machine externally. This will server two purposes: test the Snort/MySQL/ACID operation and to test the hardening of the machine security. To scan the machine, I used nmap (v3.00) from http://www.insecure.org. Nmap compiles and installs using the normal "`./configure; make; sudo make install`" procedure. NmapFE for OSX, a Mac OS X GUI front end to nmap developed by Mathew Rothenberg, is available from http://faktory.org/m/software/nmap.

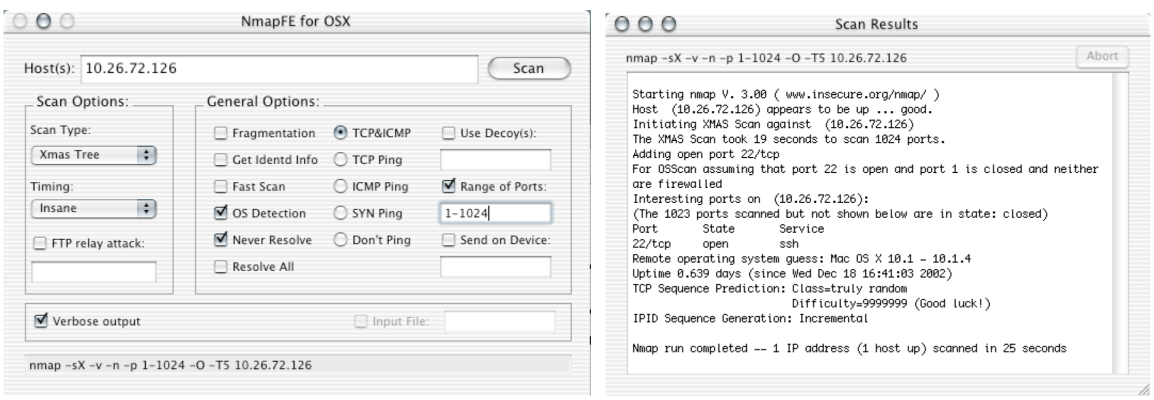


Figure 1-12, Configuration and Scan Results of NMapFE for OSX Scan of Snort Box

For the scanning, I used a standard Ethernet cable to connect the Snort box to the network. The Ethernet card was in the normal configuration (not the "receive only" mode). The setup and results of a Christmas Tree scan of the Snort box with nmap are shown in Figure 1-12. The scan shows one open TCP port: 22 (sshd). Nmap identified the operating system as "`Remote operating system guess: Mac OS X 10.1 - 10.1.4`." I scanned the ports 14300/tcp to 14400/tcp using NmapFE and found no open ports.

The results of a UDP scan of shows several ports open:

```
Port          State         Service
53/udp        open          domain
123/udp       open          ntp
514/udp       open          syslog
1023/udp      open          unknown
```

These ports should be blocked by some means before the machine is placed on the network.

Nessus (http://www.nessus.com/) is an excellent free vulnerability scanner that should also be used top test the Snort box. Nessus runs as a client-server application. The client, `nessus`, controls the scans that the server, `nessusd`, performs. The `nessusd` server will compile and run on Mac OS X. The command line client will also compile and run. For the GUI version of the client, a number of other packages need to be installed prior to compiling Nessus. The Nessus client running in XDarwin is shown in Figure 1-13. The communications between the server and the client can be secured through the SSL with Nessus generated digital certificates used to authenticate both the server and the client. The results of the vulnerability scan can be viewed in the client GUI or may be saved to HTML and viewed with a web browser.

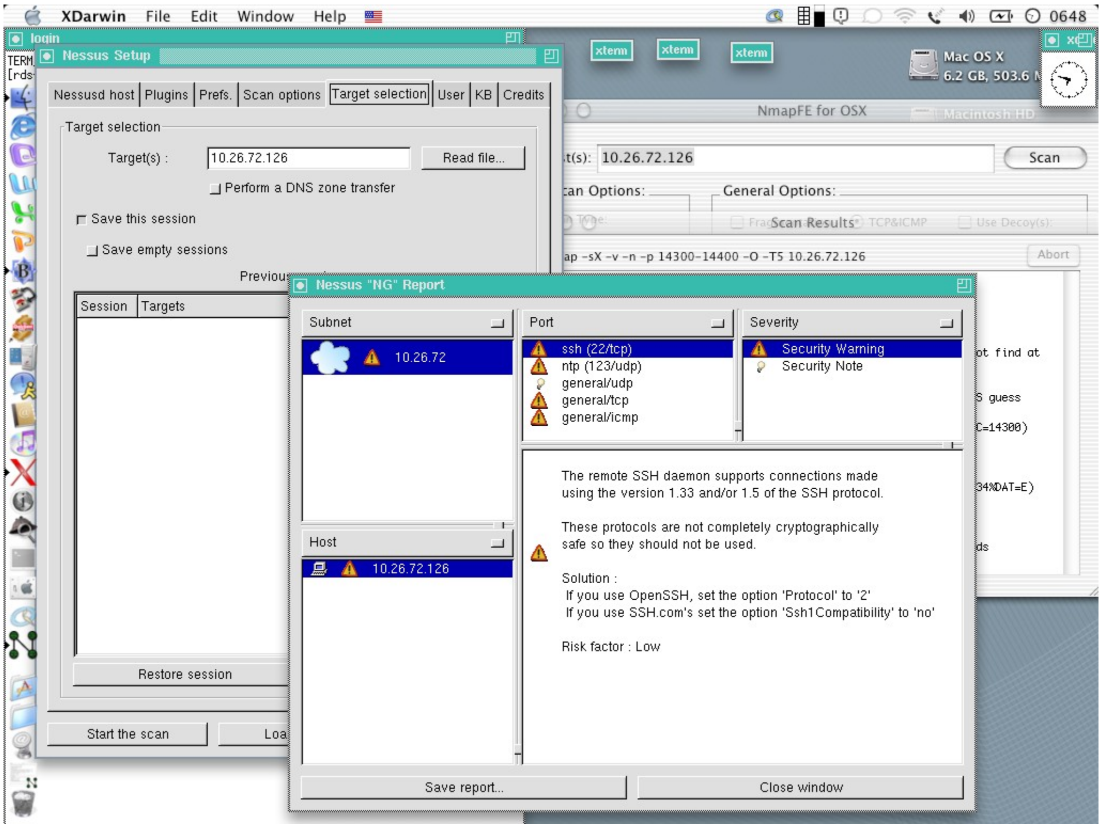


Figure 1-13, Nessus

The Nessus scan results list four security warnings and five security notes listed in Table 1-1.

Port	Warning	Solution
ssh (22/tcp)	The remote SSH daemon supports connections made using the version 1.33 and/or 1.5 of the SSH protocol. (Risk factor: Low)	Turn off Remote Login in the Sharing preference pane. Set the option 'Protocol' to '2' in `/etc/ssh-config`.
general/icmp	The remote host answers to an ICMP timestamp request. (Risk factor: Low)	Configure `ipfw` to filter out the ICMP timestamp requests (13), and the outgoing ICMP timestamp replies (14).
ntp (123/udp)	An NTP server is running on the remote host. Make sure that you are running the latest version of your NTP server. Some versions have been found out to be vulnerable to buffer overflows.	Configure `ipfw` to block traffic to port 123/udp.
general/tcp	The remote host uses non-random IP IDs, that is, it is possible to	None. Requires a patch from Apple.

	predict the next value of the `ip_id` field of the ip packets sent by this host. (Risk factor: Low)	

Table 1-1, Nessus Vulnerability Scan Results

1.5 Conclusion

This paper has shown you how to install Snort, MySQL, ACID and supporting software. The resulting machine is reasonably secure. However, as the author of ACID noted, there is no input checking in ACID so it needs to be protected. For remote access to ACID, the configuration of Apache could be modified further to permit access from specific hosts. This can be configured my modifying the authentication and access controls in `httpd.conf` and using `.htaccess` files (See the Apache Documentation for Authentication, Authorization, and Access Control and .htaccess How-To articles.) Also the installation of MySQL, could be hardened further against malicious insiders. (See the MySQL documentation.)

There are several protection layers or methods that could be added to secure the box if it is placed on an unprotected network segment. A "receive-only" Ethernet cable or a network tap similar to Finisar's SAN/LAN In-line Tap, could be used to connect to the monitored network. Another layer is a more secure configuration of the built-in firewall, `ipfw`. The configuration of the firewall can be done manually or with a GUI. A good GUI front-end for ipfw is Brickhouse by Brian Hill. The FreeBSD Project has several how-to articles on the configuration of `ipfw`.

References:

How to setup and secure Snort, MySQL and Acid on FreeBSD 4.6 Release, Keith Tokash (July 2002)
http://www.snort.org/docs/FreeBSD46RELEASE-Snort-MySQLVer1-2.pdf

How to make a sniffing (receive only) UTP cable, Sam Ng (August 9, 2001)
http://www.geocities.com/samngms/sniffing_cable/

"Receive only" sniffing cable, Iron Comet Consulting (August 24, 2002)
http://www.ironcomet.com/sniffer.shtml)

Apache HTTP Server Version 1.3 Documentation, The Apache HTTP Server Documentation Project, (2002)
http://httpd.apache.org/docs/

Apache Web-Serving with Mac OS X (series), Kevin Hemenway (various)
http://www.macdevcenter.com/pub/ct/49
http://www.macdevcenter.com/pub/a/mac/2001/12/07/apache.html
http://www.oreillynet.com/a/mac/2001/12/14/apache_two.html
http://www.oreillynet.com/a/mac/2002/01/04/apache_macosx_pt3.html
http://www.macdevcenter.com/pub/a/mac/2002/01/29/apache_macosx_four.html
http://www.macdevcenter.com/pub/a/mac/2002/03/08/apache_mac_5.html
http://www.macdevcenter.com/pub/a/mac/2002/04/23/apache_six.html

Marc Liyanage - Software - Mac OS X Packages – MySQL, Marc Liyanage,
http://www.entropy.ch/software/macosx/mysql/

MySQL Reference Manual
http://www.mysql.com/documentation/mysql/bychapter/manual_MySQL_Database_Administration.html#User_Account_Management
http://www.mysql.com/documentation/mysql/bychapter/manual_MySQL_Database_Administration.html - Privilege_system

Building GD on Mac OS X 10.1, Scott Anguish (November 15, 2002)
http://www.stepwise.com/Articles/Workbench/2001-06-12.01.html

Creating SystemStarter Starup Item Bundles HOWTO, Kevin Van Vechten. (May 7, 2002)
http://developer.apple.com/techpubs/macosx/Darwin/howto/system_starter_howto/system_starter_howto.html

ACID: Installation and Configuration, Roman Danyliw. (October 9, 2002)
http://www.andrew.cmu.edu/~rdanyliw/snort/acid_config.html

Hidden Sniffer HowTo, Rob Thomas (6 March 2000)
http://www.tlsecurity.net/Textware/Security/sniffer-trick.txt

Nmap, v3.00, Fyodor (July 28, 2002)
http://www.insecure.org

NmapFe for OSX, v0.5b2, Mathew Rothenberg, (August 21, 2002)
http://faktory.org/m/software/nmap

Nessus 1.2.7, Renaud Deraison (December 17, 2002)
http://www.nessus.com/

SAN/LAN Inline Tap, Finisar Corporation
http://www.finisar.com/product/product.home.php?product_category_id=98

Brickhouse, v1.1b6, Brian Hill (October 7, 2001)
http://personalpages.tds.net/~brian_hill/brickhouse.html

2 Part 2 -Network Detects

2.1 Detect 1: Web Server Directory Transversal Exploit

2.1.1 Source of Trace.

The alerts are from raw logs obtained from http://www.incidents.org/logs/Raw/. The log file used was 2002.6.15. The network layout of this network is unknown. Based on the logs, it appears that the target of the attack, the server at IP address 46.5.180.133, is a web server based on the fact that it responded to a connection to TCP port 80. The dump of the packet that caused the Snort alert is given below.

```
% tcpdump -nX -r 2002.6.15 "host 130.205.110.105"

07:19:21.124488 130.205.110.105.3937 > 46.5.180.133.80: P 80977326:80977385(59) ack
2095631126 win 17520 (DF)
0x0000   4500 0063 fbcb 4000 7406 3d0e 82cd 6e69        E..c..@.t.=...ni
0x0010   2e05 b485 0f61 0050 04d3 9dae 7ce8 cb16        .....a.P....|...
0x0020   5018 4470 465c 0000 4745 5420 2f73 6372        P.DpF\..GET./scr
0x0030   6970 7473 2f2e 2e25 3563 2535 632e 2e2f        ipts/..%5c%5c../
0x0040   7769 6e6e 742f 7379 7374 656d 3332 2f63        winnt/system32/c
0x0050   6d64 2e65 7865 3f2f 632b 6469 720d 0a69        md.exe?/c+dir..i
0x0060   720d 0a                                        r..
```

2.1.2 Detect was generated by:

Snort intrusion detection system (Version 1.9.0 (Build 209)) running on Mac OS X v10.2.2. The rules and `snort.conf` (`$Id: snort.conf,v 1.110.2.4 2002/11/17 04:40:07 cazz Exp $`) were downloaded from http://www.snort.org/. The rule set was used without modification.

The logs files were analyzed by Snort in a batch mode using the following command.

```
% sudo snort -A full -b -c rules/snort.conf -l logs/snort-alerts -r logs/raw_logs/2002.6.15
```

where the options are

`-A full`	Instucts Snort to use ASCII format using full alerts (default mode).
`-b`	Instructs Snort to use binary logging.
`-c <file>`	Instructs Snort to use the specified configuration file.
`-l (directory>`	Instructs snort to log alerts to the specified directory.
`-r <file>`	Instructs snort to take its input form the specified data file.

Once Snort was run on each raw log file, the logs were copied to a common directory.

The alerts from Snort were analyzed with SnortSnarf, v021111.1.

```
% ./snortsnarf.pl -d ../snortalerts -rulesdir ../rules -rulesfile ../rules/snort.conf
       -rulesscanonce -rs ../logs/snort/snort-2002-06-15.log ../logs/snort/snort-2002-06-
       16.log ../logs/snort/snort-2002-06-17.log ../logs/snort/snort-2002-06-18.log
```

where the options are

`-d <directory>`	Generate the HTML in the specified directory.
`-rulesdir <directory>`	Use the rules in the specified the specified directory.
`-rulesfiles <file>`	Instructs SnortSnarf to use the specified configuration file.
`-rulesscanonce`	Read the rules files once and retain them in memory.
`-rs`	List the alerts in reverse order in the results so that the most interesting alerts are on the top.

Tcpdump, v 3.71, was used for further analysis of the packets of interest.

The rule that alerted is in `web-iis.rules (# $Id: web-iis.rules,v 1.52.2.1 2002/11/17 04:40:09 cazz Exp $)`. The specific rule is:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"WEB-IIS cmd.exe access";
flow:to_server,established; content:"cmd.exe"; nocase; classtype:web-application-attack;
sid:1002;  rev:5;)
```

The rule is broken down into two parts: the Rule Header and the Rule Options. The Rule Header for this rule is interpreted by Snort as instructions to

`alert`	Generate and alert using the selected alert method
`tcp`	Analyze the TCP protocol
`$EXTERNAL_NET`	Match this source IP address in the packet to this value to satisfy the rule. The default `$EXTERNAL_NET` is `any`. Networks are specified by CIDR notation and the negation operator is "`!`".
`any`	Match this source port in the packet to this value to satisfy the rule. This rule specifies `any` port. Multiple continuous ports are specified with the range operator "`:`" and the negation operator is "`!`"
`->`	Direction of flow from the source to the destination. The only other option is "`<>`" which tells Snort to consider the address/port pairs in either source or destination orientation.
`$HTTP_SERVERS`	Match the destination IP address of the packet to this value to satisfy the rule. The default `$HTTP_SERVERS` is `$HOME_NET` which is `any` by default.
`$HTTP_PORTS`	The destination port of the packet to this value to satisfy the rule. The default value of `$HTTP_PORTS` is 80.

The Rule Options are

`msg:"WEB-IIS cmd.exe access";`	The message to print along with the packet dump or to an alert.
`flow:to_server,established;`	specifies that the alert triggers on client request from A to B or for an established connection. Used in conjunction with TCP stream reassembly.
`content:"cmd.exe";`	Match the string "cmd.exe" in the packet payload. This can be binary data or text.
`nocase;`	Match the content string with cse insensitivity
`classtype:web-application-attack;`	Categorizes the alert in to the web-application-attack class.
`sid:1002;`	The signature identification number
`rev:5;`	The revision number of the signature

2.1.3 Probability the source address was spoofed:

The packets are from established TCP connections to the web server, thus it is unlikely that the address is spoofed. As seen in the packet dump, the attacking machine (130.205.110.105) is acknowledging a packet from the machine under attack (46.5.180.133).

```
% tcpdump -nX -r 2002.6.15 "host 130.205.110.105"

07:19:21.124488 130.205.110.105.3937 > 46.5.180.133.80: P 80977326:80977385(59) ack
2095631126 win 17520 (DF)
```

ARIN lookup on the attacking machine IP address (130.205.110.105) gives the following information:

```
OrgName:    Thaumaturgy & Speculums Technology
OrgID:      TST

NetRange:   130.205.0.0 - 130.205.255.255
CIDR:       130.205.0.0/16
NetName:    WITTSEND
NetHandle:  NET-130-205-0-0-1
Parent:     NET-130-0-0-0-0
NetType:    Direct Assignment
NameServer: Z1.NS.NYC1.GLOBIX.NET
NameServer: Z1.NS.SJC1.GLOBIX.NET
NameServer: Z1.NS.LHR1.GLOBIX.NET
Comment:
RegDate:
Updated:    2001-10-04
```

```
TechHandle: MHW9-ARIN
TechName:   Warfield, Michael
TechPhone:  +1-770-985-6132
TechEmail:  mhw@wittsend.com
```

2.1.4 Description of attack:

This attack is part of a scan for web servers that are vulnerable to a directory transversal exploit. The tcpdump output of the packet shows the http GET command being sent to the web server was shown above.

A review of all packets captured from this source IP shows that this is a scan of the 46.5.180.0/24 subnet. The source ports of the packets increment by 1 for every target IP on the local subnet, as shown the tcpdump output below:

```
% tcpdump -n -r 2002.6.15 "host 130.205.110.105"
07:19:21.124488 130.205.110.105.3937 > 46.5.180.133.80: P 80977326:80977385(59) ack
        2095631126 win 17520 (DF)
07:19:21.124488 130.205.110.105.3938 > 46.5.180.134.80: P 81030630:81030689(59) ack
        2097180783 win 17520 (DF)
07:19:21.124488 130.205.110.105.3939 > 46.5.180.135.80: P 81087073:81087132(59) ack
        2088096245 win 17520 (DF)
07:19:21.134488 130.205.110.105.3957 > 46.5.180.153.80: P 81946892:81946951(59) ack
        2091789206 win 17520 (DF)
07:19:21.134488 130.205.110.105.3962 > 46.5.180.158.80: P 82191931:82191990(59) ack
        1048018881 win 17520 (DF)
07:19:21.144488 130.205.110.105.3949 > 46.5.180.145.80: P 81540501:81540560(59) ack
        3371690623 win 17520 (DF)
07:19:21.154488 130.205.110.105.3955 > 46.5.180.151.80: P 81859502:81859561(59) ack
        2091429905 win 17520 (DF)
07:19:21.524488 130.205.110.105.3937 > 46.5.180.133.80: P 59:117(58) ack 2921 win 0 [tos
        0x10]
07:19:22.024488 130.205.110.105.3957 > 46.5.180.153.80: P 59:117(58) ack 2921 win 0 [tos
        0x10]
07:19:28.194488 130.205.110.105.4054 > 46.5.180.250.80: P 88570441:88570500(59) ack 601480666
        win 17520 (DF)
```

The ports that are missing from the list above are most likely from machines that didn't respond to connection attempts on port 80/tcp.

Although there are many web servers that are vulnerable to directory transversal attacks, this appears to be an attempted attack on a Microsoft Internet Information Server (IIS) installation. This assessment is based on the fact that the attempt is looking for the `cmd.exe`, which indicates a Microsoft Windows machine and the attempt to access the `scripts` directory. The default installation of IIS has a `scripts` directory in the root web directory.

The Apache web server, v2.0 to 2.0.39 running on Windows, is also vulnerable to a directory transversal attack (CVE candidate CAN-2002-661 (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2002-661) and Apache Software Foundation (http://httpd.apache.org/info/security_bulletin_20020908a.txt). Apache, however, does not have a `scripts` directory in the default installation.

2.1.5 Attack mechanism:

The attack works by connecting to a vulnerable web server and sending the HTTP GET command with a request for a file outside the root web directory. A vulnerable web server will parse the file path before decoding the ASCII hex codes and will thus allow the directory transversal. In addition if the `scripts` directory is marked as executable and an executable is requested, the executable will be carried out.

In this case the `cmd.exe` executable in the `/scripts/..\\../winnt/system32` directory has been requested. The web server request passes commands to be executed on the web server if the directory traversal is successful. The results of these commands are passed back to the attacking machine as a web page.

Attackers use various means to obfuscate the directory traversal in order to get past the input validation of the web server. This can be seen in the ASCII dump of the packet shown above where the ASCII codes "`%5c`" is substituted into the HTTP GET command for the backslashes. Overlong UTF-8 encodings of "/" and "\" can also be used. This obfuscation technique is also used to try to avoid alerting IDS systems.

2.1.6 Correlations:

The attacking machine was reported conducting Code Red worm attacks on machines on the 129.105.X.X network on June 30, 2002 (http://www.mynetwatchman.com/LID.asp?IID=5748017) and machines on the 134.29.X.X network on July 30, 2002. (http://www.mynetwatchman.com/LID.asp?IID=5031511).

A Google.com search of the GIAC web site (http://www.google.com/search?hl=en&ie=ISO-8859-1&q=site%3Awww.giac.org+130.205.110.105) and the SANS web site (http://www.google.com/search?hl=en&lr=&ie=ISO-8859-1&q=site%3Awww.sans.org+130.205.110.105) did not return any hits for this IP address. A search for the 130.205.0.0/16 network returned three instances of machines but the references were not to related attacks (www.giac.org/practical/REUBEN_RUBIO_GCIA.doc, www.giac.org/practical/Robert_Nine_GCIA.doc, www.giac.org/practical/Akiva_Clark_GCIA.doc)

This type of attack is also the subject of several Microsoft Security Bulletins, MS-0057, MS00-078, and MS00-086. These bulletins discuss the file Permission Canonicalization, Web Server Folder Transversal, and Web Server File Request Parsing vulnerabilities. CERT CC discusses these vulnerabilities under CA-2001-12. It is also listed as CVE-0333 (http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0333)

Rain Forrest Puppy, http://www.wiretrip.net/rfp/p/doc.asp/i6/d57.htm, provided an analysis of the file request parsing vulnerability in IIS. Network Security Focus, http://www.nsfocus.com/English/homepage/sa_06.htm, provided the initial analysis of the web directory transversal vulnerability in IIS.

This type of attack is documented as arachNIDS IDS298 or IDS297 (http://www.whitehats.com/IDS/298 or http://www.whitehats.com/IDS/297).

The alerts could also be the work of Code Blue or Nimda worms. It doesn't appear to be the worms since the other attempts to reach `cmd.exe` using different encoded characters did not cause alerts. (See CERT CC CA-2001-26, http://www.cert.org/advisories/ca-2001-26.html and ISS X Force Security Alert http://bvlive01.iss.net/issEn/delivery/xforce/alertdetail.jsp?id=advise96.)

2.1.7 Evidence of active targeting:

The alerts appear to be a general scan of an entire network. Based on the number of packets seen in the alerts and the correlation of source ports to destination IP address.

Source Port	Last Octet of Destination IP
3937	133
Difference: 20	Difference: 20
3957	153
Difference: 97	Difference: 97
4054	250

This doesn't appear to be active targeting of a specific machine but it is a scan targeted at the 46.5.180.0/24 network. Due to the speed at which the packets were sent to the various machine on the network, it appears that an automated tool or script was used in the scan.

2.1.8 Severity:

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

Severity = (3 + 4) – (1 + 3) = 3

Each value is ranked on a scale from 1 (lowest) to 5 (highest).

The **Criticality** value is a measure of how critical the targeted system is to the network. This is a scan of the network including two web servers. Since the web server appears to be external to any firewall, the web servers are part of the corporate web presence. Without knowing the function of the web server in the overall corporate business plan, this would lead to an estimated Criticality value of at least 3.

The **Lethality** value is a measure of how severe the damage to the targeted system would be if the attack succeeded. The result of directory transversal attacks is usually compromise of the web server to some extent or the revelation of sensitive information to the attacker. This would lead to a Lethality value of 4 for this attack attempt.

The **System Countermeasures** value is a measure of the strength of the defensive mechanisms in place on the host itself. Since the hosts at 46.5.180.133 and 46.5.180.153 both responded to the initial connection. (See the packet headers below in which the attacking machine ACKs for packets from the two host.) There is the possibility that the systems were vulnerable to the attack. Also, there are no "`ATTACK RESPONSES 403 Forbidden`" alerts from the web servers that correspond to these connections. However, since no other data is available it cannot be determined conclusively that the attack was successful. The System Countermeasures value will be given a low value of 1 since the hosts responded.

```
07:19:21.524488 130.205.110.105.3937 > 46.5.180.133.80: P 59:117(58) ack 2921 win 0 [tos
0x10]
0x0000   4510 0062 0000 0000 f006 0000 82cd 6e69        E..b..........ni
0x0010   2e05 b485 0f61 0050 04d3 9de9 7ce8 d67e        .....a.P....|..~
0x0020   5018 0000 0000 0000 4745 5420 2f73 6372        P.......GET./scr
0x0030   6970 7473 2f2e 2e25 3563 2535 632e 2e2f        ipts/..%5c%5c../
0x0040   7769 6e6e 742f 7379 7374 656d 3332 2f63        winnt/system32/c
0x0050   6d64 2e65 7865 3f2f 632b 6469 720d 6469        md.exe?/c+dir.di
0x0060   720d                                           r.
07:19:22.024488 130.205.110.105.3957 > 46.5.180.153.80: P 59:117(58) ack 2921 win 0 [tos
0x10]
0x0000   4510 0062 0000 0000 f006 0000 82cd 6e69        E..b..........ni
0x0010   2e05 b499 0f75 0050 04e2 6947 7cae 36fe        .....u.P..iG|.6.
0x0020   5018 0000 0000 0000 4745 5420 2f73 6372        P.......GET./scr
0x0030   6970 7473 2f2e 2e25 3563 2535 632e 2e2f        ipts/..%5c%5c../
0x0040   7769 6e6e 742f 7379 7374 656d 3332 2f63        winnt/system32/c
0x0050   6d64 2e65 7865 3f2f 632b 6469 720d 6469        md.exe?/c+dir.di
0x0060   720d                                           r.
```

The **Network Countermeasures** value is a measure of the strength of the defensive mechanisms in place on the network. This attack is part of an established HTTP connection to a web server. Therefore, a network-based countermeasure would have to be a HTTP proxy firewall that could correctly parse the application layer traffic and filter on that traffic. This network does not appear to have that type of firewall. An estimated Network Countermeasures value is 3, based on the limited number of attacks seen in the snort alerts.

2.1.9 Defensive recommendation:

The defenses for this attack are:

- Ensuring the latest patches and hotfixes are applied to the web server.
- Setting the permissions on the `cmd.exe` and other executables that could be useful to an attacker to only allow the administrators access them.
- Move the root directory of the web server to another volume or at least out of the default installation location.
- Configure the IIS Server using Microsoft's Secure Internet Information Services 5 Checklist (http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/tools/chklist/iis5chk.asp)

- Use the Microsoft's IIS Lockdown Tool to securely configuring the web server. (http://www.microsoft.com/technet/security/tools/tools/locktool.asp)
- Use the URLScan Tool to provide continuing protection to the web server while in operation by restricting the kind of http requests that the server will process. (http://www.microsoft.com/technet/security/tools/tools/urlscan.asp)

2.1.10 Multiple choice test question:

In a directory transversal attack on a web server running on a Microsoft Windows operating system, the directory separator, "\", is always shown as a "\" in the hex dump from tcpdump.

a) True
b) False

Answer: False. The directory separator character can be sent by the attacker as the actual character or as the ASCII code (%5c) or Unicode character representation of the separation character.

2.2 Detect 2: Microsoft IIS Buffer Overflow Attack

2.2.1 Source of Trace.

The alert is from the raw logs obtained from http://www.incidents.org/logs/Raw/. The log files used were 2002.6.10 through 2002.6.18. The network layout of this network is unknown. Based on the fact that the machine responds on port 80/tcp, it appears that the machine at IP address 46.5.180.133 is a running as a web server.

2.2.2 Detect was generated by:

Snort intrusion detection system (version 1.9.0 (Build 209)) running on Mac OS X (v10.2.2). The rules and `snort.conf` (`$Id: snort.conf,v 1.110.2.4 2002/11/17 04:40:07 cazz Exp $`) were downloaded from http://www.snort.org/. The alerts were generated using the procedure discussed in the Detect 1 (see Section 2.1.2).

The rule that alerted is in `web-iis.rules` (`# $Id: web-iis.rules,v 1.30.2.12 2002/08/12 01:51:15 cazz Exp $`). The specific rule is:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"WEB-IIS ISAPI .ida attempt";
flow:to_server,established; uricontent:".ida?"; nocase; reference:arachnids,552;
classtype:web-application-attack; reference:bugtraq,1065; reference:cve,CAN-2000-0071;
sid:1243; rev:8;)
```

The Rule Header for this rule is interpreted by Snort as instructions to

`alert`	Generate and alert using the selected alert method
`tcp`	Analyze the TCP protocol
`$EXTERNAL_NET`	Match this source IP address in the packet to this value to satisfy the rule. The default `$EXTERNAL_NET` is `any`.
`any`	Match this source port in the packet to this value to satisfy the rule. This rule specifies `any` port.
`->`	Direction of flow from the source to the destination only.
`$HTTP_SERVERS`	Match the destination IP address of the packet to this value to satisfy the rule. The default `$HTTP_SERVERS` is `$HOME_NET` which is `any` by default.
`$HTTP_PORTS`	The destination port of the packet to this value to satisfy the rule. The default value of `$HTTP_PORTS` is 80.

The Rule Options are

`msg:"WEB-IIS ISAPI .ida attempt"`	The message to print along with the packet dump or to an alert.
`flow:to_server,established;`	Specifies that the alert triggers on client request from A to B or for an established connection. Used in conjunction with TCP stream reassembly.
`uricontent:".ida?";`	Match the string ".ida" in the URI portion of a request payload.
`nocase;`	Match the content string with case insensitivity
`classtype:web-application-attack;`	Categorizes the alert in to the web-application-attack class.
`reference:arachnids,552; classtype:web-application-attack; reference:bugtraq,1065; reference:cve,CAN-2000-0071;`	References for this signature from well known IDS and vulnerability listings
`sid:1243;`	The signature identification number
`rev:8;`	The revision number of the signature

2.2.3 Probability the source address was spoofed:

This detect is part of an established TCP connection to a web server. As can be seen in the tcpdump of packet from one of the attacking machines, 146.164.30.10, the ACK flag set and the TCP acknowledgement number is given for packe sent by the machine under attack. This leads to a low probability that the source address was spoofed.

```
% tcpdump -n -S -r 2002.6.17 'host  146.164.30.10'

01:21:32.664488 146.164.30.10.4383 > 46.5.180.133.80: P 2478892796:2478894260(1464) ack
3575519878 win 32120 [tos 0x10]
```

The registration information on this machine from dshield.org:

```
IP Address:                    146.164.30.10
HostName:                      divine.iq.ufrj.br
DShield Profile:

Country:                       BR
Contact E-mail:                carlos_AT_CEOP1.REDERIO.BR (bounced)
Total Records against IP:       1841
Number of targets:              676
Date Range:                    2002-11-08 to 2002-11-08Ports Attacked (up to 10):
Port                           Attacks
Fightback:                     sent to carlos@CEOP1.REDERIO.BR on 2002-07-07 18:58:26
                               message bounced
```

OrgName: Federal University of Rio de Janeiro
OrgID: FURDJ

NetRange: 146.164.0.0 - 146.164.255.255
CIDR: 146.164.0.0/16
NetName: REDE-UFRJ
NetHandle: NET-146-164-0-0-1
Parent: NET-146-0-0-0-0
NetType: Direct Assignment

NameServer: ULTRIX1.NCE.UFRJ.BR
NameServer: CEOP1.REDERIO.BR
NameServer: NOC.CERF.NET
Comment:
RegDate: 1991-02-15
Updated: 1992-10-15

TechHandle: CM169-ARIN
TechName: Mendes, Carlos
TechPhone: +55 021 598-3118
TechEmail: carlos@ceop1.rederio.br

OrgName: Various Registries (Maintained by ARIN)
OrgID: VR-ARIN

NetRange: 146.0.0.0 - 146.255.255.255
CIDR: 146.0.0.0/8
NetName: NET146
NetHandle: NET-146-0-0-0-0
Parent:
NetType: Early Registrations, Maintained by ARIN

NameServer: ARROWROOT.ARIN.NET
NameServer: BUCHU.ARIN.NET
NameServer: CHIA.ARIN.NET
NameServer: DILL.ARIN.NET
NameServer: EPAZOTE.ARIN.NET
NameServer: FIGWORT.ARIN.NET
NameServer: GINSENG.ARIN.NET
NameServer: HENNA.ARIN.NET
NameServer: INDIGO.ARIN.NET
Comment:
RegDate: 1993-05-01
Updated: 2002-08-23

OrgName: Federal University of Rio de Janeiro
OrgID: FURDJ
Address: Nucleo de Computacao Eletronica
Caixa Postal 2324
CEP 20.001
Rio de Janeiro, RJ
Country: BR
Comment:

RegDate: 1991-02-15
Updated: 1992-10-15

2.2.4 Description of attack:

This is a buffer overflow attack against a Microsoft Windows NT/2000 server running Internet Information Services (IIS) with the Indexing Service or Index Server installed. The buffer overflow attack is against the idq.dll that is part of the Indexing Server (IIS 4.0 on Windows NT 4.0) or Indexing Service (IIS 5.0 on Windows 2000). The unchecked buffer is in the code that handles the input URL. The idq.dll runs under system context so the attacker could gain complete control of the web server.

The overflow attack is identified by "`.ida?`" followed by a large crafted URL which contains the shell code. The initial portion of a trace of one of the attacks is shown below:

```
% tcpdump -nvvX -r 2002.6.17 'host 146.164.30.10'

01:21:32.664488 146.164.30.10.4383 > 46.5.180.133.80: P 2478892796:2478894260(1464) ack
3575519878 win 32120 [tos 0x10]  (ttl 240, id 0, len 1504, bad cksum 0!)
0x0000   4510 05e0 0000 0000 f006 0000 92a4 1e0a        E...............
0x0010   2e05 b485 111f 0050 93c0 e6fc d51e 1a86        .......P........
0x0020   5018 7d78 0000 0000 4745 5420 2f64 6566        P.}x....GET./def
0x0030   6175 6c74 2e69 6461 3f4e 4e4e 4e4e 4e4e        ault.ida?NNNNNNN
0x0040   4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e        NNNNNNNNNNNNNNNN
0x0050   4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e        NNNNNNNNNNNNNNNN
0x0060   4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e        NNNNNNNNNNNNNNNN
0x0070   4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e        NNNNNNNNNNNNNNNN
0x0080   4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e        NNNNNNNNNNNNNNNN
0x0090   4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e        NNNNNNNNNNNNNNNN
0x00a0   4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e        NNNNNNNNNNNNNNNN
0x00b0   4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e        NNNNNNNNNNNNNNNN
0x00c0   4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e        NNNNNNNNNNNNNNNN
0x00d0   4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e        NNNNNNNNNNNNNNNN
0x00e0   4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e        NNNNNNNNNNNNNNNN
0x00f0   4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e        NNNNNNNNNNNNNNNN
0x0100   4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e 4e4e        NNNNNNNNNNNNNNNN
0x0110   4e4e 4e4e 4e4e 4e4e 4e00 0000 0000 0000        NNNNNNNNN.......
0x0120   0000 0000 0000 0000 c303 0000 0078 00fa        .............x..
0x0130   2025 7539 3039 3025 7536 3835 3825 7563        .%u9090%u6858%uc
0x0140   6264 3325 7537 3830 3125 7539 3039 3025        bd3%u7801%u9090%
0x0150   7536 3835 3825 7563 6264 3325 7537 3830        u6858%ucbd3%u780
0x0160   3125 7539 3039 3025 7539 3039 3025 7538        1%u9090%u9090%u8
0x0170   3139 3025 7530 3063 3325 7530 3030 3325        190%u00c3%u0003%
0x0180   7538 6230 3025 7535 3331 6225 7535 3366        u8b00%u531b%u53f
0x0190   6625 7530 3037 3825 7530 3030 3025 7530        f%u0078%u0000%u0
0x01a0   303d 6120 2048 5454 502f 312e 300d 0a43        0=a..HTTP/1.0..C
0x01b0   6f6e 7465 6e74 2d74 7970 653a 2074 6578        ontent-type:.tex
0x01c0   742f 786d 6c0a 484f 5354 3a77 7777 2e77        t/xml.HOST:www.w
0x01d0   6f72 6d2e 636f 6d0a 2041 6363 6570 743a        orm.com..Accept:
0x01e0   202a 2f2a 0a43 6f6e 7465 6e74 2d6c 656e        .*/*.Content-len
0x01f0   6774 683a 2033 3536 3920 0d0a 0d0a 558b        gth:.3569.....U.
...(packet dump continues)
```

This attack is listed in the Common Vulnerabilities and Exposures (CVE) as CVE-2001-0500 (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0500).

2.2.5 Attack mechanism:

Code Red: The worm attacks by establishing a web session with a web server. The attack is sent as a HTTP GET request with a crafted URL to the web server for `default.ida` and the attack payload. The crafted URL fills and overflows a buffer in the ISAPI DLL that handles the Indexing Service (or Index Server). After the buffer overflow, the attack payload is executed and becomes the worm. The worm is now running at the system context since the Indexing Service would run at the system context. Note that the indexing service does not have to be running in order for the attack to succeed since the buffer overflow occurs before the indexing function is actually requested. If the Index Server is installed, the script mappings for internet data administration (`.ida`) and internet data query (`.idq`) files exist in the IIS metabase.

Once the Code Red worm has compromised the web server, it will continue on with a variety of activities. It begins by generating a total of 100 threads. All of the threads will check to see if the `c:\notworm` file exists, if the file exists then the worm goes dormant. The first 99 threads are used to attack other hosts if the day of the month is less than 19 (GMT). The worm randomly generates the IP addresses of the machines to attack next. The 100th thread is used to deface the web site if the web server is using US English as the default language. If the day of the month is 20 or greater, then the worm will conduct a denial of service flooding attack on www.whitehouse.gov by sending 100k bytes of data (1 byte at a time + 40 bytes overheard for the actually TCP/IP packet).

The definitive detailed analysis and disassembly of the Code Red worm can be found on the eEye Digital Security web site. (http://www.eeye.com/html/Research/Advisories/AL20010717.html). This detect is the original Code Red worm and not the Code Red II worm. Both worms use the same vulnerability to attack the IIS servers, the difference between the attacks on the IIS server is the filler string used to overflow the buffer. The original Code Red uses the character "N" in the http GET command and Code Red II uses "X" multiple times. The worm payload is also different for the two worms. The eEye Digital Security analysis of the Code Red II worm (http://www.eeye.com/html/Research/Advisories/AL20010804.html) discusses the differences between the two worms. Also, Code Red II worm will only run on Windows 2000 machines. It will crash other Windows variants.

Securityfocus.com has also produced analyses of the Code Red worm (http://aris.securityfocus.com/alerts/codered/010720-Analysis-CodeRed.pdf) and of the Code Red II worm (http://aris.securityfocus.com/alerts/codered2/010805-Analysis-CodeRedII.pdf) that are both partially based on eEye's work.

Cooperative Association for Internet Data Analysis (http://www.caida.org/) has an excellent analysis and descriptions of the Code Red variants and the initial timeline of their appearance at http://www.caida.org/analysis/security/code-red/.

2.2.6 Correlations:

This machine was reported as conducting probable Code Red or Nimda attacks against machines on the 134.29.X.X network on 6/14/2002 (http://www.mynetwatchman.com/LID.asp?IID=5151873). This machine was reported by numerous sources with the same signature as early as April 12,/2002 and as recently as November 8, 2002.

The rule that is the source of the alerts is not specific to a particular exploit although it is specific to a particular vulnerability. The alerts are definitely Code Red attempts to compromise the web server. A number of the source IP addresses are listed on http://sucrose.sugarmotor.net/worm.html which lists and graphs the current activity of Code Red, Code Red II and Nimda worms that attempt to attack that web server.

A Google.com search of the GIAC and SANS web sites did not return any hits for this IP address.

eEye Digital Security initially announced the discovery of the vulnerability (http://www.eeye.com/html/Research/Advisories/AD20010618.html) that is the basis of the Code Red. Microsoft confirmed the vulnerability in Security Bulletin MS01-033 (http://www.microsoft.com/technet/security/bulletin/MS01-033.asp).

The Code Red worm attack is also documented in arachNIDS (contributors: Dr SuSE (drsuse@drsuse.org) and C. Mayor: Based on packet trace from Nessus by Renaud Deraison). CERT CC issued CA2001-19 (http://www.cert.org/advisories/CA-2001-19.html) on the Code Red worm.

2.2.7 Evidence of active targeting:

There are multiple detects for this rule all against the web server at 46.5.180.133. The some of the source IP addresses are 146.164.30.10, 61.221.6.251, 212.180.27.59, 12.96.216.4, 65.103.247.126, 61.157.84.222, and 193.251.185.52. These IP addresses are registered in various countries: US, Brazil, China, and France. Each source IP address had one detect. It doesn't appear that the attacks are focused but appear as the random scanning of the Code Red worm.

2.2.8 Severity:

Severity is calculated with the following formula:

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

Severity = (3 + 5) – (2 +1) = 5

Each value is ranked on a scale from 1 (lowest) to 5 (highest).

The **Criticality** value is a measure of how critical the targeted system is to the network. DNS servers are normally critical pieces of the network infrastructure. The system appears to be an external web server. The criticality of the server is difficult to estimate. A reasonable approach is to assume that the web server provides a company's web presence but not e-commerce. Thus the criticality would be approximately 3.

The **Lethality** value is a measure of how severe the damage to the targeted system would be if the attack succeeded. This is a reconnaissance probe of the network. The attack is a known vulnerability that will lead to complete system compromise, thus the lethality is 5.

The **System Countermeasures** value is a measure of the strength of the defensive mechanisms in place on the host itself. The web server appears to be patched for this vulnerability since the alerts are seen though out the period of the raw log files. Without full knowledge of the state of countermeasures, an estimate of the system Countermeasure strength value is 2.

The **Network Countermeasures** value is a measure of the strength of the defensive mechanisms in place on the network. Again, the network is unknown. However since the snort logs were available, a reasonable amount of care has been taken for the network countermeasures. A rough estimate would be a value of 1 since the web server appears to be on an exposed network segment.

2.2.9 Defensive recommendation:

The defenses for this attack are:

- Ensuring the latest patches and hotfixes are applied to the web server
- Configure the IIS Server using Microsoft's Secure Internet Information Services 5 Checklist. (http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/tools/chklist/iis5chk.asp)
- Use the Microsoft's IIS Lockdown Tool to securely configuring the web server. (http://www.microsoft.com/technet/security/tools/tools/locktool.asp)
- Use the URLScan Tool to provide continuing protection to the web server while in operation by restricting the kind of http requests that the server will process. (http://www.microsoft.com/technet/security/tools/tools/urlscan.asp)
- Remove any unused ISAPI extension mappings.
- Disable the Indexing Service or Index Server as appropriate.

This web server appears to be patched correctly for this attack. The other network defenses cannot be assessed with any confidence.

2.2.10 Multiple choice test question:

In the Code Red worm, buffer overflow attacks which functions of the Microsoft Internet Information Services (IIS)?

a) the Indexing Service
b) the www service
c) the mapping of `.idq` and `.ida` extension
d) both b and c

Answer: D

The Code Red worm exploits a buffer overflow in a request for a .ida file to the IIS web server. The Indexing Service does not need to be running since the overflow occurs prior to the request reaching the Indexing Service.

2.3 Detect 3: DNS named Version Attempt

2.3.1 Source of Trace.

The alerts are from the raw logs obtained from http://www.incidents.org/logs/Raw/. The log files used were 2002.6.10 through 2002.6.18. The network layout of this network is unknown. The logs indicate that it is a large network and that the network ID has been obfuscated to a reserved address space.

2.3.2 Detect was generated by:

Snort intrusion detection system (version 1.9.0, build 209) running on Mac OS X (v10.2.2). The rules and `snort.conf ($Id: snort.conf,v 1.110.2.4 2002/11/17 04:40:07 cazz Exp $)` were downloaded from http://www.snort.org. The alerts were generated using the procedure discussed in the Detect 1 (see Section 2.1.2).

The rule that alerted is in `dns.rules (# $Id: dns.rules,v 1.26 2002/08/18 20:28:43 cazz Exp $).` The specific rule is :

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"DNS named version attempt";
flow:to_server,established; content:"|07|version"; offset:12; content:"|04|bind"; nocase;
offset: 12; reference:nessus,10028; reference:arachnids,278; classtype:attempted-recon;
sid:257; rev:4;)
```

The Rule Header for this rule is interpreted by Snort as instructions to

`alert`	Generate and alert using the selected alert method
`tcp`	Analyze the TCP protocol
`$EXTERNAL_NET`	Match this source IP address in the packet to this value to satisfy the rule. The default `$EXTERNAL_NET` is `any`.
`any`	Match this source port in the packet to this value to satisfy the rule. This rule specifies `any` port.
`->`	Direction of flow from the source to the destination only.
`$HOME_NET`	Match the destination IP address of the packet to this value to satisfy the rule. The default `$HOME_NET` which is `any` by default.
`53`	Match the destination port of the packet to this 53 to satisfy the rule.

The Rule Options are

`msg:"DNS named version attempt";`	The message to print along with the packet dump or to an alert.
`flow:to_server,established;`	Specifies that the alert triggers on client request from A to B or for an established connection. Used in conjunction with TCP stream reassembly.
`content:"|07|version";`	Match the string "|07|version" in the URI portion of a request payload.
`offset:12;`	Modifies the starting search position for the pattern matching function to 12 from the beginning of the packet payload
`content:"|04|bind";`	Match the string "|04|version" in the URI portion of a request payload.
`nocase;`	Match the content string with case insensitivity
`offset:12;`	Modifies the starting search position for the pattern matching function to 12 from the beginning of the packet payload

`reference:nessus,10028;`
`reference:arachnids,278;`
`classtype:attempted-recon;`
	References for this signature from well known IDS and vulnerability listings

`sid:257;`	The signature identification number
`rev:4;`	The revision number of the signature

2.3.3 Probability the source address was spoofed:

The packets that generated the alerts are from UDP connections to a machine. It is possible that the source IP address is spoofed since that is easily accomplished. However, this is a reconnaissance probe and the attacker would need to be able to intercept the response packets from the machine probed for the probe to be useful. Therefore, it is unlikely that the source address is spoofed.

For this analysis I will concentrate on 203.122.47.137, the source IP address that was the source of the largest number of alerts, as seen in the SnortSnarf output in Figure 2-3. This source IP address was active performing the reconnaissance during the entire period of the logs review (Earliest: 21:25:33 on 07/09/2002 to Latest: 07:09:5207/18/2002).

APNIC lookup on the source IP address (203.122.47.137) gives the following information:

```
% [whois.apnic.net node-2]

inetnum:      203.122.47.0 - 203.122.47.255
netname:      SHARED-DSL-OKH-II
country:      IN
descr:        Pool of IP's dynamically assigned to DSL routers of Okhla
descr:        email : j.grewal@in.spectranet.com
admin-c:      JG131-AP
tech-c:       JG131-AP
status:       ASSIGNED NON-PORTABLE
changed:      harpreet.singh@in.spectranet.com 20021025
mnt-by:       MAINT-IN-SPECTRA-NET-LTD
source:       APNIC
```

2.3.4 Description of attack:

The alert rule is written to detect the attempt of a malicious user to remotely determine the version of the BIND named daemon. The tcpdump of two of the 104 packets sent by the attacking machine are shown below.

```
% tcpdump -nvvX -r 2002.6.10 'host 203.122.47.137'

21:25:33.684488 203.122.47.137.21218 > 46.5.192.160.53: [bad udp cksum f9f9!]  4660
[b2&3=0x80] TXT CHAOS? version.bind. (30) (ttl 40, id 17927, len 58, bad cksum 6909!)
0x0000   4500 003a 4607 0000 2811 6909 cb7a 2f89       E..:F...(.i..z/.
0x0010   2e05 c0a0 52e2 0035 0026 948e 1234 0080       ....R..5.&...4..
0x0020   0001 0000 0000 0000 0776 6572 7369 6f6e       .........version
0x0030   0462 696e 6400 0010 0003                      .bind.....
23:19:08.074488 203.122.47.137.18548 > 46.5.245.34.53: [bad udp cksum faf7!]  4660
[b2&3=0x80] TXT CHAOS? version.bind. (30) (ttl 42, id 10259, len 58, bad cksum 4f7d!)
0x0000   4500 003a 2813 0000 2a11 4f7d cb7a 2f89       E..:(...*.O}.z/.
0x0010   2e05 f522 4874 0035 0026 697c 1234 0080       ..."Ht.5.&i|.4..
0x0020   0001 0000 0000 0000 0776 6572 7369 6f6e       .........version
0x0030   0462 696e 6400 0010 0003                      .bind.....
```

The packets show UDP packets, protocol 17 (`0x11`), going to destination port 53 (`0x35`). The DNS portion of the packet begins at offset byte 14. The first two bytes of the DNS portion of the packet is the DNS query/response identification number. Note that in the two packets the ID number is identical, i.e. `1234`. This ID number is consistent throughout the DNS probe packets sent by this source IP address. This leads me to the conclusion that an automated tool crafts the packets used to probe for vulnerable DNS servers.

2.3.5 Attack mechanism:

The attempt to obtain the version of the `named` daemon is reconnaissance in preparation of an attack on the DNS server. There are several vulnerabilities that apply to various versions of BIND (http://www.isc.org/products/BIND/bind-security.html). The vulnerabilities vary in severity and some will lead to root compromise of the DNS server while others will allow the malicious intruder viewing to environmental variables of the DNS server and others are denial of service attacks.

The CVE (http://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=bind+named) has several vulnerabilities listed for BIND:

CVE-1999-0835	Denial of service in BIND named via malformed SIG records.
CVE-1999-0848	Denial of service in BIND named via consuming more than "fdmax" file descriptors.
CVE-1999-0849	Denial of service in BIND named via maxdname.
CVE-1999-0851	Denial of service in BIND named via naptr.
CVE-2000-0887	named in BIND 8.2 through 8.2.2-P6 allows remote attackers to cause a denial of service by making a compressed zone transfer (ZXFR) request and performing a name service query on an authoritative record that is not cached, aka the "zxfr bug."
CVE-2000-0888	named in BIND 8.2 through 8.2.2-P6 allows remote attackers to cause a denial of service by sending an SRV record to the server, aka the "srv bug."
CAN-1999-1499	named in ISC BIND 4.9 and 8.1 allows local users to destroy files via a symlink attack on (1) named_dump.db when root kills the process with a SIGINT, or (2) named.stats when SIGIOT is used.
CAN-2002-1219	Buffer overflow in named in BIND 4 versions 4.9.10 and earlier, and 8 versions 8.3.3 and earlier, allows remote attackers to execute arbitrary code via a certain DNS server response containing SIG resource records (RR).

This type of DNS probing is also documented as arachNIDS IDS278 (http://www.whitehats.com/IDS/278). There are also several other `named` probes and exploits listed in arachNIDS:

• IDS277/named-probe-iquery	• IDS278/named-probe-version
• IDS480/named-probe-authors	• IDS482/named-exploit-infoleak-lsd
• IDS489/named-exploit-tsig-lsd	• IDS490/named-exploit-tsig-lucysoft
• IDS491/named-exploit-tsig-tsig0wn	

2.3.6 Correlations:

The source IP address has a record with dshield.org of attempting to attack DNS servers (http://www.dshield.org/ipinfo.php?ip=203.122.47.137).

IP Address:	203.122.47.137
HostName:	203.122.47.137
DShield Profile:	
Country:	IN
Contact E-mail:	sachin.mehra_AT_in.spectranet.com (bounced)
Total Records against IP:	2168
Number of targets:	1856
Date Range:	2002-12-06 to 2002-12-06
Ports Attacked (up to 10):	Port Attacks 53 24

This machine was also reported conducting DNS attacks on the 216.37.X.X network on June 8, 2002 (http://www.mynetwatchman.com/LID.asp?IID=5093973). This machine started these attacks around October 11, 2001 and they continued until December 13, 2002.

This IP address was also noted in the GCIA detect analysis of Ewen YW Fung that was submitted to the intrusions@incidents.org mailing list on 22 August 2002. A Google.com search of the GIAC and SANS web sites did not return any hits for this IP address.

This source IP address was also noted doing DNS probing by the IDS at the University of Notre Dame on March 20, 2002. These probes were analyzed with Snort and SnortSnarf (v020316.1) (http://www.nd.edu/~dmehlber/ids/html3/html/203/122/47/src203.122.47.137.html).

This IP address was also logged by Virginia Polytechnic Institute and State University attempting a DNS `named` version attempt on May 18, 2002 against one of their machines. (http://rdweb.cns.vt.edu/~lat/log_archives/020518.txt)

2.3.7 Evidence of active targeting:

The alerts indicate that the network is being probed for vulnerable DNS servers. A search of the Internet using Google.com leads to several instances of the source IP address performing the same reconnaissance on other networks. This is not active targeting at this point. If a vulnerable DNS server were located by the probing, then active targeting would probably occur.

2.3.8 Severity:

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

Severity = (4 + 1) – (3 + 1) = 1

Each value is ranked on a scale from 1 (lowest) to 5 (highest).

The **Criticality** value is a measure of how critical the targeted system is to the network. DNS servers are normally critical pieces of the network infrastructure. Assuming that one of the machines probed was a DNS server for the network, the Criticality value is estimated at 4.

The **Lethality** value is a measure of how severe the damage to the targeted system would be if the attack succeeded. This is a reconnaissance probe of the network. The probe itself is benign, although it may lead to more offensive actions. This would lead to a Lethality value of 1 for this probe.

The **System Countermeasures** value is a measure of the strength of the defensive mechanisms in place on the host itself. Without knowledge of the configuration of the probed systems or types of responses from the probed systems to the attacking systems, it is difficult to assess the System Countermeasures. A System Countermeasures value of 3 is estimated.

The **Network Countermeasures** value is a measure of the strength of the defensive mechanisms in place on the network. There are no network countermeasures that will prevent an attack or probe of a DNS serve except for a firewall that either blocks or proxies DNS traffic. The DNS proxy would have filter the queries sent to the DNS server to prevent the probes and subsequent attacks. With no knowledge of the network, an estimated Network Countermeasures value is 1.

2.3.9 Defensive recommendation:

- Update DNS servers running BIND named to the latest version that is not vulnerable to known exploits. The latest version is 9.2.1. However, BIND 8.3.4 and 4.9.11 are available if it is not possible to upgrade to DIND 9.2.1.
- Configure BIND not to answer to the version queries that are used in the probing.
- Run BIND in a `chroot()` environment. References for setting up the `chroot()` are available at http://www.isc.org/products/BIND/contributions.html.
- Download the source code for BIND. Change the version information in the code and recompile. This will obscure the correct version of BIND form the attacker.
- Use a split horizon DNS configuration. This configuration has one DNS server that handles DNS queries from outside your network and another DNS server that responds to queries from your internal network. With the different security configurations on each machine, if one is compromised it is unlikely that the other will be affected.
- Install tcpwrappers and configure it to block access to `named` except from trusted machines.

- Disallow zone transfers except to trusted hosts. This is also good security practice by itself.
- If the DNS server is internal to any firewall, the firewall could be configured to block these probes. However this will not be possible for public DNS servers exposed to the internet.

2.3.10 Multiple choice test question:

As part of an effort to baseline the traffic on your network prior to implementing a intrusion detection system, you are sniffing the network external to your network's firewall with tcpdump. While you are looking at the tcpdump output, you notice packets similar to the following:

```
21:25:33.684488 203.122.47.137.21218 > 46.5.192.160.53: [bad udp cksum f9f9!]  4660
[b2&3=0x80] TXT CHAOS? version.bind. (30) (ttl 40, id 17927, len 58, bad cksum 6909!)
0x0000   4500 003a 4607 0000 2811 6909 cb7a 2f89       E..:F...(.i..z/.
0x0010   2e05 c0a0 52e2 0035 0026 948e 1234 0080      ....R..5.&...4..
0x0020   0001 0000 0000 0000 0776 6572 7369 6f6e       .........version
0x0030   0462 696e 6400 0010 0003                      .bind.....

23:19:08.074488 203.122.47.137.18548 > 46.5.245.34.53: [bad udp cksum faf7!]  4660
[b2&3=0x80] TXT CHAOS? version.bind. (30) (ttl 42, id 10259, len 58, bad cksum 4f7d!)
0x0000   4500 003a 2813 0000 2a11 4f7d cb7a 2f89       E..:(...*.O}.z/.
0x0010   2e05 f522 4874 0035 0026 697c 1234 0080      ..."Ht.5.&i|.4..
0x0020   0001 0000 0000 0000 0776 6572 7369 6f6e       .........version
    0x0030   0462 696e 6400 0010 0003                      .bind.....
```

This traffic is

a) normal traffic going to a DNS server
b) part of a Nessus scan of your network
c) the result of an attacker running a denial of service exploit on your DNS servers
d) the result of an attacker scanning for DNS servers running a vulnerable version of named.

Answer: D

The packets shown are queries for the version of BIND running on the destination machine. Based on the results of the queries, the attacker can determine if the BIND `named` daemon is vulnerable to any exploits.

3 Part 3 – Analyze This

3.1 Executive Summary

This is portion of the practical is a security audit for a University. The audit consists of analyzing the various logs from their intrusion detection system to produce a report. The intrusion detection system that produced the data is a Snort IDS system with a fairly standard rule base. The data was downloaded from http://www.incidents.org/logs. The log data from five consecutive days, November 11-15, 2002, were analyzed.

There are number of machines on the network that must be investigated for possible Distributed Denial of Service (DDoS) clients worms, Trojans infection. Also, close examination of a number of other machines is required due to suspicious network activity directed at them or coming from them These machines are identified in the analysis of the various log files. A recommendation is put forth for the entire network to be scanned for vulnerabilities and correct the identified discrepancies in a prioritized manner. This will be a large undertaking for a large university network but the security of the network wills probably degrade unless these actions are taken on an ongoing manner.

There were a significant number of scans occurring during this period. Several hundred thousand scans were logged. In addition a large number of reconnaissance type traffic, i.e., Queso and nmap scans, was also noted. The large volume of the scans made analysis difficult. A significant amount of the volume from scans was removed using quickly generated Perl scripts. This allowed the analysis of the data with SnortSnarf (v021111.1). The data from the logs was then analyzed to identify the most likely candidate machines for investigation and to identify any unusual activity.

3.2 File Formats

List of Files:

Alerts	Scans	Out of Specification
alert.021011.gz	scans.021011.gz	OOS_Report_2002_10_11_21861
alert.021012.gz	scans.021012.gz	OOS_Report_2002_10_12_29999
alert.021013.gz	scans.021013.gz	OOS_Report_2002_10_13_9575
alert.021014.gz	scans.021014.gz	OOS_Report_2002_10_14_21815
alert.021015.gz	scans.021015.gz	OOS_Report_2002_10_15_13854

3.2.1 Alerts Logs

The alerts are in a single line format. The alerts contain the basic packet information; date-time stamp, source and destination IP addresses and port numbers for TCP and UDP packets. No ICMP alerts were in the logs. The first two octets of the internal network IP addresses were obfuscated with "MY.NET" which prevented use of SnortSnarf. To allow use of SnortSnarf, the obfuscated octets were changed to "255.255" which should not be in the logs as an external address. The IP addresses were changed back to the MY.NET format in this paper

```
<date-time stamp>  [**]message [**] <src IP>:<src port> -> <dest IP>:<dest port>

10/11-00:07:11.260858  [**] SMB Name Wildcard [**] 200.53.80.43:1032 -> MY.NET.134.95:137
10/11-00:43:13.484548  [**] Incomplete Packet Fragments Discarded [**] 216.54.222.175:0 ->
        MY.NET.53.36:0

10/11-00:37:27.220192  [**] High port 65535 udp - possible Red Worm - traffic
        [**]MY.NET.198.204:1214 -> 137.45.71.61:65535
10/11-00:37:25.228161  [**] IDS552/web-iis_IIS ISAPI Overflow ida nosize [**]
        218.16.42.154:3374 -> MY.NET.184.24:80
```

3.2.2 Scan Logs

The scan logs use a format similar to the alert logs. The differences are the date-time stamp does not contain the microseconds and last entry of the line contains either "UDP" or the TCP flags. In this case the local IP addresses were not obfuscated in the files that were down loaded. The IP addresses will be obfuscated in same manner as the alerts in this paper

```
<date> <time> <src IP>:<src port> -> <dest IP>:<dest port> [<TCP flags> | UDP]
Oct 11 00:02:34 MY.NET.84.147:4002 -> 24.94.50.252:1751 SYN ******S*
Oct 11 00:02:34 MY.NET.84.147:1214 -> 35.11.178.29:1531 UDP
Oct 11 00:02:37 MY.NET.84.147:1214 -> 35.11.178.29:1531 UDP
```

3.2.3 OOS Logs

The OOS logs use the multi-line format from Snort running in sniffer mode (`snort -v`). The first line of the format contains the basic packet information. The rest of the log entry contains the some specifics from the packet header and optionally part of the packet contents.

```
<date-time stamp> <src IP>:<src port> -> <dest IP>:<dest port>

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

10/11-00:17:25.702792 209.116.70.75:37014 -> MY.NET.100.217:25
TCP TTL:50 TOS:0x0 ID:52463 IpLen:20 DgmLen:60 DF
12****S* Seq: 0x839A3BC7  Ack: 0x0  Win: 0x16D0  TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 911537960 0 NOP WS: 0

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

10/11-00:18:28.615098 200.221.192.194:2735 -> MY.NET.91.81:1214
TCP TTL:104 TOS:0x0 ID:36915 IpLen:20 DgmLen:391 DF
***P*** Seq: 0x829D40A  Ack: 0x0  Win: 0x2000  TcpLen: 20
47 45 54 20 2F 35 33 2F 54 68 65 5F 43 6F 75 6E  GET /53/The_Coun
74 5F 6F 66 5F 4D 6F 6E 74 65 5F 43 72 69 73 74  t_of_Monte_Crist
6F 5F 25 32 38 32 30 30 32 25 32 39 2E 43 44 32  o_%282002%29.CD2
2E 52 46 74 61 2E 53 68 61 72 65 52 65 61 63 74  .RFta.ShareReact
6F 72 2E 61 76 69 20 48 54 54 50 2F 31 2E 31 0D  or.avi HTTP/1.1.
0A 48 6F 73 74 3A 20 31 33 30 2E 38 35 2E 39 31  .Host: MY.NET.91
2E 38 31 3A 31 32 31 34 0D 0A 55 73 65 72 41 67  .81:1214..UserAg
65 6E 74 3A 20 4B 61 7A 61 61 43 6C 69 65 6E 74  ent: KazaaClient
20 44 65 63 20 31 34 20 32 30 30 31 20 31 37 3A   Dec 14 2001 17:
33 39 3A 33 34 0D 0A 58 2D 4B 61 7A 61 61 2D 55  39:34..X-Kazaa-U
73 65 72 6E 61 6D 65 3A 20 61 76 61 6E 73 6F 0D  sername: avanso.
0A 58 2D 4B 61 7A 61 61 2D 4E 65 74 77 6F 72 6B  .X-Kazaa-Network
3A 20 3F 3F 3F 0D 0A 58 2D 4B 61 7A 61 61 2D 49  : ???..X-Kazaa-I
50 3A 20 31 39 32 2E 31 36 38 2E 30 2E 35 38 3A  P: 192.168.0.58:
31 32 31 34 0D 0A 58 2D 4B 61 7A 61 61 2D 53 75  1214..X-Kazaa-Su
70 65 72 6E 6F 64 65 49 50 3A 20 31 35 32 2E 33  pernodeIP: 152.3
30 2E 39 38 2E 31 31 30 3A 31 35 34 35 0D 0A 52  0.98.110:1545..R
61 6E 67 65 3A 20 62 79 74 65 73 3D 32 37 31 39  ange: bytes=2719
36 30 38 30 38 2D 34 31 32 38 36 36 33 36 31 0D  60808-412866361.
0A 43 6F 6E 6E 65 63 74 69 6F 6E 3A 20 63 6C 6F  .Connection: clo
73 65 0D 0A 58 2D 4B 61 7A 61 61 2D 58 66 65 72  se..X-Kazaa-Xfer
49 64 3A 20 39 36 35 30 31 36 36 0D 0A 0D 0A     Id: 9650166....

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

3.3 Analysis

3.3.1 Alerts

There were a large number of alerts during the five-day period covered by the logs. The alert log for two of the days was over 50MB in size once they were gunzipped. The size of these logs prevented their successful analysis with SnortSnarf. SnortSnarf is limited in the size of the logs it can analyze at one time due to it memory usage. The Apple PowerMac 9600, which has upgraded wit a G3/400 Mhz processor, that was used to analyze the logs has 768 MB of RAM. Analysis of the larger logs without removing the two most common log entries failed.

In order to process the logs with SnortSnarf, the large numbers of `spp_portscan` log entries were removed by editing the files with BBEdit 7.0. This process removed approximately 344,000 log entries over the five-day period of the logs.

Also, the logs from 10/12 and 10/13, contained an extremely large number of entries between the machines at MY.NET.83.146:1379 and 24.59.33.240:65535. The entries were due to the rule with the message "`High port 65535 tcp - possible Red Worm - traffic.`" A BBEdit was also used to remove the approximately 751,600 entries for this rule.

The combination of removing the `spp_portscan` entries and the "`possible Red Worm`" entries reduced the total size of the files from 146.9MB down to 16.0MB. The resulting log files were small enough to be processed by SnortSnarf in a single batch.

3.3.1.1 SnortSnarf

The SnortSnarf command used to analysis the alert data was

```
% ./snortsnarf.pl -rs -d ../snarf-alert-part3 ../logs/alerts/alert-rds-mod.021011
       ../logs/alerts/alert-rds-mod.021012 ../logs/alerts/alert-rds-mod.021013
       ../logs/alerts/alert-rds-mod.021014 ../logs/alerts/alert-rds-mod.021015
```

The files specified are the log files that were modified to remove the `spp_portscan` and "`possible Red Worm`" the The options for SnortSnarf are

`-d <directory>`	<directory> is the path to the directory the HTML pages will be generated in.
`-rs`	reverses the order of signature listing on the first page so that the most interesting signatures appear first

The SnortSnarf results are shown in the following table. There were a total of 152,028 alerts processed by SnortSnarf (v021111.1).

The complete list of signatures is in the following table. The "`High Port`" entries were manually inserted back into the table.

Table 3-2, Alert Signatures (Sorted by Number of Alerts)

Signature	# Alerts	# Sources	# Dests
High port 65535 tcp - possible Red Worm - traffic	752351	12	12
spp_http_decode: IIS Unicode attack detected	64191	566	1066
CS WEBSERVER - external web traffic	20956	1176	1
Watchlist 000220 IL-ISDNNET-990517	20011	98	70
SMB Name Wildcard	19261	530	897
SUNRPC highport access!	5491	41	41
FTP DoS ftpd globbing	3736	15	2
SYN-FIN scan!	3064	1	3064
spp_http_decode: CGI Null Byte attack detected	2542	53	68
Queso fingerprint	2355	117	1568
IDS552/web-iis_IIS ISAPI Overflow ida nosize	2193	2019	549
High port 65535 udp - possible Red Worm - traffic	1934	111	122
Incomplete Packet Fragments Discarded	1033	19	20
External RPC call	965	2	598
Watchlist 000222 NET-NCFC	964	29	42
EXPLOIT x86 NOOP	618	26	28
Port 55850 tcp - Possible myserver activity - ref. 010313-1	302	33	34
MYPARTY - Possible My Party infection	190	1	1
connect to 515 from outside	184	2	174
Null scan!	183	36	25
Tiny Fragments - Possible Hostile Activity	160	4	4
CS WEBSERVER - external ftp traffic	150	33	1
EXPLOIT x86 setuid 0	136	50	30
IRC evil - running XDCC	123	1	4
NMAP TCP ping!	104	26	24

Signature	# Alerts	# Sources	# Dests
SMB C access	94	54	15
TCP SRC and DST outside network	68	4	13
Port 55850 udp - Possible myserver activity - ref. 010313-1	48	9	7
EXPLOIT x86 setgid 0	46	33	18
Possible trojan server activity	43	9	9
TFTP - Internal UDP connection to external tftp server	32	4	6
External FTP to HelpDesk MY.NET.70.49	17	6	1
TFTP - External TCP connection to internal tftp server	13	8	8
External FTP to HelpDesk MY.NET.70.50	13	8	1
Bugbear@MM virus in SMTP	13	11	7
RFB - Possible WinVNC - 010708-1	10	6	7
Attempted Sun RPC high port access	7	4	4
TFTP - External UDP connection to internal tftp server	6	5	5
HelpDesk MY.NET.70.50 to External FTP	4	1	2
HelpDesk MY.NET.70.49 to External FTP	4	1	4
EXPLOIT NTPDX buffer overflow	3	3	3
HelpDesk MY.NET.83.197 to External FTP	2	1	1
ICMP SRC and DST outside network	2	1	1
Fragmentation Overflow Attack	1	1	1
Probable NMAP fingerprint attempt	1	1	1
DDOS TFN client command BE	1	1	1
Back Orifice	1	1	1
External FTP to HelpDesk MY.NET.83.197	1	1	1
DDOS shaft client to handler	1	1	1

3.3.1.2 Alert Signature Sources

The source of the signatures in the alert logs was investigated using Google.com and the signature database search feature on snort.org (http://www.snort.org/snort-db/sid.html?sid=). The list of signatures was broken up into four categories by the estimated source:

- snort.org with current SID or a current SID that covers that signature
- snort.org but does not currently appear in the database
- Port Scan preprocessor output
- Site generated.

Some of the site-generated signatures appear to have been obtained from arachNIDS. Several of the site signatures are specific to the site, specifically the Help Desk and CS WEBSERVER signatures. Table 3-2 lists the signatures by the relative urgency of the attack or signature and also gives the estimated source of the signature. The top five signature in Trojan/DdoS category will be investigated in the Link Analysis Graph in Section 3.6.

Table 3-3, Alert Signatures (Sorted by Type)

Signature	Source
DDOS TFN client command BE	SID 228
DDOS shaft client to handler	SID 230
TFTP - Internal UDP connection to external tftp server	site
TFTP - External TCP connection to internal tftp server	site
TFTP - External UDP connection to internal tftp server	site/arachNIDS
SUNRPC highport access!	SID ?
High port 65535 tcp - possible Red Worm - traffic	SID ?
High port 65535 udp - possible Red Worm - traffic	SID ?
External RPC call	site

Signature	Source
Port 55850 tcp - Possible myserver activity - ref. 010313-1	site
Port 55850 udp - Possible myserver activity - ref. 010313-1	site
MYPARTY - Possible My Party infection	site
Back Orifice	site
IRC evil - running XDCC	site
Possible trojan server activity	site
EXPLOIT NTPDX buffer overflow	SID 312
Tiny Fragments - Possible Hostile Activity	SID 522
SMB C access	SID 533
EXPLOIT x86 setgid 0	SID 649
EXPLOIT x86 setuid 0	SID 650
FTP DoS ftpd globbing	SID 1377
EXPLOIT x86 NOOP	SID 1394
Attempted Sun RPC high port access	SID ?
SMB Name Wildcard	site
connect to 515 from outside	site
IDS552/web-iis_IIS ISAPI Overflow ida nosize	site/arachNIDS
Fragmentation Overflow Attack	site
Null scan!	SID 623
SYN-FIN scan!	SID 624
NMAP TCP ping!	SID 628
Probable NMAP fingerprint attempt	SID 629
Queso fingerprint	SID ?
spp_http_decode: IIS Unicode attack detected	Port Scan Preprocessor
spp_http_decode: CGI Null Byte attack detected	Port Scan Preprocessor
Incomplete Packet Fragments Discarded	site
TCP SRC and DST outside network	site (misc. network anomalies)
ICMP SRC and DST outside network	site (misc. network anomalies)
Bugbear@MM virus in SMTP	site
RFB - Possible WinVNC - 010708-1	site
Watchlist 000220 IL-ISDNNET-990517	site
Watchlist 000222 NET-NCFC	site
CS WEBSERVER - external web traffic	site
CS WEBSERVER - external ftp traffic	site
HelpDesk MY.NET.70.49 to External FTP	site
HelpDesk MY.NET.70.50 to External FTP	site
HelpDesk MY.NET.83.197 to External FTP	site
External FTP to HelpDesk MY.NET.70.49	site
External FTP to HelpDesk MY.NET.70.50	site
External FTP to HelpDesk MY.NET.83.197	site

Color Code Key for Table 3-2

(pink)	Trojans/DDoS: Machines on the local network should have the file system and audit record examined closely for possible compromise.
(yellow)	Exploits: Machine on the local network should be closely monitored, both network activity and suditing, for signs of compromise due the attack signature noted in the alert.
(gray)	Scanning activity
(blue)	Miscellaneous network anomalies which should be investigated for possible problems with the network,
(light cyan)	Watchlist: External subnets which have probably been a source of attacks in the past based on the rules.

	Local Web Server: A local rule to monitor the traffic to and from the web server.
	Monitoring the Help Desk machine

3.3.1.3 Top Talkers

The Top Talkers List for both source and destination are shown below without the port scans.

Table 3-4, Top Ten Talker Alert Sources

Rank	Total # Alerts	Source IP	# Signatures triggered	Destinations involved
		MY.NET.83.146	1 signatures	24.59.33.240
		24.59.33.240	1 signatures	MY.NET.83.146
1	24621 alerts	MY.NET.85.74	1 signatures	(5 destination IPs)
2	12388 alerts	212.179.83.64	1 signatures	MY.NET.114.88, MY.NET.84.147
3	4572 alerts	MY.NET.84.133	1 signatures	(16 destination IPs)
4	3167 alerts	128.8.120.85	1 signatures	MY.NET.99.205
5	3064 alerts	152.101.81.195	1 signatures	(3064 destination IPs)
6	2809 alerts	66.77.73.144	1 signatures	MY.NET.100.165
7	2583 alerts	MY.NET.152.22	1 signatures	218.55.184.152
8	2034 alerts	212.179.97.145	1 signatures	MY.NET.133.216, MY.NET.104.204
9	2011 alerts	129.6.153.67	1 signatures	MY.NET.109.85
10	1815 alerts	66.77.73.236	1 signatures	MY.NET.100.165

Table 3-5, Top Ten Talker Alert Destinations

Rank	Total # Alerts	Destination IP	# Signatures triggered	Originating sources
		24.59.33.240	1 signatures	MY.NET.83.146
		MY.NET.83.146	1 signatures	24.59.33.240
1	21108 alerts	MY.NET.100.165	3 signatures	(1209 source IPs)
2	12402 alerts	207.200.86.66	1 signatures	(8 source IPs)
3	12399 alerts	207.200.86.97	1 signatures	(5 source IPs)
4	12377 alerts	MY.NET.114.88	3 signatures	(7 source IPs)
5	6894 alerts	218.55.184.152	1 signatures	(7 source IPs)
6	3745 alerts	MY.NET.100.158	6 signatures	(21 source IPs)
7	3168 alerts	MY.NET.99.205	2 signatures	128.8.120.85, 152.101.81.195
8	2247 alerts	211.115.212.150	1 signatures	(9 source IPs)
9	2138 alerts	MY.NET.104.204	4 signatures	(8 source IPs)
10	2012 alerts	MY.NET.109.85	2 signatures	152.101.81.195, 129.6.153.67

3.3.2 Out of Specification

There were 18277 alerts from 274 sources to 8224 destinations. The vast majority was generated by three sources doing scans of various parts of the /16 network. Two of the sources were external and one internal. The two external sources are investigated in the External Address Registration section of this paper.

3.3.2.1 Perl Script

The Perl script, `parse-oos.pl`, was used to convert the OOS logs to the format of the alert logs so that SnortSnarf could be used to analyze the logs. (See Appendix A for a listing of the Perl code.) These are the log entries after being modified for use with SnortSnarf by the Perl script:

```
10/10-22:18:10.649752  [**] OOS [**]  68.43.37.13:0 -> MY.NET.70.176:2247 TCP TTL:112 TOS:0x0
       ID:34896 IpLen:20 DgmLen:177 DF 12UA**** Seq: 0x1A2B00B1  Ack: 0x28D296AC  Win:
       0x5018  TcpLen: 60  UrgPtr: 0x3D2F TCP Options (1) => EOL

10/10-22:25:52.781015  [**] OOS [**]  68.43.37.13:117 -> MY.NET.70.176:2247 TCP TTL:112
       TOS:0x0 ID:40363 IpLen:20 DgmLen:40 DF 12**P*SF Seq: 0x1A2B00B7  Ack: 0x419296BC
       Win: 0x5010  TcpLen: 28 TCP Options (1) => EOL
```

The same log entries in the original format:

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

10/10-22:18:10.649752 68.43.37.13:0 -> MY.NET.70.176:2247
TCP TTL:112 TOS:0x0 ID:34896 IpLen:20 DgmLen:177 DF
12UA**** Seq: 0x1A2B00B1  Ack: 0x28D296AC  Win: 0x5018  TcpLen: 60  UrgPtr:
0x3D2F
TCP Options (1) => EOL
9C 8F D0 FA D4 1C 8C 2A 46 54 40 A8 C2 DE EE 40  .......*FT@....@
19 AE CE B8 95 D7 1E DE 55 AB B4 31 A8 A7 B6 19  ........U..1....
93 6A 1F ED DD 43 7D F4 52 16 11 D3 2D CC A5 9C  .j...C}.R...-...
CD B0 E8 C5 55 CB 79 18 FD 68 26 FC 4B FB 8C D8  ....U.y..h&.K...
CA 54 5A 47 A6 BA 15 CD 4A 10 9D 16 05 33 D3 7C  .TZG....J....3.|
7D 66 26 16 E3 6F E0 2D A4 8C 8C E6 75 57 EA EF  }f&..o.-....uW..
93                                               .

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

10/10-22:25:52.781015 68.43.37.13:117 -> MY.NET.70.176:2247
TCP TTL:112 TOS:0x0 ID:40363 IpLen:20 DgmLen:40 DF
12**P*SF Seq: 0x1A2B00B7  Ack: 0x419296BC  Win: 0x5010  TcpLen: 28
TCP Options (1) => EOL

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

3.3.2.2 SnortSnarf

SnortSnarf was then used to conduct the initial analysis. All of the OOS log files were combined in to a single file, `full-oos-parsed`. SnortSnarf was the used to analyze the combined file using the command:

```
% ./snortsnarf.pl -d ../snarf-oos-part3 -rs ../part\ 3/full-oos-parsed
```

The options for SnortSnarf are

`-d <directory>`	<directory> is the path to the directory the HTML pages will be generated in.
`-rs`	reverses the order of signature listing on the first page so that the most interesting signatures appear first

3.3.2.3 Top Talkers

The "Top Talkers" are shown below.

Table 3-6, OOS Top Ten Talkers

Rank	Source IP	# Alerts	# Dsts
1	152.101.81.195	7186	7186
2	MY.NET.28.2	3638	1709
3	64.52.4.180	3558	3554
4	209.116.70.75	1000	3
5	MY.NET.70.183	610	1
6	200.221.192.194	484	1
7	200.221.192.245	380	1
8	MY.NET.165.20	207	1

Rank	Source IP	# Alerts	# Dsts
9	148.65.203.115	62	1
10	148.63.246.3	55	1

Table 3-7, OOS Top Ten Destinations

Rank	Destination IP	Total # Alerts	# Signatures triggered	Originating sources
1	MY.NET.100.217	974 alerts	2 signatures	(3 source IPs)
2	MY.NET.91.81	865 alerts	1 signatures	(3 source IPs)
3	MY.NET.1.4	610 alerts	1 signatures	MY.NET.70.183
4	MY.NET.6.40	449 alerts	1 signatures	(92 source IPs)
5	MY.NET.90.114	208 alerts	1 signatures	MY.NET.28.2, MY.NET.165.20
6	MY.NET.185.48	190 alerts	1 signatures	(58 source IPs)
7	MY.NET.150.133	66 alerts	1 signatures	(3 source IPs)
8	MY.NET.84.245	58 alerts	1 signatures	(3 source IPs)
9	MY.NET.168.105	52 alerts	1 signatures	(6 source IPs)
10	MY.NET.139.230	33 alerts	1 signatures	(6 source IPs)

3.3.2.4 Types of OOS Traffic:

TCP Flags	Number of alerts	Number of Source IP Addresses	Number of Destination Ports
******SF	7187	2	2
12****S*	5450	206	21
********	2861	26	161
**U*P*SF	1634	3	6
****P***	1080	26	8
12***R**	36	2	32
1*UAP*SF	2	2	2
*****RSF	1	1	1
****PRSF	1	1	1
***A**SF	1	1	1
***APRSF	1	1	1
UASF	1	1	1
**UA*RSF	1	1	1
**UAP*SF	1	1	1
1*****SF	1	1	1
1**A**SF	1	1	1
12****SF	1	1	1
12***R*F	1	1	1
12**P*SF	1	1	1
12**PRS*	1	1	1
12*A****	1	1	1
12*AP**F	1	1	1
12*APR**	1	1	1
12*APRSF	1	1	1
12U****F	1	1	1
12U***S*	1	1	1
12U**R**	1	1	1
12U*P*S*	1	1	1
12U*P*SF	1	1	1

TCP Flags	Number of alerts	Number of Source IP Addresses	Number of Destination Ports
12U*PR*F	1	1	1
12U*PRS*	1	1	1
12UA****	1	1	1
12UA*RSF	1	1	1
12UAP***	1	1	1

Table 3-8, Type of OOS

The SYN-FIN OOS packets are a scan on 10/11 from 152.101.81.195 of several /24 networks. The source and destination port for the scan was 21/tcp (ftp). The exception is one packet from 218.164.100.185 to MY.NET.111.214 on port 4662/tcp (eDonkey).

The ”2****S*” packets come from a wide variety of source IP address and go to number o=f different destination ports. On 10/13, 64.52.4.180 did a scan of the /16 probing port 21/tcp. This scan created 3558 of the 5450 entries or 65.3%. Many of the other packets were sent to port 6346/tcp (gnutella) and 4662/tcp (eDonkey). There also an number of packets sent to port 25/tcp (smtp) on MY.NET.6.40 and MY.NET.100.217, MY.NET.139.230.

The “**U*P*SF” OOS packets are a scan by MY.NET.28.2 of the /16 network on 10. The scan was probing ports 21/tcp (ftp), 22/tcp (ssh), 23/tcp (telnet), 25/tcp (smtp), 80/tcp (http) and 139/tcp (netbios-ssn). The two exceptions are a packet on 10/13 from 203.46.103.130 to 255.255.53.8 on port 23/tcp (telnet) and two packet from 255.255.70.97 to255.255.111.44, port 22/tcp (ssh) on 10/15.

Of the Null scan packets (“********”), 99% were from three internal sources MY.NET.70.183 MY.NET.28.2, and MY.NET.165.20. The ports that were being scanned by these machines were 21/tcp (ftp), 22/tcp (ssh), 23/tcp (telnet), 25/tcp (smtp), 37/tcp (time), 80/tcp (http) and 139/tcp (netbios-ssn). The rest of the 161 ports were the from the other 23 source machines.

There is an interesting and unusual series of OOS packets being sent by 68.43.37.13 to MY.NET.70.176. The source IP address doesn’t appear in the alerts or scan logs. This appears to a scan using OOS packets that would not normally be caught by the IDS.

Logs in the modified format for use with SnortSnarf:

```
10/10-22:18:10.649752  [**] OOS [**]  68.43.37.13:0 -> MY.NET.70.176:2247 TCP TTL:112 TOS:0x0
        ID:34896 IpLen:20 DgmLen:177 DF 12UA**** Seq: 0x1A2B00B1  Ack: 0x28D296AC  Win:
        0x5018  TcpLen: 60  UrgPtr: 0x3D2F TCP Options (1) => EOL

10/10-22:25:52.781015  [**] OOS [**]  68.43.37.13:117 -> MY.NET.70.176:2247 TCP TTL:112
        TOS:0x0 ID:40363 IpLen:20 DgmLen:40 DF 12**P*SF Seq: 0x1A2B00B7  Ack: 0x419296BC
        Win: 0x5010  TcpLen: 28 TCP Options (1) => EOL

10/10-22:28:32.159116  [**] OOS [**]  68.43.37.13:34 -> MY.NET.70.176:2247 TCP TTL:112
        TOS:0x0 ID:31949 IpLen:20 DgmLen:40 DF *****RSF Seq: 0x1A2B00B9  Ack: 0xDD796C1
        Win: 0x5010  TcpLen: 32 TCP Options (1) => EOL

10/10-22:38:52.195640  [**] OOS [**]  68.43.37.13:2247 -> MY.NET.70.176:6699 TCP TTL:112
        TOS:0x0 ID:40529 IpLen:20 DgmLen:40 DF 12UAP*** Seq: 0xBF90EB  Ack: 0x96D7  Win:
        0x5010  TcpLen: 0  UrgPtr: 0x2254

10/10-22:41:13.734728  [**] OOS [**]  68.43.37.13:2247 -> MY.NET.70.176:6699 TCP TTL:112
        TOS:0x0 ID:62576 IpLen:20 DgmLen:40 DF ***A**SF Seq: 0xC0D5CE  Ack: 0x96DB  Win:
        0x5010  TcpLen: 20
```

Logs in the original format:

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

10/10-22:18:10.649752 68.43.37.13:0 -> MY.NET.70.176:2247
TCP TTL:112 TOS:0x0 ID:34896 IpLen:20 DgmLen:177 DF
12UA**** Seq: 0x1A2B00B1  Ack: 0x28D296AC  Win: 0x5018  TcpLen: 60  UrgPtr:
0x3D2F
TCP Options (1) => EOL
9C 8F D0 FA D4 1C 8C 2A 46 54 40 A8 C2 DE EE 40  .......*FT@....@
19 AE CE B8 95 D7 1E DE 55 AB B4 31 A8 A7 B6 19  ........U..1....
```

```
93 6A 1F ED DD 43 7D F4 52 16 11 D3 2D CC A5 9C  .j...C}.R...-...
CD B0 E8 C5 55 CB 79 18 FD 68 26 FC 4B FB 8C D8  ....U.y..h&.K...
CA 54 5A 47 A6 BA 15 CD 4A 10 9D 16 05 33 D3 7C  .TZG....J....3.|
7D 66 26 16 E3 6F E0 2D A4 8C 8C E6 75 57 EA EF  }f&..o.-....uW..
93                                               .

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

10/10-22:25:52.781015 68.43.37.13:117 -> MY.NET.70.176:2247
TCP TTL:112 TOS:0x0 ID:40363 IpLen:20 DgmLen:40 DF
12**P*SF Seq: 0x1A2B00B7  Ack: 0x419296BC  Win: 0x5010  TcpLen: 28
TCP Options (1) => EOL

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

10/10-22:28:32.159116 68.43.37.13:34 -> MY.NET.70.176:2247
TCP TTL:112 TOS:0x0 ID:31949 IpLen:20 DgmLen:40 DF
*****RSF Seq: 0x1A2B00B9  Ack: 0xDD796C1  Win: 0x5010  TcpLen: 32
TCP Options (1) => EOL

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

10/10-22:38:52.195640 68.43.37.13:2247 -> MY.NET.70.176:6699
TCP TTL:112 TOS:0x0 ID:40529 IpLen:20 DgmLen:40 DF
12UAP*** Seq: 0xBF90EB  Ack: 0x96D7  Win: 0x5010  TcpLen: 0  UrgPtr: 0x2254

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

10/10-22:41:13.734728 68.43.37.13:2247 -> MY.NET.70.176:6699
TCP TTL:112 TOS:0x0 ID:62576 IpLen:20 DgmLen:40 DF
***A**SF Seq: 0xC0D5CE  Ack: 0x96DB  Win: 0x5010  TcpLen: 20

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

The registration information for 68.43.37.13 from dshield.org:

```
IP Address:                    68.43.37.13
HostName:                      bgp01011611bgs.rockwd01.mi.comcast.net
DShield Profile:
Country:                       US
Contact E-mail:                abuse_AT_comcastpc.com (bounced)
Total Records against IP:
Number of targets:
Date Range:                    to
Ports Attacked (up to 10):     <none listed>
Fightback:                     not sent
```

Whois:
OrgName: Comcast Cable Communications, Inc.
OrgID: CMCS

NetRange: 68.40.0.0 - 68.43.255.255
CIDR: 68.40.0.0/14
NetName: JUMPSTART-MICHIGAN-A
NetHandle: NET-68-40-0-0-1
Parent: NET-68-32-0-0-1
NetType: Reassigned
NameServer: NS01.JDC01.PA.COMCAST.NET
NameServer: NS02.JDC01.PA.COMCAST.NET
Comment:
RegDate: 2002-01-01
Updated: 2002-07-16

TechHandle: IC161-ARIN

TechName: Comcast Cable Communications, Inc.
TechPhone: +1-856-317-7300
TechEmail: cips-ip-registration@cable.comcast.com

There is an interesting and unusual series of OOS packets being sent by 218.164.5.69 to MY.NET.111.214. The source IP address doesn't appear in the alerts logs but appears 12 times as a source in the scan logs. The destination port for all of these packets is 4662/tcp (eDonkey). This appears to a eDonkey client that is having problems. I found no references to eDonkey clients creating OOS tcp packets by search with Google.com.

Logs in the modified format for use with SnortSnarf:

```
10/12-16:23:07.715781  [**] OOS [**]  218.164.5.69:61863 -> 255.255.111.214:4662 TCP TTL:107
    TOS:0x0 ID:12750 IpLen:20 DgmLen:40 DF 12U*PRS* Seq: 0x1CDA9BF  Ack: 0x929048C7  Win:
    0xE979  TcpLen: 8  UrgPtr: 0x6637

10/12-20:06:25.176972  [**] OOS [**]  218.164.5.69:63200 -> 255.255.111.214:4662 TCP TTL:107
    TOS:0x0 ID:57159 IpLen:20 DgmLen:40 DF 1*UAP*SF Seq: 0x2944CB3  Ack: 0x94029E69  Win:
    0x5C0  TcpLen: 8  UrgPtr: 0xE43

10/12-20:06:50.696496  [**] OOS [**]  218.164.5.69:63200 -> 255.255.111.214:4662 TCP TTL:107
    TOS:0x0 ID:52297 IpLen:20 DgmLen:40 DF ***APRSF Seq: 0x2944D0A  Ack: 0x8F50DE4D  Win:
    0x41F  TcpLen: 8

10/12-20:10:07.957709  [**] OOS [**]  218.164.5.69:63200 -> 255.255.111.214:4662 TCP TTL:107
    TOS:0x0 ID:58456 IpLen:20 DgmLen:40 DF 12UA*RSF Seq: 0x2944D16  Ack: 0x37082686  Win:
    0x1A  TcpLen: 56  UrgPtr: 0xD3B9 TCP Options (1) => EOL

10/12-20:15:48.820524  [**] OOS [**]  218.164.5.69:63200 -> 255.255.111.214:4662 TCP TTL:107
    TOS:0x0 ID:52593 IpLen:20 DgmLen:40 DF 12**PRS* Seq: 0x6E163FD0  Ack: 0xC6C45939  Win:
    0x23C9  TcpLen: 60 TCP Options (1) => EOL
```

From the scan logs:

```
Oct 12 19:55:28 218.164.5.69:63200 -> 255.255.111.214:4662 NOACK **U**R**
Oct 12 19:57:52 218.164.5.69:63200 -> 255.255.111.214:4662 INVALIDACK **UA*R*F
Oct 12 20:03:05 218.164.5.69:63200 -> 255.255.111.214:4662 INVALIDACK *2*A*RS* RESERVEDBITS
Oct 12 20:06:03 218.164.5.69:63200 -> 255.255.111.214:4662 UNKNOWN 1**A***F RESERVEDBITS
Oct 12 20:06:25 218.164.5.69:63200 -> 255.255.111.214:4662 INVALIDACK 1*UAP*SF RESERVEDBITS
Oct 12 20:06:31 218.164.5.69:63200 -> 255.255.111.214:4662 NOACK **U**R**
Oct 12 20:06:39 218.164.5.69:63200 -> 255.255.111.214:4662 INVALIDACK ***A*R*F
Oct 12 20:06:50 218.164.5.69:63200 -> 255.255.111.214:4662 INVALIDACK ***APRSF
Oct 12 20:06:58 218.164.5.69:63200 -> 255.255.111.214:4662 NOACK 1*U*P*S* RESERVEDBITS
Oct 12 20:10:07 218.164.5.69:63200 -> 255.255.111.214:4662 INVALIDACK 12UA*RSF RESERVEDBITS
Oct 12 20:13:08 218.164.5.69:63200 -> 255.255.111.214:4662 UNKNOWN 1*UA**** RESERVEDBITS
Oct 12 20:15:48 218.164.5.69:63200 -> 255.255.111.214:4662 NOACK 12**PRS* RESERVEDBITS
```

The other flag combinations of OOS appeared to be random mangling of the packets by the network. There was no apparent correlation between the packets to give the impression that they were a part of a deliberate scan.

3.3.3 Scans

Processing of the scan log files was performed with two Perl scripts. These scripts parsed the logs to identify the Top Talkers Source IP addresses, Top Ten Destination IP addresses and Top Ten destination ports (TCP and UDP). The Perl script, `parsescan.pl`, identifies and counts the number of times an IP address is used as a source IP and as a destination IP. Next, the Perl script, `parse-scan_port.pl`, counted the number of times a destination port was scanned. (See Appendix A for a listing of the Perl code for the two scripts.)

Once the Perl scripts generated the lists, I used BBEdit to sort the ports by the number of scans each port received. BBEdit was also used to split the source list was into both internal and external source lists since the external sources with the highest number of scans was 19th on the overall list.

Table 3-9, Top Tem External and Internal Source Addresses of Scans

Rank	External Source IP Address	Total Count		Rank	Internal Source IP Address	Total Count

1	128.40.166.12	12211
2	63.175.180.250	10920
3	205.162.110.102	10602
4	218.156.189.181	10440
5	198.172.121.81	10173
6	153.19.64.111	10024
7	200.29.128.40	9281
8	166.82.91.27	8345
9	MY.NET.91.237	7572
10	192.116.247.218	6653

1	MY.NET.70.176	285169
2	MY.NET.165.24	254247
3	MY.NET.84.147	182775
4	MY.NET.91.240	152509
5	MY.NET.83.146	135804
6	MY.NET.198.204	125375
7	MY.NET.150.113	98767
8	MY.NET.88.165	92032
9	MY.NET.111.214	76805
10	MY.NET.70.207	68729

Services associated with specific port numbers were investigated through the use of the Internet Assigned Number Authority's port list (http://www.iana.org/assignments/port-numbers), incidents.org (http://isc.incidents.org/port_report.html) and web searches using Google.com.

Table 3-10, Top Ten TCP and UDP Destination Ports of Scans

Rank	TCP Destination Port	Total Count
1	6257 (WINMX)	1070025
2	80 (http)	85384
3	4665 (eDonkey P2P)	53619
4	1 (tcpmux)	45222
5	1214 (kazaa)	40214
6	21 (ftp)	20986
7	53 (dns)	18089
8	27005 (flex-lm)	14997
9	443 (https)	13941
10	25 (smtp)	11680

Rank	UDP Destination Port	Total Count
1	6257 (WINMX)	141418
2	1214 (kazaa)	34565
3	27005(flex-lm)	16250
4	4665 (eDonkey P2P)	9561
5	1 (tcpmux)	4022
6	53 (dns)	3615
7	22321 (**trojan?)	1086
8	6112 (***)	1080
9	17024	970
10	14237	940

** This may be to be a dobol trojan per the discussion on insecure.org (http://lists.insecure.org/lists/incidents/2002/Sep/0056.html) and the information at http://www.simovits.com/trojans/tr_data/y921.html.
*** This is probably the game Diablo running on BattleNet servers (http://cert.uni-stuttgart.de/archive/incidents/2000/03/msg00207.html).

Table 3-11, Top Ten Destination IP Addresses for Scans

Rank	**Destination IP**	**Total Count**
1	204.183.84.240	10979
2	24.120.194.178	7617
3	12.220.145.126	5620
4	12.245.31.155	3936
5	68.39.48.75	3694
6	MY.NET.70.207	2863
7	151.204.131.129	2636
8	146.115.121.119	2486
9	141.149.54.140	2238
10	200.52.195.1	2225

3.4 Host Table

This is a listing of host on the internal network that can be identified as important servers based of the alerts in the log files. A search of the alerts was conducted using BBEdit's grep-like search features. Several types of servers, e.g, DNS, were not identified

Web Servers:

MY.NET.100.165 is obviously a web server based on the alert signature message.

```
10/13-11:42:05.357115 [**] CS WEBSERVER - external web traffic [**]
        64.157.224.115:2280 -> MY.NET.100.165:80
```

FTP Servers:

The CS Web Server (MY.NET.100.165) also appears to be running a FTP server.

```
10/13-12:56:39.955002 [**] CS WEBSERVER - external ftp traffic [**]
        213.140.15.167:2236 -> MY.NET.100.165:21
```

Based on the following alert signature message;

```
10/11-03:25:04.748156 [**] FTP DoS ftpd globbing [**] 80.13.254.146:4043 ->
        MY.NET.100.158:21
```

The machines, MYNET.100.158 and MY.NET.114.116, appear to be FTP servers due to large number of connections to port 21/tcp. There appeared to be several large FTP download sessions from MY.NET.100.158.

Mail Servers:

Based on the alert signature message, MY.NET.6.40 is a SMTP (port 25/tcp) server.

```
10/11-00:51:59.344562 [**] Bugbear@MM virus in SMTP [**] 195.92.193.19:3736 ->
        MY.NET.6.40:25
10/11-17:16:14.341249 [**] Bugbear@MM virus in SMTP [**] MY.NET.6.40:42295 ->
        65.212.73.209:25
10/14-19:16:31.954179  [**] Bugbear@MM virus in SMTP [**] 64.8.50.53:45210 ->
        MY.NET.145.9:25
```

Note that this alert signature was also found in outbound traffic from the internal network:

```
10/12-14:53:13.578638  [**] Bugbear@MM virus in SMTP [**] MY.NET.139.230:2822 ->
        64.156.215.5:25
10/15-23:25:23.626264  [**] Bugbear@MM virus in SMTP [**] MY.NET.144.59:55482 ->
        128.183.107.56:25
```

These machines should be checked for virus infection and install an anti-virus product with current signatures.

Two other machines were noted as having traffic to port 25/tcp and are probably SMTP servers.. The traffic to these machines was picked up by alert signatures due to the choice of source port by the external machine.

```
10/11-00:50:44.137240  [**] High port 65535 tcp - possible Red Worm - traffic [**]
        130.44.1.6:65535 -> MY.NET.145.13:25
10/15-08:21:39.459997  [**] Port 55850 tcp - Possible myserver activity - ref. 010313-1
        [**] 66.180.244.27:55850 -> MY.NET.179.78:25
```

From the OOS logs, MY.NET.12.4 is an IMAP mail server as evidenced by the port 143/tcp (imap) and 993/tcp (imaps)

```
10/11-08:44:59.362404  [**] OOS [**]  MY.NET.12.4:143 -> 255.255.17.56:2430 TCP TTL:255
        TOS:0x0 ID:2476 IpLen:20 DgmLen:40 12***R** Seq: 0x208FACDA  Ack: 0x0  Win: 0x0
        TcpLen: 20
10/11-13:26:46.307774  [**] OOS [**]  MY.NET.12.4:993 -> 255.255.178.72:1222 TCP TTL:255
        TOS:0x0 ID:61439 IpLen:20 DgmLen:40 12***R** Seq: 0x5EBE1EAE  Ack: 0x0  Win: 0x0
        TcpLen: 20
```

Help Desk:

Three help desk machines were easily identified form the alert logs: MYNET.70.49, MY.NET.70.50, MY.NET.83.197.

```
10/12-07:45:11.823424 [**] HelpDesk MY.NET.70.49 to External FTP [**]
        MY.NET.70.49:1041 -> 161.69.211.239:21
10/12-09:48:39.050037 [**] External FTP to HelpDesk MY.NET.70.49 [**] 62.123.114.218:3791
        -> MY.NET.70.49:21
10/12-09:50:08.471092 [**] HelpDesk MY.NET.70.50 to External FTP [**]
        MY.NET.70.50:4123 -> 161.69.201.237:21
```

```
10/12-09:48:38.602878 [**] External FTP to HelpDesk MY.NET.70.50 [**] 62.123.114.218:3709
       -> MY.NET.70.50:21
10/11-15:02:02.015350 [**] HelpDesk MY.NET.83.197 to External FTP [**]
       MY.NET.83.197:1326 -> 161.69.201.238:21
10/15-19:19:36.443218 [**] External FTP to HelpDesk MY.NET.83.197 [**]
       209.240.169.251:1601 -> MY.NET.83.197:21
```

SSH Servers:

One session (source port 22/tcp) to internal port 22/tcp servers was found. MY.NET.163.97 appears to be a rogue SSH server. This machine should be examined immediately for evidence of intrusion and possible compromised

```
10/15-06:05:24.652711  [**] EXPLOIT x86 setuid 0 [**] 198.118.229.166:64495 ->
       MY.NET.163.97:22
10/15-19:35:43.814097  [**] EXPLOIT x86 setuid 0 [**] 137.78.58.62:22 ->
       MY.NET.163.97:4475
10/15-21:36:56.841753  [**] EXPLOIT x86 setgid 0 [**] 137.78.58.62:22 ->
       MY.NET.163.97:4697
10/15-22:55:00.590398  [**] EXPLOIT x86 setuid 0 [**] 137.78.58.62:22 ->
       MY.NET.163.97:4717
```

Three sets of multiple alerts involving port 22/tcp similar to the following were found.

```
10/11-10:15:02.755048  [**] SUNRPC highport access! [**] 129.6.153.67:22 ->
       MY.NET.109.85:32771
10/13-22:04:48.892287  [**] SUNRPC highport access! [**] 128.8.120.85:22 ->
       MY.NET.99.205:32771
10/14-15:24:49.318797  [**] SUNRPC highport access! [**] 68.55.246.114:22 ->
       MY.NET.149.14:32771
```

All three of the sets appear to be outbound SSH sessions from the internal machine where the internal machines' choice of a port resulted in the alert. The reverse DNS information for the external machines:

```
129.6.153.67:      dromio.nist.gov
128.8.120.85:      vidar.umiacs.umd.edu
68.55.246.114:     pcp240040pcs.elictc01.md.comcast.net
```

Telnet servers:

No sessions (source port 23/tcp) to internal telnet servers were found.

DNS servers:

No sessions (source port 53) to internal DNS servers were found. Two instances of NMAP TCP Ping showed connections to MY.NET.137.7. This is a possible DNS server.

```
10/11-18:07:51.614970  [**] NMAP TCP ping! [**] 208.29.51.48:80 -> MY.NET.137.7:53
10/13-17:03:52.757624  [**] NMAP TCP ping! [**] 194.123.157.151:80 -> MY.NET.137.7:53
```

Print Servers:
No sessions (source port 515/tcp) to internal IPP print servers were found. Two external machines were found to be conducting scans for open port 515/tcp.

```
10/12-02:47:50.144447 [**] connect to 515 from outside [**] 141.150.69.195:1137 ->
       MY.NET.133.67:515
10/11-03:14:20.131637 [**] connect to 515 from outside [**] 217.83.3.90:1369 ->
       MY.NET.135.77:515
```

The reverse DNS and whois information for the two external machines:

141.150.69.195:

```
rDNS:   client-141-150-69-195.ba-dsg.net

whois:  CustName:    Hamilton Scientific
        Address:     101 Eisenhower Pkwy Roseland NJ 07068
        Country:     US
        RegDate:     2001-11-03
        Updated:     2001-11-03

        NetRange:    141.150.69.192 - 141.150.69.207
        CIDR:        141.150.69.192/28
        NetName:     VZ-HMLTNSCNTFC-1
```

```
NetHandle:  NET-141-150-69-192-1
Parent:     NET-141-149-0-0-1
NetType:    Reassigned
Comment:
RegDate:    2001-11-03
Updated:    2001-11-03
```

217.83.3.90:

```
rDNS:   pD953035A.dip.t-dialin.net

whois:  netname:      DTAG-DIAL14
        descr:        Deutsche Telekom AG
        country:      DE
        admin-c:      DTIP-RIPE
        tech-c:       ST5359-RIPE
        status:       ASSIGNED PA
        remarks:      ************************************************************
        remarks:      * ABUSE CONTACT: abuse@t-ipnet.de IN CASE OF HACK ATTACKS, *
        remarks:      * ILLEGAL ACTIVITY, VIOLATION, SCANS, PROBES, SPAM, ETC.  *
        remarks:      ************************************************************
        notify:       auftrag@nic.telekom.de
        notify:       dbd@nic.dtag.de
        mnt-by:       DTAG-NIC
        changed:      auftrag@nic.telekom.de 20020108
        source:       RIPE

        route:        217.80.0.0/12
        descr:        Deutsche Telekom AG, Internet service provider
        origin:       AS3320
        mnt-by:       DTAG-RR
        changed:      rv@NIC.DTAG.DE 20001027
        source:       RIPE
```

3.5 External Address Registration Information:

The addresses were chosen from the Top Talker list for each type of log. All addresses in this section were investigated by doing a Google.com search of the GIAC web site. None of the IP addresses returned any hits to correlate to other GCIA practical.

3.5.1 Alerts

3.5.1.1 24.59.33.240:

This address isn't listed on the Alerts Top Talker list for alerts but it would be if all of the alerts from this IP were not deleted prior to being processed by SnortSnarf.

```
OrgName:    ROADRUNNER-NYC
OrgID:      RRNY

NetRange:   24.58.0.0 - 24.59.255.255
CIDR:       24.58.0.0/15
NetName:    RR-NYS-3BLK
NetHandle:  NET-24-58-0-0-1
Parent:     NET-24-0-0-0-0
NetType:    Direct Allocation
NameServer: DNS1.RR.COM
NameServer: DNS2.RR.COM
NameServer: DNS3.RR.COM
NameServer: DNS4.RR.COM
Comment:    ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate:    2001-11-02
Updated:    2002-08-30

TechHandle: ZS30-ARIN
TechName:   ServiceCo LLC
TechPhone:  +1-703-345-3416
TechEmail:  abuse@rr.com
```

Information from dshield.org:

```
IP Address:                 24.59.33.240
HostName:                   syr-24-59-33-240.twcny.rr.com
DShield Profile:
Country:                    US
```

```
Contact E-mail:                abuse@rr.com
Total Records against IP:       1
Number of targets:              1
Date Range:                    2002-11-06 to 2002-11-06
Ports Attacked (up to 10):     <none listed>
Fightback:                     <none sent>
```

No useful information was obtained by searching on Google.com.

3.5.1.2 152.101.81.195:

There are 3064 distinct destination IP addresses in the alerts of the type “SYN-FIN scan!” on this page. This host is also one of the Top Talkers on the OOS list.

```
OrgName:    Hong Kong Internet & Gateway Services Ltd.
OrgID:      HKIGSL

NetRange:   152.101.0.0 - 152.101.255.255
CIDR:       152.101.0.0/16
NetName:    HKNET
NetHandle:  NET-152-101-0-0-1
Parent:     NET-152-0-0-0-0
NetType:    Direct Assignment
NameServer: HK.NET
NameServer: HKIGS.HK.NET
Comment:
RegDate:    1993-09-23
Updated:    2001-07-10

TechHandle: ZP69-ARIN
TechName:   CPCNet Hong Kong Ltd. NOC
TechPhone:  +852-2331-8123
TechEmail:  hostinfo@cpcnet-hk.com
```

Information from dshield.org:

```
IP Address:                    152.101.81.195
HostName:                      152.101.81.195
DShield Profile:

Country:                       HK
Contact E-mail:                chin_AT_HK.LINKAGE.NET (bounced)
Total Records against IP:       29
Number of targets:              28
Date Range:                    2002-10-07 to 2002-10-07
Ports Attacked (up to 10):     <none listed>
Fightback:                     sent to chin@HK.LINKAGE.NET on 2002-05-01 18:11:26
                               no reply received
```

No useful information available on this IP address by searching on Google.com.

3.5.1.3 Networks:

These two networks, 212.179.16.0/18 and 159.226.0.0/16 have been “honored” with a specific Snort rule to monitor their traffic. These networks must have been evaluated as continuing sources of attack attempts and require constant monitoring of their activity.

3.5.1.3.1 212.179.16.0/18

The alert signature message for this network is:

```
10/12-17:32:32.826294 [**] Watchlist 000220 IL-ISDNNET-990517 [**] 212.179.81.54:3866 ->
MY.NET.84.147:80
```

The registration information for this network is:

```
inetnum:      212.179.80.0 - 212.179.94.255
netname:      CABLES-CONNECTION
mnt-by:       INET-MGR
descr:        CABLES-CUSTOMERS-CONNECTION

country:      IL
admin-c:      MR916-RIPE

tech-c:       ZV140-RIPE
status:       ASSIGNED PA
```

```
route:        212.179.64.0/18
descr:        ISDN Net Ltd.
origin:       AS8551
notify:
hostmaster@bezeqint.net
mnt-by:       AS8551-MNT
changed:
hostmaster@bezeqint.net 20020618
source:       RIPE
```

```
remarks:      please send ABUSE complains to
               abuse@bezeqint.net
remarks:      INFRA-AW
notify:       hostmaster@bezeqint.net
changed:      hostmaster@bezeqint.net 20021029
source:       RIPE
```

Information from dshield.org:

```
IP Address:                    212.179.16.0
HostName:                      212.179.16.0
DShield Profile:

Country:                       IL
Contact E-mail:                hostmaster@bezeqint.net
Total Records against IP:
Number of targets:
Date Range:                    to
Ports Attacked (up to 10):     <none listed>
Fightback:                     not sent
```

3.5.1.3.2 159.226.0.0/16

```
10/15-20:54:40.921140 [**] Watchlist 000222 NET-NCFC [**] 159.226.47.236:4059 ->
MY.NET.70.52:80

OrgName:    The Computer Network Center Chinese Academy of Sciences
OrgID:      CNCCAS


NetRange:   159.226.0.0 - 159.226.255.255
CIDR:       159.226.0.0/16
NetName:    NCFC
NetHandle:  NET-159-226-0-0-1
Parent:     NET-159-0-0-0-0
NetType:    Direct Assignment
NameServer: NS.CNC.AC.CN
NameServer: GINGKO.ICT.AC.CN
Comment:    The information for POC handle QH3-ARIN has been reported to
            be invalid. ARIN has attempted to obtain updated data, but
            has been unsuccessful. To provide current contact
            information, please email hostmaster@arin.net.
RegDate:    1992-06-11
Updated:    2002-10-08


TechHandle: QH3-ARIN
TechName:   Qian, Haulin
TechPhone:  +86 1 2569960
TechEmail:  hlqian@ns.cnc.ac.cn


OrgName: The Computer Network Center Chinese Academy of Sciences
OrgID:   CNCCAS
Address: P.O. Box 2704-10,
          Institute of Computing Technology Chinese Academy of Sciences
          Beijing 100080, China
Country: CN
Comment:
RegDate: 1992-06-11
Updated: 1994-07-25
```

Information from dshield.org:

```
IP Address:                    159.226.0.0
HostName:                      159.226.0.0
DShield Profile:

Country:                       CN
Contact E-mail:                hlqian_AT_NS.CNC.AC.CN (bounced)
Total Records against IP:
Number of targets:
Date Range:                    to
Ports Attacked (up to 10):     <none listed>
Fightback:                     not sent
```

3.5.2 OOS

3.5.2.1 209.116.70.75:

This machine appears to be an SMTP server since all connections were to internal SMTP servers. All connections were from a high port on the external machine to port 25 on two internal machines (MY.NET.100.217 and MY.NET.139.230).

```
Business Internet, Inc. ICIX-MD-BLK15 (NET-209-116-0-0-1)
                                  209.116.0.0 - 209.119.255.255
Inflow INFLOW-RDU2-1 (NET-209-116-68-0-1)
                                  209.116.68.0 - 209.116.71.255
Red Hat, Inc. INFLOW-18773-5591 (NET-209-116-70-64-1)
                                  209.116.70.64 - 209.116.70.95

# ARIN WHOIS database, last updated 2002-11-10 19:05

CustName:   Red Hat, Inc.
Address:    4518 South Miami Blvd. Suite #100 Durham NC 27703
Country:    US
RegDate:    2002-09-23
Updated:    2002-09-23

NetRange:   209.116.70.64 - 209.116.70.95
CIDR:       209.116.70.64/27
NetName:    INFLOW-18773-5591
NetHandle:  NET-209-116-70-64-1
Parent:     NET-209-116-68-0-1
NetType:    Reassigned
Comment:
RegDate:    2002-09-23
Updated:    2002-09-23
```

Information from dshield.org:

```
IP Address:                  209.116.70.75
HostName:                    vger.kernel.org
DShield Profile:

Country:                     US
Contact E-mail:              abuse@inflow.com
Total Records against IP:     121
Number of targets:            41
Date Range:                  2002-11-29 to 2002-11-29
Ports Attacked (up to 10):   <none listed>
Fightback:                   not sent
```

Using Google.com, it appears that this machine is definitely a SMTP server for kernel.org. From http://www.cs.helsinki.fi/linux/linux-kernel/2002-44/0237.html:

```
Content-Type: message/rfc822
Content-Transfer-Encoding: 7bit
Content-Disposition: inline

Received: from vger.kernel.org ([209.116.70.75]) by smtp.cwctv.net with Microsoft
SMTPSVC(5.5.1877.447.44);
Sun, 3 Nov 2002 19:06:38 +0000
Received: (majordomo@vger.kernel.org) by vger.kernel.org via listexpand
id <S262322AbSKCTAv>; Sun, 3 Nov 2002 14:00:51 -0500
Received: (majordomo@vger.kernel.org) by vger.kernel.org
id <S262325AbSKCTAv>; Sun, 3 Nov 2002 14:00:51 -0500
Received: from twilight.ucw.cz ([195.39.74.230]:6075 "EHLO twilight.ucw.cz")
by vger.kernel.org with ESMTP id <S262322AbSKCTAo>;
```

3.5.2.2 64.52.4.180:

This machine was responsible for 1546 Queso fingerprint alerts.

```
OrgName:    Eureka Broadband
OrgID:      EBRB

NetRange:   64.52.0.0 - 64.52.255.255
CIDR:       64.52.0.0/16
NetName:    EUREKA-BLK1
NetHandle:  NET-64-52-0-0-1
Parent:     NET-64-0-0-0-0
```

```
NetType:    Direct Allocation
NameServer: AUTH1.EUREKADNS.NET
NameServer: AUTH2.EUREKADNS.NET
Comment:    ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate:    2000-03-06
Updated:    2001-10-12

TechHandle: LH3-ORG-ARIN
TechName:   EurekaGGN
TechPhone:  +1-212-897-8442
TechEmail:  hostmaster@ggn.net
```

Information from dshield.org:

```
IP Address:                 64.52.4.180
HostName:                   nat.64-52-4.180.ip.ebrb.net
DShield Profile:

Country:                    US
Contact E-mail:             ip.admin@eurekabroadband.com
Total Records against IP:    2915
Number of targets:           2813
Date Range:                 2002-10-20 to 2002-10-20
Ports Attacked (up to 10):  <none listed>
Fightback:                  not sent
```

3.5.3 Scans

3.5.3.1 128.40.166.12

```
inetnum:      128.40.0.0 - 128.40.255.255
netname:      UCL-ETHERNET
descr:        University College London
country:      GB
admin-c:      AK4586-RIPE
tech-c:       BL305-RIPE
mnt-by:       RIPE-NCC-NONE-MNT
changed:      tony@noc.ulcc.ac.uk 19911112
changed:      dfk@cwi.nl 19911113
changed:      ripe-dbm@ripe.net 19990706
changed:      ripe-dbm@ripe.net 20000225
source:       RIPE

route:        128.40.0.0/16
descr:        University College London
descr:        Department of Computer
Science
descr:        Gower Street
descr:        London
descr:        WC1E 6BT
descr:        UNITED KINGDOM
origin:       AS786
mnt-by:       JIPS-NOSC
changed:      selina@ans.net 19951011
source:       RIPE
```

Information from dshield.org:

```
IP Address:                 128.40.166.12
HostName:                   mild.physiol.ucl.ac.uk
DShield Profile:

Country:                    GB
Contact E-mail:             andrew_AT_BMADS.UCL.AC.UK (bounced)
Total Records against IP:    1
Number of targets:           1
Date Range:                 2002-10-13 to 2002-10-13
Ports Attacked (up to 10):  <none listed>
Fightback:                  not sent
```

3.5.3.2 63.175.180.250

```
OrgName:    ECTISP
OrgID:      ECTISP

NetRange:   63.175.180.0 - 63.175.180.255
CIDR:       63.175.180.0/24

NetName:    FON-106847948870285
NetHandle:  NET-63-175-180-0-1
Parent:     NET-63-160-0-0-1

NetType:    Reassigned
Comment:
RegDate:    2001-01-07
Updated:    2001-01-07

TechHandle: SG337-ARIN
TechName:   Greenwalt, Steven
TechPhone:  +1-972-923-9090

TechEmail:  steven@ectisp.net

OrgName: ECTISP
OrgID:   ECTISP
Address: PO BOX 2674 WAXAHACIE TX
75165
Country: US
Comment:
RegDate: 2000-02-01
Updated: 2000-02-01
```

Information from dshield.org:

```
IP Address:                 63.175.180.250
HostName:                   63.175.180.250
```

```
DShield Profile:

Country:                        US
Contact E-mail:                 steven@ectisp.net
Total Records against IP:        56
Number of targets:               52
Date Range:                     2002-10-10 to 2002-10-12
Ports Attacked (up to 10):      <none listed>
Fightback:                      not sent
```

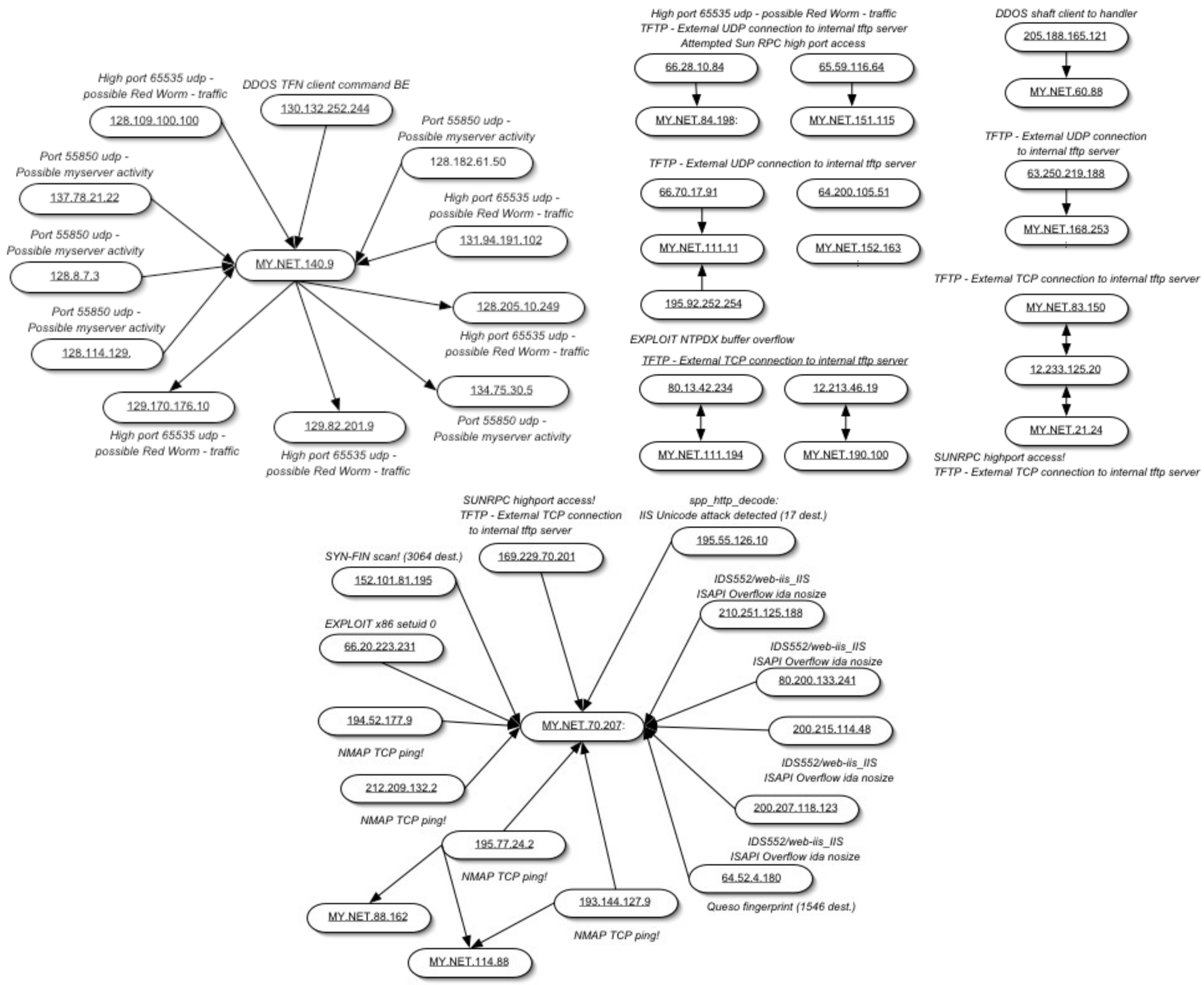


Figure 3-14, Link Analysis Graph

3.6 Link Analysis Graph

Using the top 5 signatures from list of Trojans/DDoS alert signatures (Table 3-2), the link graphs shown below were generated. All destinations for the five signatures were plotted along with the sources of the alert signature. The direction of the arrows indicates direction of packet flow. All of the machines on the internal network should be examined closely for possible compromise. Recovery actions should be taken, if necessary, in accordance with the computer forensics policy of the university.

Two machines, MY.NET.140.9 and, have numerous connections from multiple machines. MY.NET.140.9 is both a source and destination of suspicious activity; an indicator that the machine has been compromised due to the fact that it. This machine should be investigated immediately for signs of probable compromise. MY.NET.70.207 appears to be the destination of but not the source of any alerts. This machine should be investigated for possible compromise.

3.7 Defensive Recommendations

Machines on the internal network that were responsible for DDoS and worm alerts should have the file system, and audit logs examined closely immediately. The operating system should be reinstalled if there are any signs of compromise on the machine.

Internal machines that were responsible for the possible Trojan alert activity should be examined for compromise. The file system, open ports and audit logs should be examined. Recovery actions should be taken, if necessary, in accordance with the computer forensics policy of the university. If necessary, reinstall the operating system and all applications.

The machines that were destination machine of the traffic that generated the exploit alerts should be closely monitored for possible compromise.

There were numerous packets coming form the external network going to port 139/tcp, 135/udp and 137/udp. Access to these ports on windows machines and other machines running CIFS clients or servers is very dangerous. These packets should be blocked at the perimeter firewall. Also these ports can be blocked on the individual machines using personal firewall software (e.g., for Windows machines: Zone Alarm, Tiny Personal firewall, or for *nix boxes running a CIFS client/server: ipfw, iptables, tcpwrappers, etc.).

All server machines and, where possible, the workstations should be subject to vulnerability scan and risk analysis. The results of the risk analysis should be used top prioritize the security upgrades of the systems. This will be a large project for the university's /16 network but is done for the servers it will improve security. For the workstations, the use of the Windows 2000 Gold Standard where possible will help raise the security posture of the network. A similar standard should be generated for the other workstation operating systems and applied across the network.

The IDS system should be modified to use a database for storage and analysis of log data. This will minimize the time and expense of generating tools that are required to analyze data. ACID with MySQL can handle Snort logs and firewall logs and makes an excellent analyst's workstation.

3.7.1 Things to look at further

These machines have some unusual traffic not mentioned elsewhere in the paper but they warrant further investigation:

The machine at MY.NET.114.116 has a suspicious number of alerts with it as the destination.

```
3 different signatures are present for MY.NET.114.116 as a destination

*       1 instances of FTP DoS ftpd globbing
*       4 instances of spp_http_decode: IIS Unicode attack detected
*       6 instances of IDS552/web-iis_IIS ISAPI Overflow ida nosize
```

This machine (MY.NET.190.36) has some suspicious inbound activity. This machine appears to have a number of ports open and has attracted some attention.

```
6 different signatures are present for MY.NET.190.36 as a destination

*       1 instances of SYN-FIN scan!
*       2 instances of External RPC call
*       2 instances of spp_http_decode: IIS Unicode attack detected
*       3 instances of connect to 515 from outside
*       5 instances of IDS552/web-iis_IIS ISAPI Overflow ida nosize
*       13 instances of SMB Name Wildcard
```

```
There are 22 distinct source IPs in the alerts of the type on this page.
```

Resources:

Google.com (http://www.google.com/)

The Internet Ports Database (http://www.portsdb.org/)

Dshield.org (http://www.dshield.org/)

MyNetWatchMan.com (http://www.mynetwatchman.com/)

Sam Spade (http://www.samspade.org/)

Incidents.org (http://www.incidents.org)

Common Vulnerabilities and Exposures (CVE®) (http://cve.mitre.org/)

CERT® Coordination Center (CERT/CC) (http://www.cert.org)

Snort - The Open Source Network IDS (http://www.snort.org/)

Snort Signatures Database (http://www.snort.org/snort-db)

Snort Ports Database (http://www.snort.org/ports.html)

BBedit, Bare Bones Software (http://www.barebones.com/)

Whitehats ArachNIDS Database (http://www.whitehats.com/ids/)

Insecure.org (http://lists.insecure.org/)

Trojan List, Simovits Consulting (http://www.simovits.com/trojans/trojans.html.)

Internet Assigned Numbers Authority (http://www.iana.org/assignments/port-numbers)

American Registry for Internet Numbers (ARIN) (http://www.arin.net/)

Réseaux IP Européens Network Coordination Centre (RIPE) (http://www.ripe.net/)

Asia Pacific Network Information Centre (APNIC) (http://www.apnic.net/)

Geektools.com (http://www.geektools.com/cgi-bin/proxy.cgi)

eDonkey 2000. (http://www.edonkey2000.com/)

So, Hee. GCIA Practical. (http://www.giac.org/practical/Hee_So_GCIA.doc)

Beardsley, Todd. GCIA Practical. (http://www.giac.org/practical/Tod_Beardsley_GCIA.doc)

Walker, Wade. GCIA Practical. (http://www.giac.org/practical/Wade_Walker_GCIA.doc)

Poor, Mike. GCIA Practical. (http://www.giac.org/practical/Mike_Poor_GCIA.doc)

Appendix A

Perl script: `parse-oos.pl`

```
#! /usr/bin/perl

# use strict;

##########
#
#       Author:  Rick Smith
#         Date:  13 Dec 2002
#      Version:  1.0
#
#####
#
#      Purpose:  Parse the OOS log files and convert to tab-delimited
#                for import into Excel spreadsheets.
#
#  Description:  1. Uses the specified oos log file to creates a parsed-<oos-file>.txt
#                   in the current directory which is ready for use with SnortSnarf.
#
#        Usage:  parse-oos.pl <path_to_log>/<oos file>
#
#   Parameters:  1. The oos file to parse including the full path
#
##########

############
## MAIN
############

##Initialize variables.
$logfile = 0;
$resultsfilename = 0;
$linecount = 0;

# Check for null input
if (! $ARGV[0])  {
        die "Usage: parse-oos.pl <path_to_logs>/<oos file>\n\n";
};  #if

## Get the oos file name and path to the log files from command line.
$logfile = @ARGV[0];

open (LOG, $logfile) || warn "Cannot open $logfile : $!";

$resultsfile = ">parsed-". $logfile ."oos.txt";
open RESULTS, $resultsfile || die "Cannot open $resultsfile : $!";



## Parse through the log files.
## Sequentially read in the lines from the log file and print to the RESULTS
## file if the line in the correct format for SnortSnarf
while (<LOG>) {
        $linecount++;  ## increment the line count.
        if (/(\=\+)+$/||/(\w\w\s\w\w\s)+.+$/)
        {  ## throw away the line if a separator line or if hex dump
        }  #if
        elsif (/(\d\d\/\d\d-\d\d:\d\d:\d\d\.\d\d\d\d\d\d\s)(\w.+?\:.+?->\s.+?\s)(.*)/)
        {
                ## put the entry in the correct format
                $line = $1 . " [**] OOS [**]  " . $2 . $3;
                chomp($line);
                ## finalize the previous entry and output the beginning of the entry
                print RESULTS "\n", $line, " ";
        }  #elsif
        else
        {
                chomp($_);
                print RESULTS $_, " ";  ## add the remaining lines of the entry to the
same line
        }  #else
};  #while
```

```
1;
```

Perl script: `parsescan.pl`,

```
#! /usr/bin/perl

use strict;
use Fcntl;
use DB_File;

##########
#
#       Author:  Rick Smith
#         Date:  13 Dec 2002
#      Version:  1.0
#
#####
#
#      Purpose:  Parse the scan log files and convert to colon-delimited list of
#                source and destination IP addresses for import into Excel
#                spreadsheets for further analysis.
#
#  Description:  1. Creates two files that contains a list of the destination IP
#                   address (parse-<scan file>-dst.txt) and a list of the source IP
#                   address (parse-<scan file>-src.txt).
#                2. The files contain the list of the IP address and the number of
#                   times the IP address as was listed in the scan file.
#
#        Usage:  parsescan.pl <path_to_log>/<scan file>
#
#   Parameters:  1. the scan file to parse including the full path
#
##########



############
## MAIN
############


## Declare variables.
my ($logfile, %srcIPlist, %dstIPlist);
my ($k, $v, $resultsfile, %hold);

# Check for null input
if (! $ARGV[0])  {
        die "Usage: parsealert.pl <path_to_logs> <alert-file>\n\n";
};  #if

## Get the machine name and path to the log files from command line.
$logfile = @ARGV[0];
open (LOG, $logfile) || warn "Cannot open $logfile : $!";
print "Opening the logfile $logfile \n";

## Create the databases to hold the source and destination IP address list hashes
tie(%srcIPlist, 'DB_File', undef, O_RDWR|O_CREAT, 0, $DB_BTREE) or die "Can't tie
DB_File: $1\n\n";
tie(%dstIPlist, 'DB_File', undef, O_RDWR|O_CREAT, 0, $DB_BTREE) or die "Can't tie
DB_File: $1\n\n";

while (<LOG>) {
if (/(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}):\d{1,5}\s-
>\s(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})/)
        {
                $srcIPlist{$1}++;
                $dstIPlist{$2}++;
        }  #if
        else
        {
                print "no match!!\n";
        } #else
};  #while

## Create name for parsed and sorted list of Source IP addresses.
$resultsfile = ">parsed-src-" . $day;
open RESULTS, $srcresultsfile || die "Cannot open $srcresultsfile : $!";
```

```
## Output the Destination IP List and count
foreach $k (keys %hold) {
        $srcIPlist{$k} = $hold{$k};
}
print RESULTS "Source IP List :: Count\n";
while (($k,$v) = each %srcIPlist ) {
        print RESULTS "$k :: $v\n";
}

## Create name for parsed and sorted list of Destination IP addresses.
$resultsfile = ">sparsed-" . $logfile . "dst-txt";
open RESULTS, $resultsfile || die "Cannot open $dstresultsfile : $!";

## Output the Destination IP List and count
foreach $k (keys %hold) {
        $dstIPlist{$k} = $hold{$k};
}
print RESULTS "Dest. IP List :: Count\n";
while (($k,$v) = each %dstIPlist ) {
        print RESULTS "$k :: $v\n";
}
1;
```

Perl script: `parse-scan_port.pl`

```
#! /usr/bin/perl

use strict;
use Fcntl;
use DB_File;

##########
#
#       Author:  Rick Smith
#         Date:  13 Dec 2002
#      Version:  1.0
#
#####
#
#      Purpose:  Parse the scan log files and convert to colon-delimited list of
#                destination TCP and UDP ports for import into Excel spreadsheets
#                for further analysis.
#
#  Description:  1. Creates two files that contains a list of the destination TCP
#                   ports (parse-<scan file>-port-TCP.txt) and a list of the
#                   destination UDP ports (parse-<scan file>-port-UDP.txt).
#                2. The files contain the list of the ports and the number of times
#                   the port was listed in the scan file.
#
#        Usage:  parse-scan_port.pl <path_to_log>/<scan file>
#
#   Parameters:  1. The scan file to parse including the full path
#
##########



############
## MAIN
############


## Declare variables.
my ($logfile,$linecount, %udpDstPortlist, %tcpDstPortlist, $udpLinecount, $tcpLinecount);
my ($dstresultsfile, $k, $v, %hold);


## Check for null input
if (! $ARGV[0] )  {
        die "Usage: parse-scan_port.pl <path_to_log>/<alert-file>\n\n";
};  #if


## Get the path and name of log file to parse and then open the file.
$logfile = @ARGV[0];
open (LOG, $logfile) || warn "Cannot open $logfile : $!";
print "Opening logfile $logfile \n";


## Create the databases to hold the TCP and UDP port list hashes
tie(%tcpDstPortlist, 'DB_File', undef, O_RDWR|O_CREAT, 0, $DB_BTREE) or die "Can't tie
DB_File1: $1\n\n";
tie(%udpDstPortlist, 'DB_File', undef, O_RDWR|O_CREAT, 0, $DB_BTREE) or die "Can't tie
DB_File2: $1\n\n";


$linecount = 0;


while (<LOG>) {
        if (/\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}:\d{1,5}\s-
>\s\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}:(\d{1,5})\s(\w{1,3})/)
        {
                if ($2 eq "UDP")
                {
                        $udpDstPortlist{$1}++;
                        $udpLinecount++;
                }
                else
                {
                        $tcpDstPortlist{$1}++;
                        $tcpLinecount++;
                }
        }  #if
```

```
		else
		{
			print "no match!!: $_\n";
		} #else
		$linecount++;
	};  #while

	## Create name for the parsed and sorted TCP ports list.
	$dstresultsfile = ">parsed-" . $logfile . "-ports-TCP.txt";

	open RESULTSDST, $dstresultsfile || die "Cannot open $dstresultsfile : $!";
	print "Creating TCP Port List file: $dstresultsfile \n";

	foreach $k (keys %hold) {
		$tcpDstPortlist{$k} = $hold{$k};
	}
	print RESULTSDST "Total line count: $linecount\n\n";
	print RESULTSDST "TCP line count: $tcpLinecount\n\n";
	print RESULTSDST "Dest. TCP Port List :: Count\n";
	while (($k,$v) = each %tcpdstPortlist ) {
		print RESULTSDST "$k :: $v\n";
	}

	## Create name for the parsed and sorted UDP ports list.
	$dstresultsfile = ">parsed-" . $logfile . "-ports-UDP.txt";

	open RESULTSDST, $dstresultsfile || die "Cannot open $dstresultsfile : $!";
	print "Creating UDP Port List file: $dstresultsfile \n";

	foreach $k (keys %hold) {
		$udpDstPortlist{$k} = $hold{$k};
	}
	print RESULTSDST "Total line count: $linecount\n\n";
	print RESULTSDST "Total UDP line count: $udpLinecount\n\n";
	print RESULTSDST "Dest. UDP Port List :: Count\n";
	while (($k,$v) = each %udpDstPortlist ) {
		print RESULTSDST "$k :: $v\n";
	}
	1;
```