Global Information Assurance Certification Paper

GIAC Certified Intrusion Analyst (GCIA)
Practical Assignment

Version 3.3

Authored By: Mark Faske

December 7, 2003

Table of Contents

Assignment 1: Describe the State of Intrusion Detection

Factors Affecting Human Performance With Intrusion Detection Analysis

Introduction

It's been nearly two decades since the concepts of intrusion detection were first introduced by the likes of Dorothy Denning and Peter Neumann (Hill). As the theory of intrusion detection continued to mature in the mid to late 90s, intrusion detection system (IDS) developers and vendors began to introduce grand visions of IDSs capable of automatically detecting and thwarting attacks. However, through years of IDS implementation in diverse network environments we've learned a few important lessons about IDSs and the related processes to effectively implement these devices. Two of these lessons are:

1) It's much easier to theorize about intrusion detection than it is to implement efficiently and correctly in today's diverse networked environments.

2) Reliable intrusion detection cannot be performed by machines alone. The special cognitive abilities of humans must be fully integrated into the intrusion detection process to realize success.

For many years, organizations have endeavored to remove human analysts from the intrusion detection process. The primary drivers behind these efforts proved to be reduction in costs and the lack of talented individuals to perform intrusion detection work. Undeniably, humans are very expensive to staff, and if there is a better replacement for a human, then the organization should probably move forward with automated intrusion detection. However, intrusion detection technology has not progressed to the point to allow removal of human analysts from the process, despite what many IDS vendors are marketing. Current systems have not exhibited the capability to execute the intrusion detection task on "auto-pilot." On the other hand, it is proven that humans are extremely flexible in conducting analysis and have special capabilities to understand situational awareness that cannot be matched by machines. "Humans are still utilized with IDSs because they induce special cognitive processing capabilities that are not available in artificial intelligence systems or expert systems" (Yurcik).

Given the analysts' important role in the intrusion detection process, it is critical to understand the factors that affect human performance in the intrusion analyst role. Thus, the purpose of this paper is to identify many of the factors affecting analyst performance. This paper does not endeavor to cover the cognitive ability of humans, only the external factors affecting human performance. Any reference to "analyst" or "intrusion analyst" in this paper specifically refers to an Intrusion Detection Analyst.

Modes of Human Information Processing

Prior to discussing the elements affecting human performance with intrusion detection analysis, it is necessary to review the steps that humans undergo to process information. This cycle must be understood because it is the underlying process that must take place for successful analysis and detection to occur. The human information processing sequence can be characterized by several distinct functions, as depicted in Figure 1 below. It is very interesting to note the similarities of this cycle to that of an automated intrusion detection/prevention system. However, there is one big difference, the human cycle is proven to work if implemented correctly in the intrusion detection process.

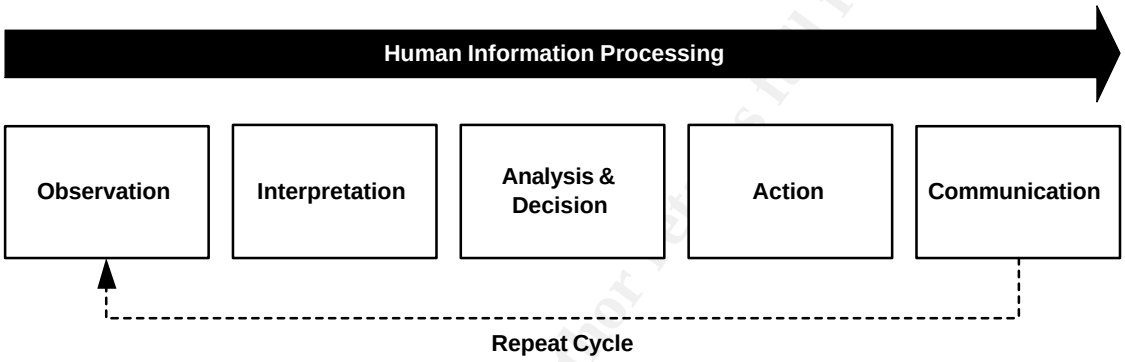


Figure 1. Human Information Processing Cycle (Norwegian University of Science and Technology)

Elements Affecting Human Performance With Intrusion Analysis

Reliable analysis of intrusion detection data by human beings is impacted directly by a number of factors. For ease of discussion, these factors can be rolled-up into major groups that I've called *Elements*. It is the combination of all these elements that can affect a human's ability to reliably perform intrusion detection analysis. In principle, each element may have an adverse effect on the performance of any of the other elements and vice versa, with the human analyst interacting with all the elements. All of the elements would commonly be found in, or affect, a Computer Emergency Response Team (CERT) organization. These elements and their relationship to one another are presented in Figure 2 below.

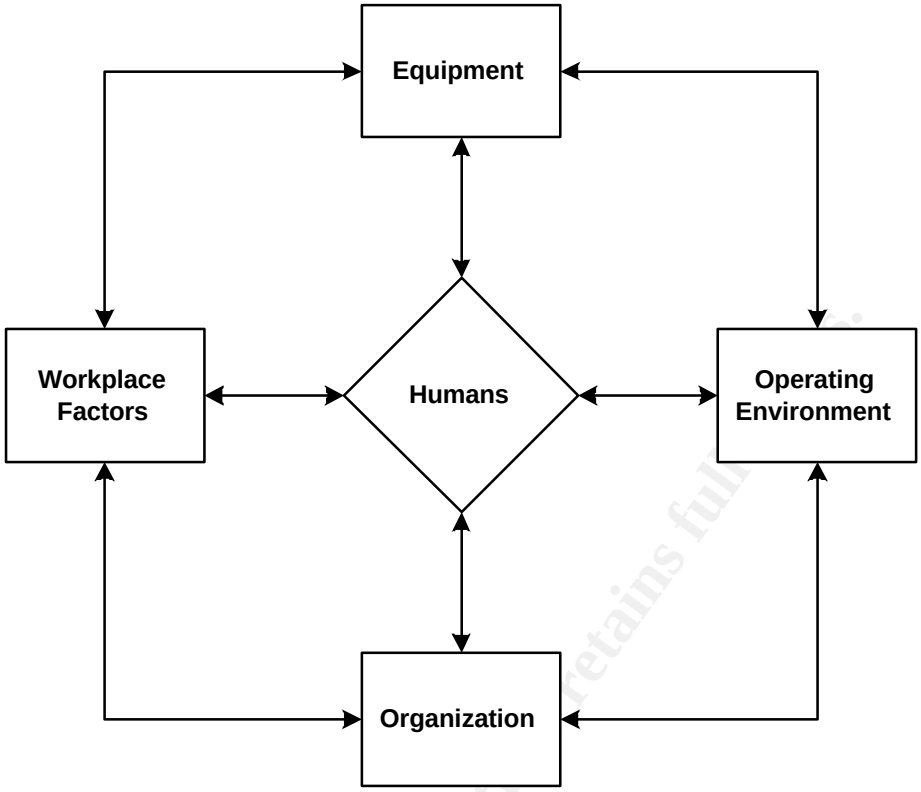


Figure 2. Elements Affecting Intrusion Detection Analysis

Equipment

IDS equipment includes any combination of hardware, software, firmware and related tools used by the system to generate data and present that data to a human analyst for review. For the purposes of this discussion, the type of IDS is somewhat irrelevant. By type, I mean, network-based or host-based, signature-based or anomaly-based, etc. However, the IDS must perform at a level so as not to negatively impact human performance. The IDS must be deployed so that each packet crossing the wire is collected and logged by the system. The remaining information in this section is dedicated to discussing three areas that greatly affect human performance with IDS equipment.

Incorporate Human Factors Into the Initial IDS Design Process

"An effective IT system is best achieved using an integrated product team (ITP) approach from the inception of the system design process. To champion the important role humans play in proper system operation, the IPT should include a human factors specialist or an industrial engineer well versed in the design of systems for humans" (LaSala).

Designing an IT system suitable for human analysis is realized by engineering the system to function with the human abilities that most affect reliable

performance, which are, receipt of visual, auditory, and possible tactile stimuli that initiate task performance; information processing; decision making; and finally responding correctly (LaSala). In other words, humans need more than countless text-based log files to make timely and accurate intrusion analysis decisions. They require information presented in a format that is directed to their sensory capabilities and provides some level of situational awareness. Situational awareness is defined, "As the ability to effectively determine an overall computer network status based on relationships between security events in multiple dimensions" (Yurick).

Visual and auditory stimuli are the most common forms of stimuli designed into today's IDSs. Both of these forms of stimuli have been around for a number of years, but are still fairly limited in their use to humans. It is a well-known fact, that current implementations of these two mediums are frequently overwhelmed due to the sheer number of events generated by IDSs, especially those systems that are ill tuned and produce a large number of false positives.

IDS Visual and Auditory Stimuli

With visual stimuli, IDS developers find it difficult to display the numerous events in a simple fashion that would enable a human analyst to quickly synthesize heterogeneous data, and at the same time, relay some sense of situational awareness to the analyst. We commonly find that events are displayed in some sort of text-based hierarchal structure, which relates the event to a specific signature, anomaly or activity. However, this format does not relate well to human visual capabilities. Humans must still search through listings of events, or drill down, in order to locate information and piece together the situational puzzle.

What is needed, is a visualization capability that can display and link related events and associated data from multiple sensors, thus giving situational awareness to the human analyst. There are several on-going initiatives with intrusion detection data visualization. For an overview of some of the leading solutions to the data visualization problem, please refer to Brian Sheffler's GCIA Practical "The Design and Theory of Data Visualization Tools and Techniques", at http://www.sans.org/rr/papers/30/359/pdf. Mr. Sheffler does a wonderful job at discussing some of the available visualization solutions.

I find that a fully separate discussion of IDS generated auditory stimuli (alarms) is not necessary. This is due to the fact that auditory alarms generated by IDSs simply notify the analyst of an event. The analyst must still review the event as displayed by the system, which would likely be presented in the same hierarchal formats described earlier. In effect, any auditory stimuli must still be evaluated visually by the analyst, which has already been covered in the discussion above.

Data Correlation Capability

The ability to quickly and accurately analyze log data can be directly attributed to

the amount and type of log data produced by the monitored systems. This data is often produced by IDSs (host and network), firewalls, routers, host system logs and application logs. Given the heterogeneous nature of the systems and logs, the problem of correlation becomes very evident. To date, IDS vendors and open source providers have not designed workable tools to assist the analyst with the enormous task of searching through and correlating the mounds of log data produced by these systems. Without a tool to correlate this data, it is simply impossible for the analyst to fully process all the log files. Occasionally what occurs is these logs are gathered and stored, then used for post-incident analysis to uncover exactly how the compromise transpired. If this is occurring, it's too late and the damage has already been done.

Yes, there are tools available, but these tools are often home-grown and not available to the general public. Until reliable commercially available tools are developed to handle massive data correlation, analysts will continue to struggle with this decades old problem.

Operating Environment

The Operating Environment element includes external environmental factors that may/may not be present in the analysts' workplace. Normally the analysts' employer has responsibility for controlling these factors.

"Factors in the operating environment affecting human performance include:

- Ambient noise
- Ambient lighting
- Temperature-humidity combination
- Pressure-oxygen combination
- Vibration

Each of the factors above has a range of values over which human performance is unaffected adversely. At extremes of the range, however, the human will not be functional" (LaSala).

The key is to maintain a steady and comfortable environment that is conducive to positive human information processing. For example, it would not be a good idea to locate your intrusion detection analysts in a data center where there are high amounts of ambient noise, vibration, and low temperatures. All of these negative factors would prove detrimental to effectively processing sensor data. A better location for your intrusion detection analysts would be in a security operations center. This venue would offer the flexibility to somewhat control the analysts surrounding environment, thus making the analyst more productive.

Organization

There are a myriad of organizational issues that can affect the performance of an intrusion detection analyst. Organizationally, it is common for the analyst to work as part of a Computer Emergency Response Team (CERT). An outstanding source of information concerning CERT organizations is the Carnegie Mellon Software Engineering Institute's CERT Coordination Center (CERT/CC) http://www.cert.org/.

Asset and Threat Identification

"Before an organization makes an investment in security technologies, it is important that the organization understand what assets require protection, as well as, the real and perceived threats to those assets" (Carnegie Mellon Software Engineering Institute). Organizations must realize that threats can be internal or external to their organization, with internal threats proving to be much more common.

While all corporate assets require some level of protection, it is necessary for organizations to rank their assets in importance and protect those assets with the appropriate security measures. The determination and ranking of business processes and related assets is usually obtained through the execution of a business impact analysis (BIA). When organizations conduct BIAs, it would be a good idea to include the intrusion detection analysts in this process. This would give the analysts first hand knowledge of the critical systems/data, and also give them a break from the routine of conducting intrusion analysis.

Once the organization understands the threat it faces and determines the assets it wants to protect, it must then relay this information to its staff of intrusion detection analysts. This information will tell the analyst where to focus their analytical energy. Without this information, the analyst will be forced to treat all threats and corporate assets equal, thus severely minimizing his or her effectiveness. Analysts must have the big picture.

Strategy and Operational Plan

Corporate security organizations must communicate to their analysts the CERTs strategy and operational plan. This must be done to ensure analysts' efforts are focused correctly to deliver the desired services to the CERTs customers. Otherwise, the analysts' effort will be severely out of line with the established objectives of the CERT.

A good start to the development of a CERT strategy and operation plan can be generated by posing the simple questions Who, What, When, Where, Why and How. Some examples of the types of things a CERT would want to answer regarding these questions appear below:

Who – Identify your customers and staff. What corporate entities are you

servicing? Who will receive the products and advice of the CERT? Will the CERT utilize full-time or part-time employees?

What – Determine the CERTs mission, goals and required resources. What are the objectives of the CERT?

When – Identify the operational hours of the CERT and when services will be provided, 7X24X365, M-F office hours only, etc.

Where – Identify operational locations for the ID equipment, analysts, etc. Will the CERT structure be centralized or decentralized?

Why – Identify the true need for the CERT. It'll probably deal with the protection of corporate information assets.

How – Determine how the CERT will function on a daily basis. Identify budget, procedures, resources, staff, equipment and infrastructure.

The organizational conclusions to the questions above will directly impact the performance of the analyst on a daily basis. Even though strategy and operational planning are managerial tasks, it would be appropriate to solicit input from the analysts during the initial ramp-up of the CERT and continually throughout the lifecycle of the CERT. CERT missions change regularly, so reviewing operational plans for meeting new objectives must be done regularly.

Management Commitment and Support

"Often the most significant obstacle to the success of an information security improvement initiative is the lack of managerial support" (Carnegie Mellon Software Institute, CERT Coordination Center). This lack of corporate support effectively translates into a lack of authority for the CERT. Managerial commitment and support to the CERT effort should come from the top, meaning either the CEO or CIO.

A good sign of management's support of the CERT organization is the recognition of the CERT in the organizational security policies and procedures and the budget available to the CERT. Policies and procedures should contain specific information on the methods for detecting and responding to information security incident. The policy document should also explicitly state that the CERT should retain the full cooperation of other corporate groups when working to secure the environment. If cooperation from fellow corporate groups is not garnered, then CERT analysts will find it very difficult to remediate security problems on the network. In the long run, this will cause the analysts to become jaded into thinking their work will not make a difference in the security posture of the network.

Policies and Procedures

Ah yes, let's not forget about information security policies and procedures, the backbone of all good information security programs. Specifically, CERT policies and procedures should document the threats you intend to guard against and the actions the corporation intends to take in response to security incidents. The roles and responsibilities of all participating corporate groups and related staff should also be identified, so that misunderstanding of duties does not arise during security incident analysis and investigation.

When dealing with security incidents, the pressure on analysts can be immense. Analysts must ensure their analysis is not only extremely accurate, but timely as well. During these stressful situations, analysts must have some mechanism to refer to that provides instant guidance on how to handle each situation. This will allow the analyst to properly follow all the steps in the incident handling and response process without worrying too much if everything has been done properly. It would probably be a good idea to put the incident handling and response process in checklist format. This provides the analyst a quick tracking mechanism to ensure all steps are completed. This especially becomes critical when organizations choose to involve legal authorities.

Training

In order for organizations to provide applicable training, management must first understand the training requirements for their intrusion detection analysts. To understand these requirements, organizations must correctly determine the type of training needed and the periodicity for that training. Intrusion detection and corresponding incident handling and response are difficult tasks to understand. Therein lies the difficulty with developing and instituting an effective training plan.

> For a complex and evolving technology such as intrusion detection, standard approaches to training (e.g., stand-up or video presentations, computer-based training, and other forms of self-paced tutorials) may not be sufficient. This is due to the need for hands-on experience in analyzing data and tracking patterns of intrusive behavior. The dynamic nature of computer technology in general, and rapidly changing threats and tactics in particular, may stress traditional forms of mentoring and on-the-job training (Carnegie Mellon Software Engineering Institute).

I would also add that any training program for analysts should ensure the analyst develops their thinking and reasoning skills. Sometimes it is extraordinarily difficult to detect an attack or compromise and these simple skills will often need to be utilized to piece together the intrusion puzzle.

The analysts' performance depends partly on the amount and quality of training provided by the employer. It is also likely that analysts will have to pitch in and continue to improve their skills on their own volition. If organizations chose not to

keep their analysts skills current, they can rest assured intrusions will increasingly go undetected, thus endangering corporate information assets.

Workplace Factors

The conditions in which the intrusion analyst is expected to perform work play a key role in the overall effectiveness of the analysts' performance. By workplace factors, I am referring to the situational conditions at the analysts' workplace that directly impact his/her performance. Although these factors may seem rather innocuous, they can become very powerful allies to the organization and staff, or they can become very disruptive if not controlled.

At any time, any single condition may become a particular problem for the organization and analyst. For example, organizations that staff their intrusion detection efforts 24x7x365 must constantly deal with scheduling issues. Organizations and employees must work through weekend staffing, nighttime staffing, holiday staffing, working odd hours, etc. This can ultimately work against analytical performance because analysts will often find themselves battling the work schedule instead of concentrating on intrusion analysis.

One can only imagine how powerful work factors become when many of these factors are negatively impacting the analyst at the same time. If this happens, the organization can expect a sharp decrease in team morale and analytical performance. Together, management and analysts must come to agreeable mediums on these issues, so they do not become detrimental to the intrusion detection effort. Figure 3 on the following page depicts many factors of the workplace that regularly affect analyst performance.

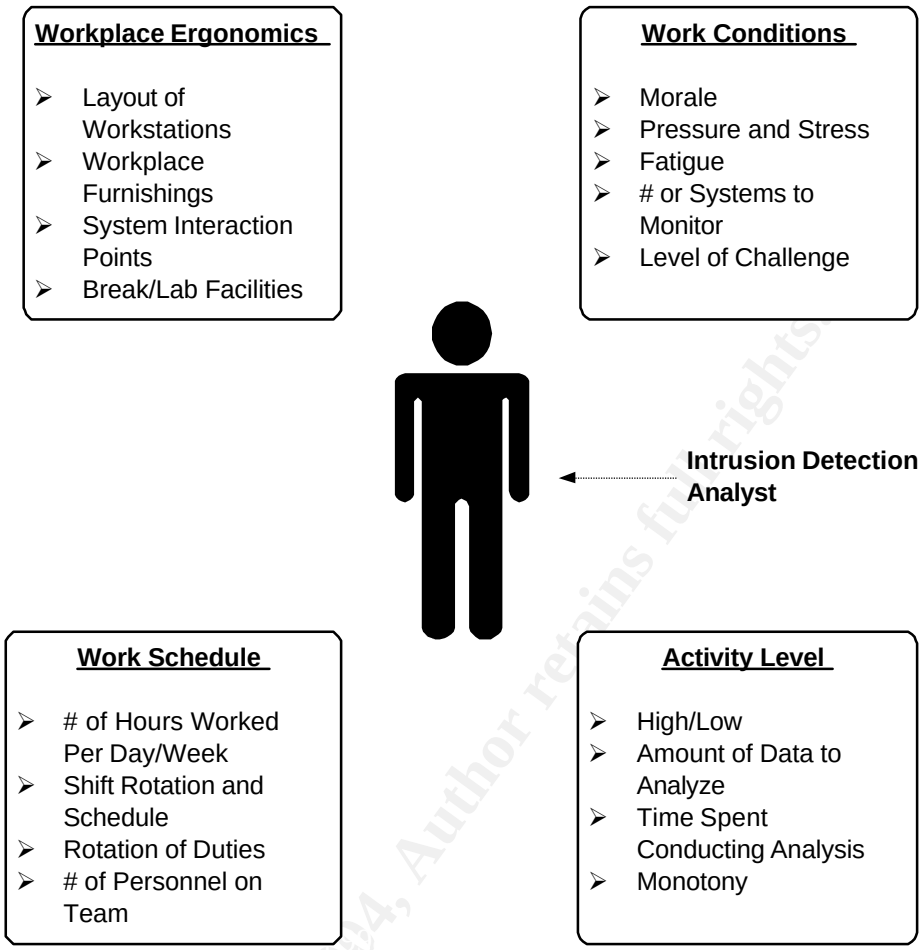


Figure 3. Workplace Factors

Conclusions

There are many conclusions that can be drawn from this paper. I have endeavored to list a few of the most important below.

1. Current IDSs have not yet matured to a level to enable the exclusion of human analysts. Until intrusion detection or prevention matures to the point of removing human analysts, organizations must determine how they will utilize humans in the intrusion detection effort. Organizations will find that they must insou rce, outsource or use a combination of both to meet this task.

2. Organizations must recognize the criticality of the Intrusion Detection Analyst to the intrusion detection process. They must also tailor the process to maximize human cognitive ability.

3. Organizations must recognize the factors that affect human analytical performance and manage those factors to their advantage.

References

Carnegie Mellon Software Engineering Institute. “State of the Practice of Intrusion Detection Technologies.” http://www.sei.cmu.edu/publications/documents/99.reports/99tr028/99tr028chap04.html. (1 Oct 2003).

Carnegie Mellon Software Engineering Institute, CERT Coordination Center. “Creating a Computer Security Incident Response Team: A Process For Getting Started.” http://www.cert.org/csirts/Creating-A-CSIRT.html. (21 Oct 2003).

Carnegie Mellon Software Engineering Institute, CERT Coordination Center. “Establish Policies and Procedures For Responding To Intrusions.” http://www.sert.org/security-improvement/practices/p044.html. (21 Oct 2003).

Carnegie Mellon Software Engineering Institute, CERT Coordination Center. “Staffing Your Computer Security Incident Response Team – What Basic Skills Are Needed?” http://www.cert.org/csirts/csirt-staffing.html. (21 Oct 2003).

Higgins, Kelly Jackson. “Human Element Is Key To Stopping Hackers.” May 29, 2000. http://www.informationweek.com/788/intrusion.htm. (6 Oct 2003).

Hill, Daniel O. “GSEC Practical Assignment, Version number 1.4b, Amended August 29, 2002, “The Human Factor – Adding Intelligence and Action To Intrusion Detection.” http://www.giac.org/practical/GSEC/Daniel_Hill_GSEC.pdf. (4 Nov 2003).

Innella, Paul, McMillan, Oba, Trout, David, Bace, Rebecca. “Managing Intrusion Detection Systems in Large Organizations, Part One.” April 4, 2002. http://www.securityfocus.com/infocus/1564. (21 Oct 2003)

LaSala, Kenneth P., Ph.D. “Designing Information Technology Systems for Reliable Human Performance.” Third Quarter-2000. http://rac.alionscience.com/pdf/3RD_Q2000.pdf. (24 Sept 2003).

Norwegian University of Science and Technology. “Thematic Network For Safety Assessment of Waterborne Transport.” March 2001. http://projects.dnv.com/themes/Deliverables/d1_2FINALv2.pdf. (28 Sept 2003).

Phung, Mahn. “Intrusion Detection FAQ, Data Mining In Intrusion Detection.” October 24, 2000. http://www.sans.org/resources/idfaq/data_mining.php. (7 Oct 2003).

SANS. “Intrusion Detection FAQ, The Importance of Intrusion Protections.” August 1, 2000. http://www.sans.org/resources/idfaq/ipe.php. (7 Oct 2003).

Sheffler, Brian. “The Design and Theory of Data Visualization Tools and Techniques.” 2001. http://www.sans.org/rr/papers/30/359/pdf. (6 Oct 2003).

Yurcik, William, Barlow, James, Jeff Rosendale. “Maintaining Perspective On Who The

Enemy in the Security Systems Administration of Computer Networks." http://www.ncsa.uiuc.edu/People/jbarlow/publications/Yurcik_Barlow_Rosendale_WhoIsTheEnemy.pdf. (25 Sept 2003).

Assignment 2: Network Detects

Network Detect #1 - Code Red Virus

1. Source of Trace

This network trace was extracted from the GCIA raw logs available for download at http://www.incidents.org/logs/Raw/2002.4.16. According to the READ ME file posted to this site, "The log files are the result of a Snort instance running in binary logging mode (Incidents.org)." The website or READ ME file gave no indication as to the origin of the log files. The home network for the traffic contained in this log file appears to be 78.37.0.0/16, as an address from this Class B is present in every packet. The 78.37.0.0/16 range resolves to IANA Reserved, which supports the fact that internal addresses were obfuscated.

The system hosting the Snort instance appears to be placed between devices with Media Access Control (MAC) addresses: 00:03:e3:d9:26:c0 and 00:00:0c:04:b2:33. These two MAC addresses were present in every Ethernet frame collected by the Snort instance. Ethereal resolved both of these MAC addresses to Cisco devices, most likely routers. The external probably serves as a border router connecting the organization to an ISP and the internal router likely functions as the connection point for the local network.

2. Detect Was Generated By

To generate alerts from the 2002.4.16 file, I utilized Snort Version 2.0.2-ODBC-MySQL-Win32 (Build 92) running on a Windows XP platform. I enabled all rules in the Snort configuration file. I used the following command line string to process the file:

snort -d -A full -c C:\Snort\etc\snort.conf -l C:\Snort\log_2002.4.16 -k none -r C:\Snort\2002.4.16 -X –e

Explanation of Snort Switches

-d	Dumps the application layer
-A full	Sets the alert mode, Generates full alert information
-c	Location of the Snort configuration file
-l	Location to save logs
-k none	No checksum validation performed
-r	Specifies the input file
-X	Display ASCII and Hex information
-e	Display 2nd layer header data

My first evaluation of the Snort created “Alert.doc” file uncovered nothing of particular interest, so I opted to open the folders created for each IP address, to get a look at the packet payloads. About halfway through my analysis, I discovered the redundant “NNNNNs…” in a packet payload, which is characteristic of the Code Red virus. This was interesting because Snort did not create an alert for Code Red. However, Snort did create a `BAD-TRAFFIC` alert for the same packet. The packet trace that produced the `BAD-TRAFFIC` alert appears below:

```
[**] BAD-TRAFFIC bad frag bits [**]

05/16-02:55:36.884488 0:3:E3:D9:26:C0 -> 0:0:C:4:B2:33 type:0x800 len:0x5CA
80.2.252.223 -> 78.37.147.200 TCP TTL:108 TOS:0x0 ID:51580 IpLen:20
DgmLen:1468 DF MF

Frag Offset: 0x0000   Frag Size: 0x0014
0x0000: 00 00 0C 04 B2 33 00 03 E3 D9 26 C0 08 00 45 00  .....3....&...E.
0x0010: 05 BC C9 7C 60 00 6C 06 39 35 50 02 FC DF 4E 25  ...|`.l.95P...N%
0x0020: 93 C8 0C C0 00 50 BC FA E0 FF 55 E0 5E DA 50 18  .....P....U.^.P.
0x0030: 44 70 21 A9 00 00 2F 64 65 66 61 75 6C 74 2E 69  Dp!.../default.i
0x0040: 64 61 3F 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  da?NNNNNNNNNNNNN
0x0050: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
0x0060: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
0x0070: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
0x0080: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
0x0090: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
0x00A0: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
0x00B0: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
0x00C0: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
0x00D0: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
0x00E0: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
0x00F0: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
0x0100: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
0x0110: 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E 4E  NNNNNNNNNNNNNNNN
0x0120: 4E 4E 4E 25 75 39 30 39 30 25 75 36 38 35 38 25  NNN%u9090%u6858%
0x0130: 75 63 62 64 33 25 75 37 38 30 31 25 75 39 30 39  ucbd3%u7801%u909
0x0140: 30 25 75 36 38 35 38 25 75 63 62 64 33 25 75 37  0%u6858%ucbd3%u7
0x0150: 38 30 31 25 75 39 30 39 30 25 75 36 38 35 38 25  801%u9090%u6858%
0x0160: 75 63 62 64 33 25 75 37 38 30 31 25 75 39 30 39  ucbd3%u7801%u909
0x0170: 30 25 75 39 30 39 30 25 75 38 31 39 30 25 75 30  0%u9090%u8190%u0
0x0180: 30 63 33 25 75 30 30 30 33 25 75 38 62 30 30 25  0c3%u0003%u8b00%
0x0190: 75 35 33 31 62 25 75 35 33 66 66 25 75 30 30 37  u531b%u53ff%u007
0x01A0: 38 25 75 30 30 30 30 25 75 30 30 3D 61 20 20 48  8%u0000%u00=a  H
0x01B0: 54 54 50 2F 31 2E 30 0D 0A 43 6F 6E 74 65 6E 74  TTP/1.0..Content
0x01C0: 2D 74 79 70 65 3A 20 74 65 78 74 2F 78 6D 6C 0A  -type: text/xml.
0x01D0: 48 4F 53 54 3A 77 77 77 2E 77 6F 72 6D 2E 63 6F  HOST:www.worm.co
0x01E0: 6D 0A 20 41 63 63 65 70 74 3A 20 2A 2F 2A 0A 43  m. Accept: */*.C
0x01F0: 6F 6E 74 65 6E 74 2D 6C 65 6E 67 74 68 3A 20 33  ontent-length: 3
0x0200: 35 36 39 20 0D 0A 0D 0A 55 8B EC 81 EC 18 02 00  569 ....U.......
```

***Note: Packet trace was truncated after the important information appeared in the trace.*

The Snort signature that triggered the alert is:

```
alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"BAD-TRAFFIC bad frag bits";
fragbits:MD; sid:1322; classtype:misc-activity; rev:5;)
```

To see the "BAD-TRAFFIC bad frag bits" rule trigger on a Code Red infected packet was interesting and required further analysis. I fully expected to see an alert from the "Web IIS" rule group fire, which would have indicated the presence of Code Red. Either of the two Snort signatures below could have fired on the Code Red infected packet:

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"WEB-IIS ISAPI
.ida access"; uricontent:".ida"; nocase; flow:to_server,established;
reference:arachnids,552; classtype:web-application-activity;
reference:cve,CAN-2000-0071; reference:bugtraq,1065; sid:1242;  rev:6;)

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

The "BAD-TRAFFIC bad frag bits" alert fired because an external address transmitted a packet to the home network, in this case (78.37.0.0/16), with both the Don't Fragment (DF) and More Fragments (MF) flags set. It is not valid to have both flags set simultaneously, so there is a high probability the packet was crafted. In this case, it seems the crafting was enough to fool Snort into triggering on the fragment flags instead of the more important Code Red virus. In effect, this is a False Negative situation and could be considered a clever attempt to evade the intrusion detection device.

3. Probability The Source Address Was Spoofed

I find it highly unlikely that the source address that sent this Code Red attack was spoofed. Given the propagation characteristics of the Code Red virus, there's no reason to spoof addresses. A more likely scenario is the system was simply infected with the Code Red virus and was attempting to propagate the virus to other systems, in a somewhat random manner. The following paragraph provides some excellent insight from eEye® Digital Security on exactly how the Code Red worm propagates.

> The worm spreads itself by creating a sequence of random IP addresses. However, the worm's list of IP addresses to attack is not all together random. In fact, there seems to be a static seed (a beginning IP address that is always the same) that the worm uses when generating new IP addresses. Therefore every computer infected by this worm is going to go through the same list of "random" IP addresses. Because of this feature, the worm will end up re-infecting the same systems multiple times, and traffic will cross traffic back and forth between hosts ultimately creating a denial-of-service type effect (eEye® Digital Security).

4. Description of Attack

The Code Red virus attempts to attack vulnerable Microsoft Internet Information Server (IIS) versions 4 and 5, via a remotely exploitable buffer overflow of the IIS server's Internet/Indexing Service Application Programming Interface (ISAPI). This interface is vulnerable because it does not conduct appropriate input bounds checking on its buffer. An overflow of this buffer would allow an attacker to run arbitrary code of his/her choice at the system level.

The Code Red worm is also capable of successfully attacking Cisco Products with integrated IIS Servers and Series 600 DSL Routers. The Series 600 Routers are vulnerable to the Code Red virus because of vulnerabilities found in the CBOS operating system run by this router series. If attacked successfully, these routers will stop forwarding packets, causing a denial of service. Since the problem with Cisco DSL Routers was much less prevalent, the rest of my analysis will focus on vulnerable IIS servers only.

The Common Vulnerabilities and Exposures (CVE) website (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0500) references the ISAPI overflow vulnerability as CVE-2001-0500.

Additional information about the Code Red virus can be found at the following URLs:

http://www.cert.org/advisories/CA-2001-19.html.
http://www.microsoft.com/technet/security/bulletin/MS01-033.asp?frame=true.
http://www.cert.org/advisories/CA-2001-12.html.
http://www.securityfocus.com/bid/2880.
http://www.cisco.com/warp/public/707cisco-code-red-worm-pub.shtml.

5. Attack Mechanism

The Code Red virus first attempts to establish a HTTP session with any machine that is listening on TCP port 80 via HTTP port probing. Once Code Red finds a machine listening on port 80, it will then try to pass the crafted string below to the ISAPI buffer via an HTTP GET request.

```
HTTP GET
/default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN%u9090
%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801%u9090
%u9090%u8190%u00c3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a
```

If the machine is running a vulnerable IIS server, the string will then execute by the machine and create a buffer overflow condition of the ISAPI. It should be noted that IIS Servers do not have to be running the Index Server to get infected. As long as the .idq. or .ida files are present, and the attacker has established a web session, the machine

could be exploited (Carnegie Mellon Software Engineering Institute, CERT® Coordination Center). Once a machine becomes infected with the Code Red virus, it attempts to accomplish two primary goals:

A. Deface web pages hosted by the infected machine with "Welcome to http://www.worm.com!, Hacked By Chinese!"

B. Launch a coordinated denial of service attack against 198.137.240.91 (http://www.whitehouse.gov./).

6. Correlations

The search to correlate my findings led me to Brian Cahoon's GCIA Practical (Cahoon). Mr. Cahoon utilized raw log file 2002.5.15 for his practical, which contained traffic similar to the file I analyzed (2002.4.16). His findings supported the problems I encountered with Snort alerting to the multiple fragment flags, and not the Code Red virus.

Web searches on 8 Nov 2003 also revealed two posts to the Security Focus mailing list that support my analysis. Not enough information was available to attribute the posts to a specific person, so I have included the web links without personal reference. Both of these posts allude to Snort missing the Code Red in favor of alerting on the fragment flags.

http://cert.uni-stuttgart.de/archive/intrusions/2003/02/msg00055.html.
http://cert.uni-stuttgart.de/archive/intrusions/2002/08/msg00106.html.

7. Evidence of Active Targeting

Relating back to my answer in Number 3 above, it is very unlikely this Code Red attack specifically targeted destination IP 78.37.147.200. The Code Red virus was designed to massively propagate via the Internet to infect as many machines as possible. Traffic analysis of the file 2002.4.16 indicates that 78.37.147.200 was sent only one packet and this packet was from 80.2.252.223. This was the Code Red packet I used for the network detect. Source IP 80.2.252.223 sent only one other packet to a machine in home network 78.37.0.0/16. However, this packet was not a Code Red packet. The fact that the source IP only targeted 78.37.147.200 with Code Red indicates this attack could be the result of active targeting, but given the random method of propagation by Code Red, I believe 78.37.147.200 was simply a randomly targeted host.

8. Severity

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

Severity = (3 + 5) – (3 + 3)

Resulting Severity = 8 – 6 = 2.

***Criticality* – 3.** There is no way to deduce the criticality of the targeted system, as the

system only received one packet in the entire 2002.4.16 logfile. This packet was an unsolicited Code Red packet. Much more traffic analysis would have to take place to determine the function of 78.37.147.200. Additionally, the targeted system sent no packets to indicate the function of the machine. Therefore, I gave Criticality a three, just to be on the safe side.

Lethality **– *5.*** Even though Code Red virus is not particularly destructive, Code Red infections result in a system level compromise. Thus, I rated lethality a five.

System Countermeasures **– *3*.** There is no way to deduce the system countermeasures of the targeted system, because the system did not interact with any other systems in the 2002.4.16 file. The lack of packets coming from the system is actually a good sign, as packets coming from this machine on Port 80 to random IP addresses would indicate possible Code Red infection. Therefore, I gave System Countermeasures a three, just to be on the safe side.

Network Countermeasures **– *3.*** The external MAC address (00:03:e3:d9:26:c0) correlates to a Cisco device, which is likely a border router that performs some level of packet filtering. Thus, some form of external protection is likely available. This router would not have been configured to block port 80 traffic, thus the Code Red attack would have gotten through. Traffic analysis indicates that the border device is probably not running a "deny by default" policy, as several ports are getting through that probably shouldn't be, such as, 1080, 3128, 515, etc. I gave Network Countermeasures a three since there is some perimeter protection, but it doesn't appear to be configured very well.

9. Defensive Recommendations

Obviously, the organization isn't going to block TCP port 80 at their filtering router or firewall, as that would defeat the purpose of being connected to the Internet in the first place. However, once the propagation method of Code Red was discovered, it certainly would not have been out of the question to disconnect from the Internet until the virus was brought under control. Some organizations actually do this as part of their security response to massive port 80 viruses. The organization should also consider using a webserver that does not contain the number of default vulnerabilities as IIS.

The Code Red virus can be stopped through the application of system patches in a timely manner. The Code Red patch is available from Microsoft at http://www.microsoft.com/Windows2000/downloads/critical/q300972/download.asp.

Along with patching, the organization should ensure their machines are deployed with only the necessary applications and services running. This will serve to limit the number of machines susceptible to a range of vulnerabilities. Control of application deployment can be accomplished through controlled configuration management and deployment programs.

Certainly, the organization should also ensure all their Windows systems are running

anti-virus software with current DAT files to thwart future virus attacks.

10. Multiple Choice Test Question

Question. It has been shown that Snort fired the "BAD-TRAFFIC bad frag bits" rule instead of one several Web IIS rules that indicate the presence of Code Red? Why didn't Snort alert on one of the Web IIS "Code Red" rules?

A. Snort's "Web IIS" Rules are incapable of detecting the Code Red virus.

B. Snort has a problem detecting the Code Red virus when both the DF and MF flags are set.

C. Snort's "Web IIS" Rules were not enabled.

D. Snort's "Bad Traffic" rule set takes precedence over the "Web IIS" rule set.

Answer. B

11. Submission To Incidents.org Mail List

I submitted Network Detect #1 to Incidents.org on 16 Nov 2003. I received two responses. All comments by responders are actual quoted text, thus they have not been altered for spelling, grammar, spacing, etc.

Response #1

The first response I received was from Holger van Lengerich. Mr. Lengerich responded with the following:

> "DANGER, WILL ROBINSON!
>
> "Number Of Students Terminated/Revoked for Plagiarism or Other Ethics Violations: 90
> Don't make yourself number 91!
> Read the Administrivia and all other directions very carefully."
> (http://www.giac.org/administrivia.php or
> http://www.giac.org/COE.php)
>
> IMHO you have just released your work to the public without giving proper credits to your sources. :-O
>
> Not to say that all GCIA students, which want to use your detect as reference, will not be able to look at your sources until GIAC publishes your paper."

Mr. Lengerich responded in this manner because I did not include my references in the submission to the mail list. I explained in my initial submission to the list that I had not included references to decrease the length of submission, however I had cited all references in my actual paper. My response to Mr. Lengerich is below:

> "Thanks for your comments. After reading your comments, I agree with your position. I'm not

exactly sure what the rules are regarding posting to mailing lists, but I guess the policy of "Better Safe Than Sorry" is applicable here. Especially given the problems some students have experienced with "Referencing."

Point taken!"

Response #2

I received my second response from Joe Bowling. Mr. Bowling's questions/statements were imbedded throughout the e-mail submission, so I extracted his comments below and answered each comment.

Comment 1: "Good idea to explain the command line switches." (Snort Switches)
My Response 1: "I agree with your comment. I actually removed the switch information from my paper because it was getting too lengthy, however I've put it back in. The Snort switches appear below.

-d	Dumps the application layer
-A full	Sets the alert mode, Generates full alert information
-c	Location of the Snort configuration file
-l	Location to save logs
-k none	No checksum validation performed
-r	Specifies the input file
-X	Display ASCII and Hex information
-e	Display 2nd layer header data

Comment 2: "Next time feel free to use brievity." (Length of packet trace too long)
My Response 2: "I looked up *"brievity "* in the dictionary and could find no reference. Based on the context of your comment, I think you meant *"brevity"* which means shortness of duration. So, I think you're recommending that I chop off the packet at the important part, instead of submitting the entire packet. I wasn't aware that we could do this, but I like the idea given I could use the space. I concur with your comment."

Comment 3: "On page 131 of the Snort book from the Track 3 courseware Rules Are actually grouped by action first (alert, log etc) Next the rules are grouped by the rule header. Then options are checked in the order they are entered. Then Looking at the the rules in order for the WEB-IIS ISAPI alert to be triggered the the session has to be established before it will trigger....and the Bad bits rule does Not require the session to be established that is why it gets triggered first. I would be intrested if you passed the bad bits rule in the configs if the ISS alert would then trigger?
My Response 3: I commented out the bad-traffic rules as you suggested and now Snort doesn't alert at all on the Code Red infected packet. I fully expected to see Snort generate a Web-IIS alert. I double-checked the Snort .conf file to ensure that I had the Web-IIS rules uncommented and they were. Given the results of this test, I'm not exactly sure what is happening to give Snort problems with alerting on the Code Red packet.

Comment 4: "Don't you think that patching might be reasonable?"

My Response 4: Yes, I agree patching is a valid defensive recommendation. After reviewing the section I can see how you might have come to the conclusion that I wasn't recommending patching. In fact, I was recommending patching in combination with controlled application deployment. I will restate the section to better explain my point."

Network Detect #2 - DDOS Tribal Flood Network

1. Source of Trace

The trace below was collected from a corporate network that employs a number of Snort sensors. Generally, the sensors are placed at geographically disparate corporate locations that require Internet access. Sensor names and internal IP addresses have been sanitized to protect the privacy of the network. Sensor names have been changed to NIDxxx and internal IP addresses take the form of my.net.xxx.xxx.

```
Generated by ACID v0.9.6b22 on Mon October 06, 2003 05:23:01

---------------------------------------------------------------------------
NIDxxx [2003-10-05 20:00:36] [arachNIDS/183] [snortDB/251]  DDOS - TFN client
command LE
IPv4: 202.232.194.35 -> my.net.42.91
      hlen=5 TOS= dlen=92 ID=8927 flags= offset= TTL=112 chksum=57328
ICMP: type=Echo Reply code=0
      checksum=50701 id= seq=
Payload:  length = 64

000 : AD 47 80 3F 5F 75 0D 00 08 09 0A 0B 0C 0D 0E 0F   .G.?_u..........
010 : 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F   ................
020 : 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F    !"#$%&'()*+,-./
030 : 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F   0123456789:;<=>?

Response: none
```

Figure 4 on the following page depicts the general connectivity of any single NID on the network.

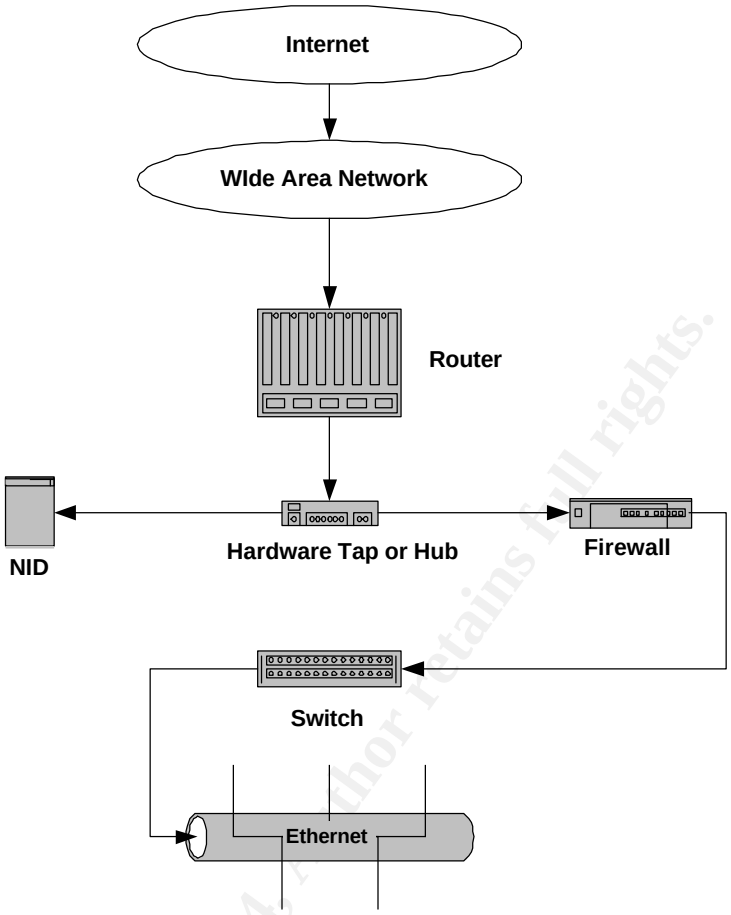


Figure 4. Network Diagram

2. Detect Was Generated By

The alert was generated by Snort version 2.0.0 Build 72 running on a Solaris platform. The alerts were parsed into an MySQL database and viewed through ACID v0.9.6b22. I chose to export the alerts out of ACID and into text format for ease of use in this practical. The Snort sensor was running the standard ruleset, however the ruleset had undergone a moderate amount of tuning to eliminate false positives and extraneous alerts. A fair amount of custom signatures were also added to the ruleset, especially with regard to detecting viral activity. The signature that alerted to the Tribal Flood Network (TFN) activity is a standard Snort signature and had not been modified in any fashion.

The following Snort signature generated the alert:

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"DDOS - TFN client command
LE"; itype: 0; icmp_id: 51201; icmp_seq: 0; reference:arachnids,183;
classtype:attempted-dos; sid:251; rev:1;)
```

This Snort signature fires when it detects a packet with the following rule header and rule option characteristics:

Rule Header

- *Protocol* – ICMP
- *Source IP Address* – any address external (variable $EXTERNAL_NET) to the defined home network (variable $HOME_NET)
- *Destination IP Address* – any address residing in the defined home network ($HOME_NET)
- *Ports* – Any source or destination port, represented by "any" in both cases
- *Direction of Traffic Flow* – From the source address to the destination address, represented by "->"

Rule Options

- *ICMP Type* – 0 (Echo Reply)
- *ICMP ID* – 51201 ("The icmp_id 51201 actually correlates to command value "456" due to different architecture (Whitehats)." Value 456 equates to "Spawn a Shell." **This is the key characteristic identifying the packet as TFN related.** All other characteristics are synonymous with a normal ICMP Echo Reply packet.
- *ICMP Sequence Number* - 0

3. Probability The Source Address Was Spoofed

The Tribal Flood Network operates in a tiered architecture where the attacker(s) control multiple Client machines. In turn, the Client machines control many more Daemon machines.

In the trace, we have a Client machine trying to issue a *spawn shell* command to a Daemon machine. If the Client is to receive any communications back from the Daemon, then the source address must be legitimate. Therefore, I believe the source address has not been spoofed. A Whois query of source address 202.232.194.35 indicates the address is registered to the AGENDA Corporation in Japan. Thus, a machine at the AGENDA Corporation appears to be functioning as a TFN Client that is likely controlling several TFN "Daemon" machines. One of those Daemon machines is the destination host in the trace above, my.net.42.91.

JPNIC Whois Query:

a. [Network Number]	202.232.194.0
b. [Network Name]	AGENDA-NET
g. [Organization]	AGENDA Corporation
m. [Administrative Contact]	SK290JP

n. [Technical Contact]	TS6271JP
p. [Nameserver]	ns1.agenda.co.jp
p. [Nameserver]	ns2.agenda.co.jp
y. [Reply Mail]	apply@iij.ad.jp
[Assigned Date]	2000/08/11
[Return Date]	
[Last Update]	2002/06/13 10:10:34 (JST) takegu@sapporo.iij.ad.jp

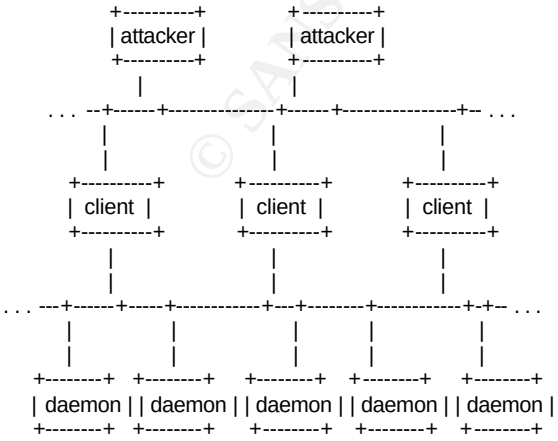
4. Description of Attack

The Tribal Flood Network is a program that specifically targets high-speed Unix and Linux systems that are continually connected to the Internet. These machines are targeted so they can be turned into operational Daemon machines, used to launch coordinated DDOS attacks against specified Internet hosts. For the DDOS to be successful, a sufficient amount of network or computing resources must be consumed, so the target machine will not be able to respond to legitimate traffic. The attack, written by Mixter, was thought to be behind the DDOS attacks against Yahoo.com, CNN.com and Buy.com. in early 2000. The Common Vulnerabilites and Exposures website references TFN as CAN-2000-0138.

http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0138.

The TFN attack is executed when the attacker sends specific instructions to Client machines that, in turn, relay the attack instructions down to the Daemon machines. The instructions contain information on what type of attack to launch and what address(s) to target. TFN is capable of launching a multitude of attacks including, UDP Flood, TCP Syn Flood, ICMP Echo Request Flood and ICMP Directed Broadcast. It is also capable of randomizing its source IP address, source port and packet size. All this, makes it extremely difficult to detect a TFN attack.

The makeup of TFN network would look something like the simple diagram below (Dittrich):



5. Attack Mechanism

The initial compromise of UNIX/Linux hosts that are to be used as TFN Daemon machines is not carried out by a component of the TFN program. Rather, these machines are first compromised by other methods, chiefly insecure default configurations of rpc.statd and wu-ftpd. Machines with insecure configurations are located through massive Internet scanning focused specifically at sunrpc ports 111 TCP/UDP and FTP port 21 TCP. More information about the rpc.statd and wu-ftpd vulnerabilities can be found at:

(http://www.cert.org/advisories/CA-2000-17.html) Input Validation Problem in rpc.statd
(http://www.cert.org/advisories/CA-2000-13.html) Two Input Validation Problems in FTPD

Once compromised, a rootkit is installed, which allows for the installation of the TFN program. Ensuing communications between the Clients and their assigned group of Daemons is carried out through ICMP Echo Reply packets with 16-bit binary values imbedded in the ICMP ID field, which in this case, is the code 51201. Due to differences in Code ID architectures, this actually equates to ICMP ID 456, which instructs the Daemon machine to spawn a shell. Other arguments may also be passed in the content portion of the packet.

"Remote control of the Daemon machines in the TFN is done through command line execution of the client program via a number of means, including (e.g., remote shell bound to a TCP port, UDP based client/server remote shells, ICMP based client/server shells such as LOKI, SSH terminal sessions, or normal "telnet" TCP terminal sessions." (Dittrich)

6. Correlations

The Tribal Flood Network scheme is well know and well documented. I ran the source address through IP Info at dshield.org and received no hits, as I expected. Until TFN unleashes a DDOS, I would not expect to see the source address listed on Dshield.org. Thus, this source IP probably has not taken part in a large scale DDOS that would've registered with Dshield.org.

The best source of information concerning TFN was produced in a lab work up by David Dittrich of the University of Washington. His paper entitled "The Tribe Flood Network Distributed Denial of Service Attack Tool" provides excellent insight into TFN. His work is available at the following URL:

http://staff.washington.edu/dittrich/misc/tfn.analysis

A few GCIA practicals have covered TFN in the past, namely the following:

http://www.giac.org/practical/Steve_Lukacs_GCIA.doc
http://www.giac.org/practical/Miika_Turkia_GCIA.html

CVE Reference: CAN-2000-0138.

http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0138

Further information regarding TFN can be found at the following:

http://www.cert.org/incident_notes/IN-2000-10.html
http://www.cert.org/incident_notes/IN-99-07.html
http://www.whitehats.com/info/IDS183
http://www.snort.org/snort-db/sid.html?sid=251

7. Evidence of Active Targeting

Active targeting of internal host my.net.42.91 is highly likely. It is probable this host was previously identified through a scan looking for machines with vulnerable Unix/Linux default configurations, especially default installs of rpc.statd and wu-ftdp. The fact that the source address was actively trying to communicate with my.net.42.91 via ICMP Echo Reply suggests the machine has already been compromised and the Client is attempting to issue commands to the Daemon machine.

8. Severity

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

Severity = (3 + 5) – (0 + 3)

Resulting Severity = 8 – 3 = 5

***Criticality* – 3.** The criticality of the system is unknown. Given TFN attacks Unix//Linux hosts, it is likely the machine is more important than a simple Windows-based workstation. Therefore, I gave Criticality a three.

***Lethality* – 5.** TFN is not particularly lethal to the Daemon machine that executes DDOS attacks, but TFN requires root level access to install. A TFN compromised machine could run forever and never be adversely affected by the TFN program. However, a machine hosting the TFN code may experience it's own DOS when the TFN program launches a DDOS attack. This depends on the machine's capability to keep up with conducting normal business and executing the DDOS at the same time. The lethality of TFN comes from the DDOS attacks produced by the program, not the damaged caused on the host machine itself. But, TFN requires root level access, which I will assert a five.

***System Countermeasures* – 0.** It appears the host my.net.42.91 probably has not been updated with the latest security patches for Unix/Linux systems and also may be running insecure default configurations. This is evident because the machine was apparently compromised prior to the loading of the TFN program. I rate System Countermeasures a zero.

***Network Countermeasures* – 3.** Our network utilizes basic packet filtering at each

gateway. Firewalls are also in place at the individual customer sites. In addition to a Snort sensor at each customer site, we also run an IDS product very similar to Shadow on the same platform (not recommended by the way). This product complements the Snort output. However, there are deficiencies in network security that I cannot detail in this document. I rate Network Countermeasures a three.

9. Defensive Recommendations

Tribal Flood Network is very difficult to defend against because once the victim machine is compromised communications are funneled through harmless looking ICMP Echo Reply traffic. When moving across the network, these packets will look like harmless PING traffic and will be very difficult to detect and weed out. Below is a list of recommendations that will contribute to successful detection and defense of TFN.

- Employ an IDS that observes the ICMP ID field. This will detect the communications between the Client and Daemon machines.

- Keep your machines from being compromised in the first place by ensuring your Unix/Linux machines are kept up to date on all operating system upgrades and patches. Also, administrators should ensure their Unix/Linux platforms are hardened to best industry standards, including the tightening of default configurations that pose security risks. Blocking ports 111 and 21 at the gateway would also get the job done, however some networks may need to allow port 21 through for legitimate FTP traffic.

- Block ICMP Echo Requests and Replies at the gateway.

10. Multiple Choice Test Question

Question. Tribal Flood Network communications are proven to be very difficult to detect. What characteristics of TFN make it so difficult to detect?

A. When TFN executes an attack, it is capable of randomizing its source address, ports and packet size.

B. TFN communications looks very similar to regular PING traffic.

C. TFN communicates via commands sent in the ICMP ID field of an ICMP Echo Reply packet.

D. All of the above.

Answer. D

Network Detect #3 - W32.BleBla Worm (aka Romeo and Juliet Worm)

1. Source of Trace

The trace below was taken from the same network as the trace in Network Detect #2. IP addresses and sensor names have been obfuscated in the same manner as Network Detect #2. E-mail addresses have also been sanitized to protect the privacy of the individual names seen in the detect. E-mail sanitization is represented by a lowercase "x".

```
Generated by ACID v0.9.6b22 on Mon October 06, 2003 04:57:02

-------------------------------------------------------------------------------

NIDxxx [2003-10-05 23:55:51] [snortDB/726]  Virus - Possible MyRomeo Worm
IPv4: 216.111.190.2 -> my.net.237.19
      hlen=5 TOS= dlen=1420 ID=3854 flags= offset= TTL=120 chksum=56198
TCP:  port=110 -> dport: 2390  flags=***AP*** seq=3033655149
      ack=1908226911 off=5 res= win=65471 urp= chksum=18045
Payload:  length = 1380

000 : 44 61 74 65 3A 20 20 20 20 20 53 75 6E 2C 20 20  Date:     Sun,
010 : 35 20 4F 63 74 20 32 30 30 33 20 31 39 3A 35 33  5 Oct 2003 19:53
020 : 3A 33 39 20 0D 0A 4D 65 73 73 61 67 65 2D 49 64  :39 ..Message-Id
030 : 3A 20 3C 31 30 33 31 30 30 35 31 39 35 33 2E xx  : <10310051953.x
040 : xx xx xx xx xx xx xx xx xx xx 40 xx xx xx xx xx  xxxxxxxxxx@xxx-x
050 : xx xx 2E 63 6F 6D 3E 0D 0A 4D 69 6D 65 2D 56 65  xx.com>..Mime-Ve
060 : 72 73 69 6F 6E 3A 20 31 2E 30 0D 0A 43 6F 6E 74  rsion: 1.0..Cont
070 : 65 6E 74 2D 54 79 70 65 3A 20 74 65 78 74 2F 70  ent-Type: text/p
080 : 6C 61 69 6E 3B 20 63 68 61 72 73 65 74 3D 75 73  lain; charset=us
090 : 2D 61 73 63 69 69 0D 0A 46 72 6F 6D 3A 20 20 20  -ascii..From:
0a0 : 20 20 22 50 6F 73 74 6D 61 73 74 65 72 22 20 3C    "Postmaster" <
0b0 : 70 6F 73 74 6D 61 73 74 65 72 40 xx xx xx 2D 6E  postmaster@xxx-n
0c0 : 65 74 2E 63 6F 6D 3E 0D 0A 53 65 6E 64 65 72 3A  et.com>..Sender:
0d0 : 20 20 20 3C 70 6F 73 74 6D 61 73 74 65 72 40 xx     <postmaster@x
0e0 : xx xx 2D 6E 65 74 2E 63 6F 6D 3E 0D 0A 54 6F 3A  xx-net.com>..To:
0f0 : 20 20 20 20 20 20 20 3C xx xx xx xx xx xx xx 40         <xxxxxxx@
100 : xx xx xx 2D 6E 65 74 2E 63 6F 6D 3E 0D 0A 53 75  xxx-net.com>..Su
110 : 62 6A 65 63 74 3A 20 20 55 6E 64 65 6C 69 76 65  bject:  Undelive
120 : 72 61 62 6C 65 20 4D 61 69 6C 0D 0A 58 2D 4D 61  rable Mail..X-Ma
130 : 69 6C 65 72 3A 20 3C 53 4D 54 50 33 32 20 76 38  iler: <SMTP32 v8
140 : 2E 30 30 3E 0D 0A 53 74 61 74 75 73 3A 20 55 0D  .00>..Status: U.
150 : 0A 58 2D 55 49 44 4C 3A 20 33 35 31 34 32 31 39  .X-UIDL: 3514219
160 : 33 31 0D 0A 0D 0A 55 6E 6B 6E 6F 77 6E 20 68 6F  31....Unknown ho
170 : 73 74 3A 20 xx xx xx xx xx xx 40 xx xx xx xx xx  st: xxxxxx@xxxxx
180 : xx xx xx xx 0D 0A 0D 0A 0D 0A 4F 72 69 67 69 6E  xxxx......Origin
190 : 61 6C 20 6D 65 73 73 61 67 65 20 66 6F 6C 6C 6F  al message follo
1a0 : 77 73 2E 0D 0A 0D 0A 44 61 74 65 3A 20 53 75 6E  ws.....Date: Sun
1b0 : 2C 20 20 35 20 4F 63 74 20 32 30 30 33 20 31 39  ,  5 Oct 2003 19
1c0 : 3A 35 33 3A 33 34 20 2D 30 34 30 30 0D 0A 4D 65  :53:34 -0400..Me
1d0 : 73 73 61 67 65 2D 49 64 3A 20 3C 32 30 30 33 31  ssage-Id: <20031
1e0 : xx xx xx xx xx xx xx 2E xx xx xx xx xx xx xx xx  xxxxxxx.xxxxxxxx
1f0 : xx xx 40 xx xx xx 2D 6E 65 74 2E 63 6F 6D 3E 0D  xx@xxx-net.com>.
200 : 0A 4D 69 6D 65 2D 56 65 72 73 69 6F 6E 3A 20 31  .Mime-Version: 1
```

```
210 : 2E 30 0D 0A 43 6F 6E 74 65 6E 74 2D 54 79 70 65  .0..Content-Type
220 : 3A 20 74 65 78 74 2F 70 6C 61 69 6E 3B 20 63 68  : text/plain; ch
230 : 61 72 73 65 74 3D 75 73 2D 61 73 63 69 69 0D 0A  arset=us-ascii..
240 : 46 72 6F 6D 3A 20 xx xx xx xx xx xx xx 20 xx xx  From: "xxxxxx xx
250 : xx xx xx 22 20 3C xx xx xx xx xx xx xx 40 xx xx  xxx" <xxxxxxx@xx
260 : xx 2D 6E 65 74 2E 63 6F 6D 3E 0D 0A 52 65 70 6C  x-net.com>..Repl
270 : 79 2D 54 6F 3A 20 3C xx xx xx xx xx xx xx 40 xx  y-To: <xxxxxxx@x
280 : xx xx 2D 6E 65 74 2E 63 6F 6D 3E 0D 0A 54 6F 3A  xx-net.com>..To:
290 : 20 22 xx xx xx xx 5F xx xx xx xx xx xx xx xx xx   "xxxx_xxxxxxxxx
2a0 : 22 20 3C xx xx xx xx xx xx 40 xx xx xx xx xx xx  " <xxxxxx@xxxxxx
2b0 : xx xx xx 3E 0D 0A 53 75 62 6A 65 63 74 3A 20 46  xxx>..Subject: F
2c0 : 57 44 3A 20 48 6F 77 20 74 6F 20 73 61 79 20 27  WD: How to say '
2d0 : 49 20 6C 6F 76 65 20 79 6F 75 27 20 69 6E 20 32  I love you' in 2
2e0 : 35 20 6C 61 6E 67 75 61 67 65 73 0D 0A 58 2D 4D  5 languages..X-M
2f0 : 61 69 6C 65 72 3A 20 3C 49 4D 61 69 6C 20 76 38  ailer: <IMail v8
300 : 2E 30 30 3E 0D 0A 0D 0A 2D 2D 2D 2D 2D 2D 2D 2D  .00>....--------
310 : 2D 2D 20 4F 72 69 67 69 6E 61 6C 20 4D 65 73 73  -- Original Mess
320 : 61 67 65 20 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D  age ------------
330 : 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D  ----------------
340 : 2D 2D 2D 2D 2D 2D 0D 0A 46 72 6F 6D 3A 20 22 xx  ------..From: "x
350 : xx xx xx 20 xx xx xx xx xx 22 20 3C xx xx xx xx  xxx xxxxx" <xxxx
360 : xx xx 40 xx xx xx xx xx xx 2E 6E 65 74 3E 0D 0A  xx@xxxxxx.net>..
370 : 52 65 70 6C 79 2D 54 6F 3A 20 3C xx xx xx xx xx  Reply-To: <xxxxx
380 : xx 40 xx xx xx xx xx xx 2E 6E 65 74 3E 0D 0A 44  x@xxxxxx.net>..D
390 : 61 74 65 3A 20 20 53 61 74 2C 20 20 34 20 4F 63  ate:  Sat,  4 Oc
3a0 : 74 20 32 30 30 33 20 32 31 3A 35 33 3A 32 36 20  t 2003 21:53:26
3b0 : 2D 30 34 30 30 0D 0A 0D 0A 48 6F 77 20 74 6F 20  -0400....How to
3c0 : 73 61 79 20 27 49 20 6C 6F 76 65 20 79 6F 75 27  say 'I love you'
3d0 : 20 69 6E 20 32 35 20 6C 61 6E 67 75 61 67 65 73   in 25 languages
3e0 : 2E 2E 2E 52 65 61 64 20 64 6F 77 6E 2E 2E 2E 0D  ...Read down....
3f0 : 0A 0D 0A 45 6E 67 6C 69 73 68 20 0D 0A 49 20 4C  ...English ..I L
400 : 6F 76 65 20 59 6F 75 20 0D 0A 0D 0A 53 70 61 6E  ove You ....Span
410 : 69 73 68 20 0D 0A 54 65 20 41 6D 6F 20 0D 0A 0D  ish ..Te Amo ...
420 : 0A 46 72 65 6E 63 68 20 0D 0A 4A 65 20 54 27 61  .French ..Je T'a
430 : 69 6D 65 20 0D 0A 0D 0A 47 65 72 6D 61 6E 0D 0A  ime ....German..
440 : 6C 63 68 20 4C 69 65 62 65 20 44 69 63 68 20 0D  lch Liebe Dich .
450 : 0A 0D 0A 4A 61 70 61 6E 65 73 65 20 0D 0A 41 69  ...Japanese ..Ai
460 : 20 53 68 69 74 65 20 49 6D 61 73 75 20 0D 0A 0D   Shite Imasu ...
470 : 0A 49 74 61 6C 69 61 6E 20 0D 0A 54 69 20 41 6D  .Italian ..Ti Am
480 : 6F 20 0D 0A 0D 0A 43 68 69 6E 65 73 65 20 0D 0A  o ....Chinese ..
490 : 57 6F 20 41 69 20 4E 69 20 0D 0A 0D 0A 53 77 65  Wo Ai Ni ....Swe
4a0 : 64 69 73 68 20 0D 0A 4A 61 67 20 41 6C 73 6B 61  dish ..Jag Alska
4b0 : 72 20 0D 0A 0D 0A 41 6C 61 62 61 6D 61 0D 0A 41  r ....Alabama..A
4c0 : 72 6B 61 6E 73 61 73 0D 0A 4B 61 6E 73 61 73 0D  rkansas..Kansas.
4d0 : 0A 4F 6B 6C 61 68 6F 6D 61 0D 0A 54 65 78 61 73  .Oklahoma..Texas
4e0 : 0D 0A 4E 6F 72 74 68 20 43 61 72 6F 6C 69 6E 61  ..North Carolina
4f0 : 0D 0A 53 6F 75 74 68 20 43 61 72 6F 6C 69 6E 61  ..South Carolina
500 : 0D 0A 47 65 6F 72 67 69 61 0D 0A 54 65 6E 6E 65  ..Georgia..Tenne
510 : 73 73 65 65 0D 0A 49 64 61 68 6F 0D 0A 4D 69 73  ssee..Idaho..Mis
520 : 73 6F 75 72 69 0D 0A 4D 69 73 73 69 73 73 69 70  souri..Mississip
530 : 70 69 0D 0A 4D 6F 6E 74 61 6E 61 0D 0A 4C 6F 75  pi..Montana..Lou
540 : 69 73 69 61 6E 61 0D 0A 56 69 72 67 69 6E 69 61  isiana..Virginia
550 : 0D 0A 57 65 73 74 20 56 69 72 67 69 6E 69 61 0D  ..West Virginia.
560 : 0A 4B 65 6E                                      .Ken
```

Response: none

2. Detect Was Generated By

This detect was generated by a Snort sensor configured identical to the sensor in Network Detect #2, running on an identical Solaris platform. But, the sensor is located at a separate physical location and on a different network segment.

The following Snort signature generated the alert:

```
alert tcp any 110 -> any any (msg:"Virus - Possible MyRomeo Worm"; content:
"I Love You"; sid:726; classtype:misc-activity; rev:4;)
```

This Snort signature fires when it detects a packet with the following rule header and rule option characteristics:

Rule Header

- *Protocol* – TCP
- *Source IP Address* – any source address
- *Destination IP Address* – any destination address
- *Source Port* – source port must by 110 (POP3)
- *Destination Port* – any destination port
- *Direction of Traffic Flow* – From the source address to the destination address, represented by "->"

Rule Options

- Content string match for "I Love You"

3. Probability The Source Address Was Spoofed

It is very unlikely the source address was spoofed, as the alert is a false positive. Thus, there is no reason to believe the source address is not the authentic sender of the e-mail. Analysis of the payload indicates this is a legitimate e-mail that has been forwarded, probably amongst friends or coworkers. This e-mail is not propagating the W32.BleBla worm. The alert generated by Snort keyed in on the "I Love You" string in the Subject Line of the e-mail. Aside from the "I Love You" string in the packet, it contains no other characteristics of the W32.BleBla worm.

4. Description of Attack

I've determined that the packet trace represents a false positive, thus no attack took place. The data that proves this to be a false positive is contained within the packet payload. In the payload, we see the string "*I Love You*", which matches the "*I Love You*" string in the Snort signature. But, my research indicates that the string used by the

BleBlah virus is actually "*I Love You ;)*". I'm not sure why the rule isn't more precise, as the addition of the ";)" would serve to eliminate a lot of false positives. Analysis of the payload also indicates this e-mail is simply a forwarded message between coworkers or friends. Further, if the machine were actually infected, we probably would have seen the following Snort alert fire, indicating the machine had been infected.

```
alert tcp any 110 -> any any (msg:"Virus - Possible MyRomeo Worm"; content:
"myjuliet.chm"; nocase; sid:724; classtype:misc-activity; rev:4;)
```

The above alert did not fire because the packet did not contain the MYJULIET.CHM file, which is contained in every email propagated by the virus.

5. Attack Mechanism

The W32.BleBla worm (aka Romeo and Juliet worm), spreads via mass mailing. This worm is thought to have originated in Poland in November 2000. The virus is transported via an HTML e-mail, which contains the malicious code in the form of two e-mail attachments: MYJULIET.CHM and MYROMEO.EXE. A BleBla infected e-mail arrives with one of the following Subject Lines:

Romeo&Juliet
:))))))
hello world
!!??!?!?
subject
ble bla, bee
I Love You ;)
sorry...
Hey you !
Matrix has you..
my picture
from shake-beer

When a user opens an infected e-mail, the HTML portion is automatically executed. The HTML portion contains a script that is automatically activated by Windows 95, 98 and Windows 2000 systems running old version of Internet Explorer. The HTML is able to automatically execute because of an "Iframe vulnerability" in Internet Explorer versions 4 & 5, and a "Cache Bypass" vulnerability in Outlook 2000 and below and Outlook Express versions 4 & 5. The patches for these vulnerabilities are available at:

http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms99-042.asp.

http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS00-046.asp.

The activation of the script mentioned above causes the MYJULIET.CHM file and MYROMEO.EXE file to save down to the Windows TEMP directory on the host machine

without any notice to the user. The CHM file then executes from the TEMP folder, where the virus takes advantage of another Microsoft vulnerability. This time it's the "scriptlet.typelib/Eyedog" vulnerability, which allows the few lines of "Help Code" in the CHM file to kick off the MYROMEO.EXE file. The CHM file is actually a compressed HTML page that contains Microsoft HTML help files. The "scriptlet.typelib/Eyedog" vulnerability and HTML Help File Code Execution Vulnerability affect IE versions 4 & 5, and patches are available at:

http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms99-032.asp.

http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms00-037.asp.

In total, four vulnerabilities must be compromised in order for the W32.BleBla virus to run. When the virus runs, it takes addresses from the Windows Address Book file, and sends the same HTML message and CHM/EXE files to all addresses in the user's address book.

The W32.BleBla virus does very little known damage, if any at all. It appears that it's nothing more than code that takes advantage of several Microsoft vulnerabilities, only to generate a bunch of e-mail. In that regard, it's nothing more than Spam. The biggest problem with the virus is the installation of unwanted code on your machine that serves no useful purpose.

6. Correlations

Web searches for GCIA practicals with My Romeo (W32.BleBla) detects came up negative. Further information on the W32.BleBla (Romeo and Juliet) virus can be found at the following websites:

Network Associates: http://vil.nai.com/vil/content/v_98894.htm

Symantec:
http://securityresponse.symantec.com/avcenter/venc/data/w32.blebla.worm.html

7. Evidence of Active Targeting

Destination address my.net.237.19 was not actively targeted with the W32.BleBla worm as the Snort alert is a false positive. An inspection of the payload indicates this is simply an e-mail probably sent as fun reading between friends or coworkers. It just so happened that the "I Love You" string was located in the Subject Line of the e-mail, which caused the Snort rule to fire.

8. Severity

***Note: This section was completed as if this detect was not a false positive.*

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

Severity = (3 + 1) – (1 + 3)

Resulting Severity = 4 – 4 = 0.

Criticality **– *3.*** The criticality of the system is completely unknown. Therefore, I gave Criticality a three.

Lethality **– *1.*** The W32.BleBla virus doesn't do any major harm to the machine that it infects. The biggest problems produced by the virus are the sending of unsolicited emails and the unknowing execution of relatively harmless code on your machine. I'll give Lethality a one.

System Countermeasures **– *1*.** If the machine actually became infected with W32.BleBla, I'd guess the machine was an old Windows machine running outdated versions of Outlook and IE, which are obviously not patched against known security vulnerabilities. This machine is probably not administered on a regular basis by qualified system administrator. Since this virus is rather harmless, it's not quite as critical to have the patches so I'll give System Countermeasures a one instead of a zero.

Network Countermeasures ***3.*** Our network utilizes basic packet filtering at each gateway. Firewalls are also in place at the individual customer sites. In addition to a Snort sensor at each customer site, we also run an IDS product very similar to Shadow on the same platform (not recommended by the way). This product complements the Snort output. However, there are deficiencies in network security that I cannot detail. I rate Network Countermeasures a three.

9. Defensive Recommendation

No defensive measures are necessary since this network detect was a false positive. However, if you wanted to defend against W32.BleBla, you could implement the following recommendations:

- Employ content filtering software at your e-mail gateway to filter out e-mails containing HTML scripts, .chm attachments and .exe attachments.
- Keep all your systems updated with current operating system and application security patches.
- Deploy antivirus software with current DAT files on all Windows-based machines.
- Employ an IDS that looks at a different Subject Line string than "I Love You". The "I Love You" string is too common and will often falsely fire. It would be better to use the content strings "ble bla, bee" or "from shake-beer" as they are less likely to show up in legitimate traffic.

10. Multiple Choice Test Question

Question. Four defensive recommendations were discussed regarding the W32.BleBla virus. Which recommendation would provide the best capability to keep the virus out of your corporate e-mail system?

A. Employ content filtering software at your e-mail gateway to filter out e-mails containing HTML scripts, .chm attachments and .exe attachments.

B. Keep all your systems updated with current operating system and application security patches.

C. Deploy antivirus software with current DAT files to all Windows-based machines.

D. Employ an IDS that inspects for a different Subject Line string other than "I Love You". The "I Love You" string is too common and will often fire. It would be better to use the content strings "ble bla, bee" or "from shake-beer" as they are less likely to show up in legitimate traffic.

Answer. A

Assignment 3: Analyze This

Executive Summary

The Faske Security Analysis Firm has recently completed a technical security audit of the University's MY.NET/16 network. The purpose of this audit was to determine the security posture of the University's network and provide applicable defensive guidance where warranted. To conduct this analysis, I was granted access to five consecutive days of the University's Snort IDS logs. These logs composed a total of 7,273,182 alerts. The high number of total alerts is indicative of a badly tuned intrusion detection sensor.

I used alert frequency as the primary factor for determining which logs would receive the most attention. By using this methodology, I effectively analyzed 99.3% of all University provided alerts. The majority of my analysis concentrated on the alert logs as the scan logs are primarily a precursor to activity, and do not lend much insight to a five-day review. If long-term logs were available, then the scan logs would become more useful in establishing trending or "low and slow" scanning activity. Also, large scans tend to be used by "Script Kiddies" looking for an easy avenue of exploitation. While it is important to watch over this crowd, it should not consume the majority of our analysis time.

The University's security policies were not provided, thus I could not determine authorized activity from unauthorized activity. Given the narrow focus of the Snort log information, the University should use the results from this report in conjunction with other security analysis data. In reality, much more information is necessary to provide a

full analysis of the University's security posture.

My review of the Snort IDS logs indicates the University does in fact have several security issues. These issues are highlighted in the following bullets:

- Ingress/egress filtering is inadequate at the gateway device(s). The University should consider implementing a "deny by default" gateway policy.
- All systems are not running current antivirus solutions and system patches. There are possible Opaserv or Bugbear infections on MY.NET.168.118 and MY.NET.150.133. The NIMDA virus may also be present on MY.NET.107.98.
- The possibility of insecure access to Novell Servers from Internet addresses.
- Eight University systems may be harboring Trojans, such as SubSeven.
- The use of VNC to remotely control University systems from external addresses.
- University systems are confirmed to be using a variety of Peer-to-Peer utilities.

List of Files Analyzed

The data analyzed for the security audit was generated by the University's Snort intrusion detection system. The data consisted of five days of alert files, scan files and Out of Spec (OOS) files. All files were concatenated to allow for improved aggregate analysis over the time period.

Alert Files	Scan Files	Out Of Spec (OOS) Files
alert031016.gz	scans031016.gz	*OOS_Report_2003_10_08_9573.txt
alert031017.gz	scans031017.gz	*OOS_Report_2003_10_09_15060.txt
alert031018.gz	scans031018.gz	*OOS_Report_2003_10_10_30875.txt
alert031019.gz	scans031019.gz	*OOS_Report_2003_10_11_14832.gz
alert031020.gz	scans031020.gz	*OOS_Report_2003_10_12_9023.gz

Table 1. List of Files Analyzed

**Note: The dates on the OOS Files do not correspond with the Alert and Scan files because no matching OOS Files were available on the day of download that would meet the imposed 60-day window. Thus, I selected the five OOS files closest in date to the Alert/Scan files. SANS officials approved the use of these files via e-mail.*

Summary of Alerts

Just over 400,000 alerts were analyzed for the five-day period between Thursday, 16 October 2003 and Monday, 20 October 2003. This number does not include Scan alerts and OOS alerts. Actually many more alerts were generated by the IDS, however corrupted log entries and alerts generated by the Snort portscan preprocessor, were removed via a Perl script (See Appendix A). The resulting alerts produced a fairly clear picture of the University's security posture.

Date	Number of Alerts
October 16, 2003 (Thursday)	179,867 Alerts
October 17, 2003 (Friday)	149,607 Alerts
October 18, 2003 (Saturday)	27,464 Alerts
October 19, 2003 (Sunday)	26,773 Alerts
October 20, 2003 (Monday)	16,535 Alerts
Total	400,617 Alerts

Table 2. Alert Count By Date

List of Frequent Alerts

Roughly 98% percent of all alerts were generated by only eight signatures. Each of these signatures fired at least one thousand times. Of these eight signatures, the "SMB Name Wildcard" signature alone generated 91% percent of all events. The breakdown of the top eight signatures is located in Table 1 below, followed by a thorough analysis of each of the top eight alerts. Each of the top eight alerts fired more than 1,000 times, thus were considered "high frequency" alerts.

Line Item	Alert Name	# of Alerts	% of All Alerts	Severity
1	SMB Name Wildcard	366437	91%	Medium
2	TCP SRC and DST outside network	8862	2.2%	Low
3	MY.NET.30.4 activity	7086	1.7%	Medium
4	MY.NET.30.3 activity	4953	1.2%	Medium
5	EXPLOIT x86 NOOP	3806	.95%	Noise
6	Possible trojan server activity	3080	.77%	High

Line Item	Alert Name	# of Alerts	% of All Alerts	Severity
7	ICMP SRC and DST outside network	1171	.29%	Low
8	Connect to 515 from inside	1001	.25%	Low
	Totals	396,396	98%	

Table 3. High Frequency Alerts

Analysis of Frequent Alerts

1. SMB Name Wildcard

Synopsis. The *"SMB Name Wildcard"* alert indicates Windows NETBIOS name resolution is occurring. In this case, every *"SMB Name Wildcard"* event originated from an internal source address. The two top talkers were MY.NET.168.118 and MY.NET.150.133, and they both targeted external Internet addresses. My efforts to find a current Snort rule for this event failed. However, an Internet search turned up the following Snort rule on a posting to the Insecure.org mailing list:

alert udp any any -> $HOME_NET 137 (msg:"SMB Name Wildcard";
content:"CKAAAAAAAAAAAAAAAAAAAAAAAAAAAA|0000|";)

This rule would not have caused alerts to fire in this case, because all of the activity I analyzed was outbound from "Homenet", whereas the rule above fires on SMB activity inbound to "Homenet". But, it's the closest rule I could find. I can only surmise that this rule was available in previous distributions of Snort and is no longer available. This leads me to believe the University has written a custom rule to track outbound Windows NETBIOS activity. I am perplexed by the fact that there were no alerts generated by normal internal NETBIOS traffic, (i.e. MY.NET/16 -> MY.NET/16). The University must have set the rule to fire on outbound port 137 packets only.

All the events generated by MY.NET.168.118 and MY.NET.150.133 originated on source ports greater that 1024, indicating the possibility of viral infection or Trojan activity on a Wintel platform. We might be dealing with Bugbear or Opaserv infections. This is further supported by the randomness of the destination IP addresses. If the alerts would have identified the protocol (TCP or UDP), it would have narrowed down the possibilities, as regular NETBIOS occurs on port 137 UDP.

A significant drop in SMB Name Wildcard alerts explains the massive reduction in total alerts after October 17th. Specifically, MY.NET.168.118 is no longer seen triggering alerts from October 18th – October 20th. This could indicate the machine was identified and taken offline to be repaired.

Correlation. Queries to the Internet Storm Center indicate port 137 is a top ten scanned port on the Internet and is also commonly utilized by several viruses or trojans. The following ISC webpage also lists the four applicable CVE references for this type of port 137 activity http://isc.incidents.org/port_details.html?port=137.

Tod Beardsley also covers the "SMB Name Wildcard" event in his practical. This was also his most frequent event, however there is a major difference in the events Mr. Beardsley analyzed. All of his alerts except for two were generated by normal NETBIOS activity. This was indicated by source and destination addresses both in the MY.NET.0.0 address range. Thus, the correlation to his results ends there, because my alerts were from internal addresses to external addresses. Correlation to scan alerts was not necessary since all "SMB Name Wildcard alerts were from internal addresses, whereas all scan alerts were from external addresses.

Defensive Recommendations. My first thought is to block port 137 UDP/TCP outbound at the firewall or packet filtering router. There's no need for this traffic to route to the Internet. But, this would serve only to protect the external addresses targeted by the likely infected internal machines. Port 137 inbound already seems to be blocked, as there is no indication of inbound port 137 activity. Keeping all Windows-based machines up-to-date on patches and current antivirus solutions also comes to mind. Both Opaserv and Bugbear have been around for a while, and should have been patched well over a year ago.

2. TCP SRC and DST Outside Network

Synopsis. This alert appears to be a custom written rule to detect traffic with source and destination addresses outside the home network. This rule may be looking for spoofed traffic originating from MY.NET.0.0 or misconfigured internal traffic. Ninety-eight percent of these alerts were generated by source address 169.254.244.56, which resolves to an IANA Special Use address. Another forty-five alerts were caused by eight different IANA source addresses in the 192.168.0.0 range. There were another seven source addresses that triggered alerts; these resolved to AOL, France Telecom, a South African Telecom, and an ISP in Brazil.

The really interesting facts about all this traffic were the primary destination addresses and the related destination ports. Ninety-eight percent of all traffic was destined for three IP addresses registered to Chinese entities, and every source address but two, addressed traffic to the Chinese entities. Additionally, all this traffic had a destination port of 21 or 996. The well known port 21 resolves to FTP, and after a quick search, I found that port 996 indicates Central Point Software Xtree License Server. Further research of Central Point, indicates they were in the antivirus software business before being bought out by Symantec. Not much information was available, but I think Central Point utilized port 996 to distribute their antivirus updates. I'm not sure of its current use.

Correlation. Given these alerts were created by a custom rule that doesn't relate to Snort rules or other IDS rules, I found no correlations on the Internet, except for

those in other GCIA practicals. Aaron Hackworth evaluated this alert and concluded the alerts were the product of "3rd party noise from a spoofed scanning attempt, a routing error or a configuration error" (Hackworth). Sam Adams also took a look at this alert and concluded that his alerts were likely the product of a scan in reverse or someone was trying to use internal machines to launch a DOS attack.

Defensive Recommendations. The outbound port 21 traffic is concerning and should be blocked at a gateway device, unless it is necessary for port 21 to traverse the gateway. Which in the case of a University, it is highly likely that port 21 will need to remain open due to FTP servers serving up content. But, a hole could be punched just for the necessary FTP servers and everything else blocked. The University should also try to track down the actual machines that are using the IANA addresses through their MAC addresses, and then investigate the nature of what is occurring. The IANA reserved packets will be routed to the destination addresses, but a TCP connection would not be possible because the IANA reserved and special use addresses are non-routable on the Internet.

3. MY.NET.30.4 Activity

Synopsis. This alert appears to be the product of a custom rule to track all external activity going to University address MY.NET.30.4. Analysis of traffic pertaining to MY.NET.30.4 indicates this machine is a Novell Enterprise Webserver. The only information available to deduce this was the primary destination ports used by systems communicating with MY.NET.30.4. The primary destination ports seen were port 80 (HTTP), port 524 (Netware Core Protocol-NCP) and 51443 (HTTPS-Netware). These three ports accounted for 99% of all activity. No machines in MY.NET.0.0 were seen communicating with the webserver. Analysis of external addresses communicating with MY.NET.30.4 did not show any specific patterns. I resolved five different source addresses and they all belonged to major ISPs.

The activity to port 524 (NCP) from external addresses is a bit troubling. The Netware Core Protocol is used to access Novell Netware file and print services and this protocol has been reported by Bindview Razor to have a problem with leaking information. Razor reports that, "All Novell Netware servers running IP (with port 524 open) can be queried and all objects with Public read access can be enumerated. Information such as account names, server services, and other various objects can be gathered" (Razor). There were 456 external connections to this port, all from 12 machines in the 66.218.172.0 Class C. This Class C belongs to Yahoo. One possibility is that a 3rd party is being used to administer the webserver, and the 3rd party uses this address range. However, using 12 addresses to do so seems strange, especially coming from Yahoo address space. The addresses could certainly be DHCP'd, but this would require giving too many addresses access to the webserver. A better solution would be to select static addresses and create a one-to-one authorized connection between two machines.

Correlation. I reviewed twenty GCIA practicals looking for someone who evaluated this alert and found none that did so. Plenty of students had this alert in their "Alert"

charts, but did not chose to further explore the alert.

Given this is a custom written alert, no hits were generated on web searches. The only correlation I was able to accomplish was with respect to the Novell Webserver itself, which was covered in the Synopsis section above.

Defensive Recommendations. Close down port 524 to external addresses. The availability of this port to external addresses represents a risk that can be easily addressed. The possibility of gaining information that can be used to launch further attacks is too great not to fix this hole.

4. MY.NET.30.3 Activity

Synopsis. This alert also appears to trigger on external access to a University server. However, MY.NET.30.3 does not appear to be a University webserver as MY.NET.30.4 was. Rather, it seems to serve as a regular Novell server. Analysis of the traffic indicated the alerts were generated by 72 distinct source addresses, all external to MY.NET.0.0. Ninety-nine percent of the alerts were destined for port 524 (NCP). Only 14 alerts were destined for port 80 (HTTP) five alerts to port 21 (FTP) and 1 alert to port 25 (SMTP). The remaining 25 alerts were destined for port numbers greater than 1024.

The question I ask myself is, "Why is the University allowing external access to one of their Novell servers. I can only surmise that they are allowing students to connect to this machine from the Internet for legitimate reasons, which is a very insecure practice. Another possibility is the University has placed their Snort instance outside their firewall, and we are seeing connection attempts only. To test this possibility, I queried the six top talking source addresses from the MY.NET.30.3 alerts against the MY.NET.30.4 alerts. I found that three of these addresses were in fact communicating with both servers. These addresses are:

216.83.163.132	(Sungard Network Solutions)
68.48.217.68	(Comcast Cable)
68.55.105.5	(Comcast Cable)

The fact that multiple machines from both alert sets are connecting to both servers indicates the connections are likely legitimate and not just connection attempts from hostile machines. Without more information it is very difficult to understand what is actually transpiring. Also, given the wide range of source addresses involved, I'm not exactly sure how University officials are able to determine legitimate users from non-legitimate users.

Correlation. Same as the correlations for the MY.NET.30.4 alert.

Defensive Recommendations. Further information is necessary to determine the true extent of the port 524 connections or connection attempts. If this server is actually serving content to external users, then the University is taking a huge security risk. A better option would be to use secure web services to serve the data

or they could also use VPN software to establish a secure tunnel between the clients and Novell server. The VPN solution may not be cost effective for a University, but it is still a legitimate security solution.

5. EXPLOIT x86 NOOP

Synopsis. An x86 NOOP attack occurs when an attacker attempts to hide a buffer overflow attack within NOP instructions for machines running Intel x86 chip designs. Each x86 attack is dependant upon the specific platform being attacked. Buffer overflow exploits are sent via NOP instructions to "pad" their chances of success. The Snort alerts below are the closest I could find in name to the actual "Exploit x86 NOOP" alert. I'm assuming that one of these Snort rules caused the alerts to fire. With this alert, Snort is keying in on the series of 0x90 and 0x61 bytes seen in the packet contents.

alert ip $EXTERNAL_NET $SHELLCODE_PORTS -> $HOME_NET any (msg:"SHELLCODE x86 NOOP"; content: "|90 90 90 90 90 90 90 90 90 90 90 90 90 90|"; depth: 128; reference:arachnids,181; classtype:shellcode-detect; sid:648; rev:6;)

alert ip $EXTERNAL_NET $SHELLCODE_PORTS -> $HOME_NET any (msg:"SHELLCODE x86 NOOP"; content:"|61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61|"; classtype:shellcode-detect; sid:1394; rev:4;)

Analysis of the x86 NOOP alerts indicates that 75% percent of the alerts were generated by source addresses 202.31.152.174 (Kwangju National University of Education-Korea) and 62.21.236.14 (Publishnet-The Netherlands). Every alert was generated by traffic coming from external sources. The top five University destination addresses are listed below:

MY.NET.5.95	(375 Alerts)
MY.NET.111.72	(213 Alerts)
MY.NET.150.101	(208 Alerts)
MY.NET.5.92	(200 Alerts)
MY.NET.5.15	(196 Alerts)

Analysis of the destination ports indicates that eighty-seven percent of all alerts had a destination port of 80 (HTTP). Port 80 is known to cause a high rate of false positives for this alert, so these alerts should be viewed with this in mind. Destination port 135 (epmap-WINS Manager) was seen in 215 alerts, representing roughly 6 percent of all alerts. Hopefully, the Snort instance is located outside the University firewall and we are only seeing a connection attempt, and not a full TCP connection, because port 135 should not be available to external users.

Correlation. Surprisingly, I was unable to correlate this alert with the Carnegie Mellon Software Engineering Institute CERT® Coordination Center or Common Vulnerabilities and Exposures database. I was able to find some information in two GCIA practicals by David Oborn and Chris Kuethe. But, both practicals concentrated on alerts that were deemed false positives by the analysts. One of the

false positives was caused by Photoshop and the other by ICQ sound scheme files. Other than the practicals listed above, my sources of information were Snort.org and Whitehats.com.

Defensive Recommendations. The best defense against NOP attacks is to keep all your hosts patched against buffer overflow vulnerabilities. The "EXPLOIT x86 NOOP" rule is known to cause a high amount of false positives generated by transfers of large files, especially GIF and JPEG files. Thus, I would consider turning this rule off given the great difficulty in finding an actual attack amongst all the false positives.

6. Possible Trojan Server Activity

Synopsis. This is a custom rule that attempts to identify Trojans via source or destination port 27374. Port 27374 is used by several versions of the SubSeven Trojan, the Bad Blood Trojan and the Ramen worm. Blackcode.com indicates that Bad Blood is capable of communicating with SubSeven, making identification of the root problem even tougher to determine (Blackcode.com).

Review of the data indicates there are eight university machines that are possibly infected with one of the trojans/worms mentioned previously. All of the machines below conducted bi-directional communications with external hosts. Machines that triggered alerts on well-known ports (e.g. 80, 443, 20) were not listed, as I believe this traffic is legitimate. The source port was ephemeral in these instances and just happened to be 27374. The University machines that need to be investigated are:

MY.NET.6.15	MY.NET.112.152
MY.NET.190.1	MY.NET.190.202
MY.NET.84.14	MY.NET.190.203
MY.NET.190.101	MY.NET.190.97

Please see the link diagram on the next page for a graphical rendering of the possible trojan communications.

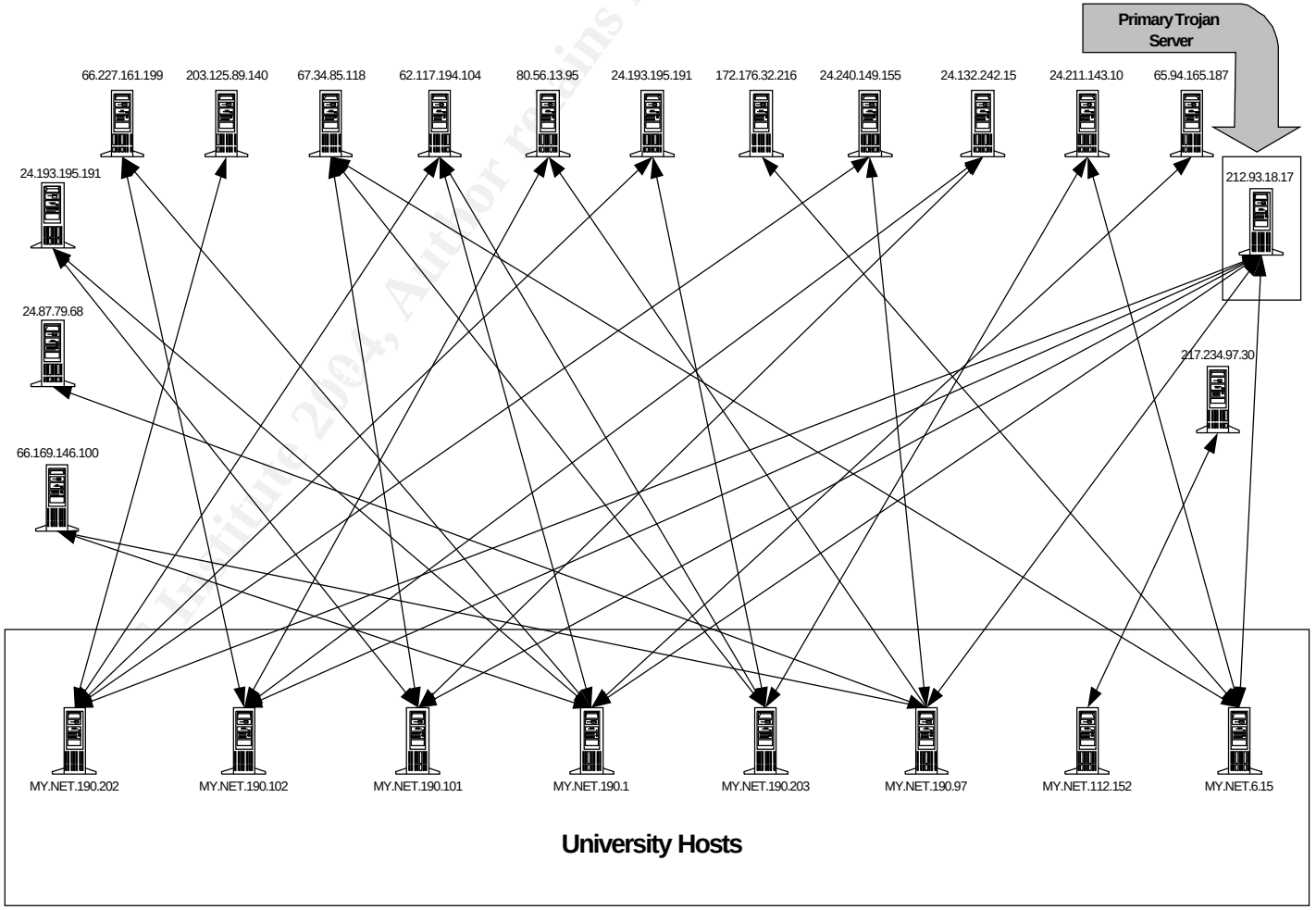


Figure 5. Possible Trojan Activity

Correlation. My results correlate nicely with those of Tod Beardsley's GCIA Practical. Mr. Beardsley also identified several University machines that were likely infected with the SubSeven Trojan or the Ramen worm. Mr. Beardsley's practical was written in May 2003, and my concern here, is that the University still has not taken steps to shut down port 27374 at their gateway points.

The remainder of my correlation came from the following websites:

http://www.blackcode.com/trojans/details.php?id=134
http://www.sans.org/resources/idfaq/oddports.php
http://www.cert.org/incident_notes/IN-2001-01.html

Defensive Recommendations. All of the University hosts sending traffic to port 27374 need to be investigated by the system administration staff as soon as possible. University administrators should also ensure all machines are currently patched and running current AV solutions.

Port 27374 should be blocked inbound and outbound at the University's gateway. This will keep trojans/worms using this port from propagating to University assets. It will also serve to contain the trojans/worms inside the University's border once the filter has been implemented.

7. ICMP SRC and DST Outside Network

Synopsis. This alert appears to be the product of ICMP traffic that contains both a source and destination address outside the University's network range. This type of activity could be indicative of internal address spoofing, crafted packets or misconfiguration. More investigation into the type of ICMP and related codes is necessary to fully determine the maliciousness of this traffic.

Analysis of the alerts shows there were 77 source addresses and 1,012 destination addresses. The small chart below provides some resolution information for the top 3 address owners. This chart indicates that an address from the entity was recorded either as the source or destination address.

IP Addresses	Whois Lookup	# of Alerts	% of Alerts
172.128.0.0-172.205.0.0	America Online	813	69%
68.42.0.0-68.60.0.0	Comcast	203	17%
192.168.0.0	IANA Reserved	153	13%
	Totals	1169	99%

Table 4. Top Three Address Ranges For "ICMP SRC and DST Outside Network" Alert

Correlation. Stephen Pederson covered this alert very briefly in his GCIA Practical. Mr. Pederson concluded this activity could be part of a DOS attack, but he concluded the volume of packets were insufficient to be successful. I agree with his point that ICMP is very amenable to denial of service attacks, and the small amount of alerts I analyzed further support the fact that there were not enough packets to cause a denial situation. Especially given the high number of source and destination addresses. In a denial of service situation, I would expect to see a many to one relationship instead of a many to many relationship, as was evidenced by the "ICMP SRC and DST Outside Network" alerts.

Defensive Recommendations. ICMP is certainly recognized as a major player in DOS attacks, given the ease of spoofing ICMP packets. The University should ensure they employ proper ingress/egress controls to limit exposure to hostile ICMP traffic. Specifically, the National Security Agency recommends the following router settings to protect against ICMP probing and attacks:

1) *Block ICMP Echo and redirect messages* - Echo messages allow an attacker to build a picture of the network structure and hosts, or flood selected hosts. Redirect messages could allow an attacker to make changes to the host's routing table.

2) *For outbound ICMP, allow only message types, Echo, Parameter Problem, Packet Too Big, and Source Quench* - Block all other ICMP message types heading outbound.

3) *Block inbound traceroute via ports 33400-34400 UDP* - This will keep attackers from mapping your network.

4) *Employ Committed Access Rate (CAR) on your gateway router* - CAR effectively limits bandwidth to defined traffic types, thus protecting from Denial of Service attacks.

8. Connect To 515 From Inside

Synopsis. This alert appears to be generated by a custom rule designed to match on outgoing packets to port 515. Port 515 is described by IANA as the "Spooler" port. Basically, this is the port used to conduct network printing. There are multiple vulnerabilities (See CVEs below) attributed to port 515, both in Windows and Unix platforms.

All of the alerts originate from six University addresses with a destination to one of three addresses registered to the National Aeronautics and Space Administration (NASA). The six University addresses belong to two different subnets: MY.NET.97.0 and MY.NET.162.0. The information above leads me to believe these alerts are not malicious in nature. Rather, I think the University and NASA are in a working partnership and this activity is likely authorized.

Correlation. Todd Chapman and John Hally both evaluated this alert in their respective GCIA practicals, however their alerts were completely internal to the University and they concluded the alerts were the product of normal network printing.

Even though I don't think this activity is hostile in nature, I've included a list of CVEs related to Port 515 vulnerabilities:

CVE-1999-0032	CVE-1999-0335	CVE-1999-1102
CVE-2001-0906	CVE-2000-1208	CVE-2003-0144
CVE-2000-0232	CVE-2000-0839	

Defensive Recommendations. Since I have concluded this is legitimate activity, I have no defensive recommendations. But, the University should establish pass rules for those machines that are legitimately conducting print operations.

Low Frequency Alerts

The alerts listed below fired less than one thousand times. These alerts accounted for roughly two percent of all alerts.

Line Item	Alert Name	# of Alerts
9	SUNRPC highport access!	755
10	Null scan!	620
11	NMAP TCP ping!	558
12	High port 65535 tcp - possible Red Worm - traffic	409
13	scan (Externally-based)	363

Line Item	Alert Name	# of Alerts
14	[UMBC NIDS IRC Alert] IRC user /kill detected	300
15	High port 65535 udp - possible Red Worm - traffic	251
16	[UMBC NIDS IRC Alert] Possible sdbot floodnet detected ...	240
17	[UMBC NIDS IRC Alert] K\:line'd user detected	102
18	SMB C access	86
19	[UMBC NIDS] External MiMail alert	76
20	Incomplete Packet Fragments Discarded	73
21	FTP passwd attempt	73
22	Tiny Fragments - Possible Hostile Activity	52
23	EXPLOIT x86 stealth noop	38
24	EXPLOIT x86 setuid 0	28
25	TFTP - Internal TCP connection to external tftp server	28
26	EXPLOIT x86 setgid 0	25
27	[UMBC NIDS IRC Alert] Possible Incoming XDCC Send Reque...	24
28	FTP DoS ftpd globbing	21
29	EXPLOIT NTPDX buffer overflow	14
30	TFTP - Internal UDP connection to external tftp server	13
31	DDOS mstream client to handler	13
32	RFB - Possible WinVNC - 010708-1	11
33	DDOS shaft client to handler	10
34	External FTP to HelpDesk MY.NET.53.29	6
35	HelpDesk MY.NET.70.49 to External FTP	5
36	Attempted Sun RPC high port access	4
37	Probable NMAP fingerprint attempt	3
38	TCP SMTP Source Port traffic	2
39	External FTP to HelpDesk MY.NET.70.49	2
40	TFTP - External UDP connection to internal tftp server	2

Line Item	Alert Name	# of Alerts
41	[UMBC NIDS IRC Alert] Possible drone command detected.	2
42	IRC evil - running XDCC	2
43	External RPC call	2
44	Traffic from port 53 to port 123	2
45	TFTP - External TCP connection to internal tftp server	1
46	External FTP to HelpDesk MY.NET.70.50	1
47	NIMDA - Attempt to execute cmd from campus host	1
48	Bugbear@MM virus in SMTP	1
49	EXPLOIT solaris NOOP	1
50	Samba client access	1
	Total Alerts	4,221

Table 5. Low Frequency Alerts

Top Ten Talkers (Alerts)

The chart below lists the top ten addresses that generated Snort alerts. These addresses accounted for 95% of all alerts. Both University addresses and external addresses are represented. A Whois lookup was also included to give some indication of external address ownership.

Line Item	IP Address	# of Alerts	Whois Lookup
1	MY.NET.162.118	290,602	University
2	MY.NET.150.133	72,920	University
3	169.254.244.56	8,697	IANA Special Use
4	202.31.152.174	2,002	Kwangju National University of Education
5	68.57.90.146	1,836	Comcast Cable Communications
6	MY.NET.84.224	1,290	University
7	151.196.19.202	997	Verizon Internet Services
8	62.21.236.14	851	PUBLISHNET-NL

Line Item	IP Address	# of Alerts	Whois Lookup
9	68.55.62.244	703	Comcast Cable Communications
10	MY.NET.162.41	608	University
	Total	380,506	

Table 6. Top Ten Talkers (Alerts)

Alerts Triggered By Scanning

Just fewer than seven million scan alerts were analyzed for the five-day period from October 16, 2003 to October 20, 2003. Ninety-nine point nine percent of all scan records could be attributed to SYN Scans and UDP Scans. The most unusual finding concerning the scan alerts were the high percentage of scan alerts produced by internal University systems. The chart below breaks down each type of scan alert and the related number of records for each alert. The corresponding percentages are also given. Following the chart, I've included an analysis of the SYN scan alerts and the UDP scan alerts. I selected only these two scan alerts for analysis because they comprised 99.9% of all scan alerts.

Line Item	Scan Type	# of Alerts	% of All Scan Alerts
1	SYN Scan	4,480,192	65.4%
2	UDP Scan	2,365,494	34.5%
3	FIN Scan	3,829	.05%
4	Invalid ACK Scan	432	.0055%
5	Null Scan	382	.0063%
6	Unknown Scan	287	.0042%
7	No ACK Scan	130	.0019%
8	Vecna Scan	65	.0009%
9	XMAS Scan	7	.0001%
10	NMAP ID Scan	5	.00007%
11	Full XMAS Scan	3	.00004%
12	SPAU Scan	3	.00004%
13	SYN/FIN Scan	3	.00004%

Line Item	Scan Type	# of Alerts	% of All Scan Alerts
	Total Scan Alerts	**6,850,832**	**100%**

Table 7. List of Scan Alerts

High Frequency Scan Alerts

1. SYN Scan

Synopsis. Roughly 90% of the 4,480,192 SYN scan alerts were produced by University source addresses, with the majority of the alerts addressed to external destination hosts. The only two destination ports seen with these alerts were port 80 and port 135, with port 135 comprising the overwhelming majority. In an attempt to determine a common external target, I analyzed the destination addresses, but they were fairly random. The only fact that stuck out was that foreign institutes of higher education owned most of the destination addresses.

It's probable that one of two things is occurring with these alerts: 1) the machines are infected with a virus that rapidly propagates via port 135 (e.g., W32 Blaster Worm) or 2) the machines are in fact conducting illegitimate SYN scanning. I selected the W32 Blaster Worm because this virus blew up in August 2003 and the University may not yet have a handle on the infections. If the SYN scanning is in fact illegitimate scanning, then the University has the legal and moral duty to track these machines down and ascertain the true nature of the activity. The chart below shows the top five generators of SYN scan alerts and related destination ports.

Line Item	IP Address	# of SYN Scan Alerts	Destination Port(s)
1	**MY.NET.70.154**	**1,105,525**	**80 (HTTP), 135 (epmap)**
2	**MY.NET.163.107**	**833,244**	**135 (epmap)**
3	**MY.NET.84.194**	**755,664**	**135 (epmap)**
4	**MY.NET.73.94**	**660,377**	**135 (epmap)**
5	**MY.NET.70.129**	**540,303**	**80 (HTTP),135 (epmap)**

Table 8. Top Five SYN Scan Alert Generators

Correlation. My employer's network experienced substantial problems with the W32 Blaster Worm during the same time frame as the scan alerts I analyzed from the University (October 16, 2003 – October 20, 2003). The Blaster Worm was discovered in August 2003, so the timing is close. The Blaster Worm also uses ports 4444 TCP and 69 UDP. My attempts to correlate the above IP addresses with the alerts and OOS alerts proved negative. I especially reviewed the TFTP alerts, but

found no matches.

More information on the Blaster Worm can be found at: http://securityresponse.symantec.com/avcenter/venc/data/w32.blaster.worm.html.

Defensive Recommendations. With regard to the possible Blaster problem, the University should ensure all systems are patched with current fixes and running antivirus software with current DAT files. Additionally, the University should block ports 135 and 4444 TCP, and 69 UDP. The Blaster Worm propagates via these ports and this will stop the worm from entering /exiting the University's network. Of course, the port 135 TCP block will also stop all illegitimate port 135 scanning of external addresses. An even better solution would be to block all ports at the gateway by default, and open only those necessary for business. Analysis of the alerts indicates this is not occurring.

2. UDP Scan

Synopsis. Again we find that internal University machines account for the vast majority of UDP scan alerts. Fully 99.6% of the 2,365,494 UDP scan alerts were from internal University source addresses. The top five University talkers accounted for 1,931,695 UDP scan alerts, representing 81% of the internally generated UDP scan alerts. The chart below lists the top five talkers for UDP scan alerts.

Line Item	IP Address	# of UDP Scan Alerts	Primary Port(s)
1	MY.NET.1.3	1,703,234	53 (DNS)
2	MY.NET.1.5	78,247	53 (DNS)
3	MY.NET.84.143	61,081	4673 (Kademlia)
4	MY.NET.84.232	60,074	Source Port 3383 ??
5	MY.NET.69.232	29,059	22321 (eDonkey), 7674 (Soribada)

Table 9. Top Five UDP Scan Alert Generators

Port 53 UDP scan alerts accounted for roughly 92% of all top talker generated alerts, with MY.NET.1.3 by far the most active. MY.NET.1.3 appears to be a University DNS server doing normal domain name resolution. However, the traffic is causing an extreme amount of false positives that make it difficult to detect the more important UDP scans. Alerts were generated in each of the five days, so one can assume this activity was occurring before October 16th and continued after October 20th. Given the alert count, MY.NET.1.3 is a likely the Primary DNS server and My.Net.1.5 is probably a Secondary DNS server.

UDP ports 4673, 22321 and 7674 are all related to various peer-to-peer (P2P) software programs. Specifically, port 4673 is associated with Kademlia, port 22321

is associated with eDonkey and port 7674 is used by the Soribada P2P tool. Therefore, it appears the UDP Scan alert is firing due to regular P2P activity from University hosts.

My search to discover the importance of source port 3383 didn't turn up much. The only place I could find any information regarding this port was on the IANA Port list. IANA lists this port as "Enterprise Software Products License Manager." But, I really don't think the port 3383 UDP traffic is related to this management tool. Rather, it's likely more P2P that's been unidentified or maybe even a new exploit.

Correlation. I correlated the port 53 alerts with Sebastien Pratte's GCIA Practical. Mr. Pratte also encountered a high number of DNS alerts caused by servers MY.NET.1.3 and MY.NET.1.5.

I correlated all P2P-related ports with the Dshield port reports. Interestingly enough, each reviewed port did not have any related CVE entries. This is one fact that leads me to believe that all the UDP ports listed above as P2P were actual P2P traffic, and not some attempt to compromise a vulnerability on a specific port. Dshield also reports that ports 22321, 7674 and 3383 all saw increased usage at some point between October 16, 2003 and October 20, 2003.

Defensive Recommendations. The University should consider putting in pass statements for the two DNS servers, to cut down on the number of alerts generated by these two systems. Or the port scan threshold values could also be adjusted in Snort to cut down on the number of alerts.

Depending on the University's P2P usage policy, the University must choose whether to remove the P2P software from the identified machines, or not.

Out of Spec (OOS) Log Analysis

A total of 21,733 OOS logs were reviewed for the period October 8, 2003 – October 12, 2003. External source addresses were responsible for generating 98.7% of the OOS alerts. This type of alert is generated by the combination of illegal flag or bit settings in TCP packets. The table below list the top five University destination addresses, along with the primary destination port.

Item Number	IP Address	# of OOS Alerts	Primary Port(s)
1	MY.NET.111.52	5,028	25 (SMTP)
2	MY.NET.12.6	4,447	25 (SMTP)
3	MY.NET.6.7	2,092	80 (HTTP)
4	MY.NET.24.44	1,568	80 (HTTP)
5	MY.NET.24.34	868	80 (HTTP)

Table 10. Top Five University Destination Addresses (OOS Alerts)

There were a number of different flag/bit settings present in the OOS alerts. The following chart lists the top five flag/bit settings by alert count. These top five flag/bit settings accounts for roughly 99.7% of all OOS alerts.

Item Number	Flag/Bit Set	# of OOS Alerts	% of OOS Alerts
1	12****S*	19,671	90.3%
2	********	1,923	8.8%
3	****P***	86	.4%
4	12***R**	35	.16%
5	*2U*P*SF	4	.02%

Table 11. Top Five Flag/Bit Settings (OOS Alerts)

Five Interesting External Addresses

1. 209.240.191.76 (Explanation found after #2 below)

Final results obtained from whois.arin.net.
Results:

OrgName: NetWest Online, Inc.
OrgID: NWOI
Address: 5000 E University #10
City: Odessa
StateProv: TX
PostalCode: 79762
Country: US

NetRange: 209.240.160.0 - 209.240.191.255

CIDR: 209.240.160.0/19
NetName: NWOL-BLK1
NetHandle: NET-209-240-160-0-1
Parent: NET-209-0-0-0-0
NetType: Direct Allocation
NameServer: NS.NWOL.NET
NameServer: NS2.NWOL.NET
Comment: ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
RegDate: 1998-06-16
Updated: 1999-06-28
TechHandle: AJ464-ARIN
TechName: Jenkins, Allen
TechPhone: +1-915-550-8766
TechEmail: gkins@nwol.net

2. 68.55.56.109

Final results obtained from whois.arin.net.

Results:
Comcast Cable Communications, Inc. JUMPSTART-1 (NET-68-32-0-0-1)
68.32.0.0 - 68.63.255.255
Comcast Cable Communications, Inc. BALTIMORE-A-6 (NET-68-55-0-0-1)
68.55.0.0 - 68.55.255.255

Both 209.240.191.76 and 68.55.56.109 were involved in bidirectional VNC (Virtual Network Computing) communications with University hosts. The related alert rule was (RFB - Possible WinVNC - 010708-1), and fired on activity to or from ports 5900 or 5901. VNC is a freeware software tool that allows users to view or administer machines remotely over networks, including the Internet. This tool works similarly to the well-known PC Anywhere product. The alerts related to this activity should definitely be investigated, unless this product has been authorized with the involved IP addresses. If VNC is authorized, then the University should mandate the encryption option be employed to keep passwords from being sent in the clear. The diagram below gives a good rendering of the power of VNC.

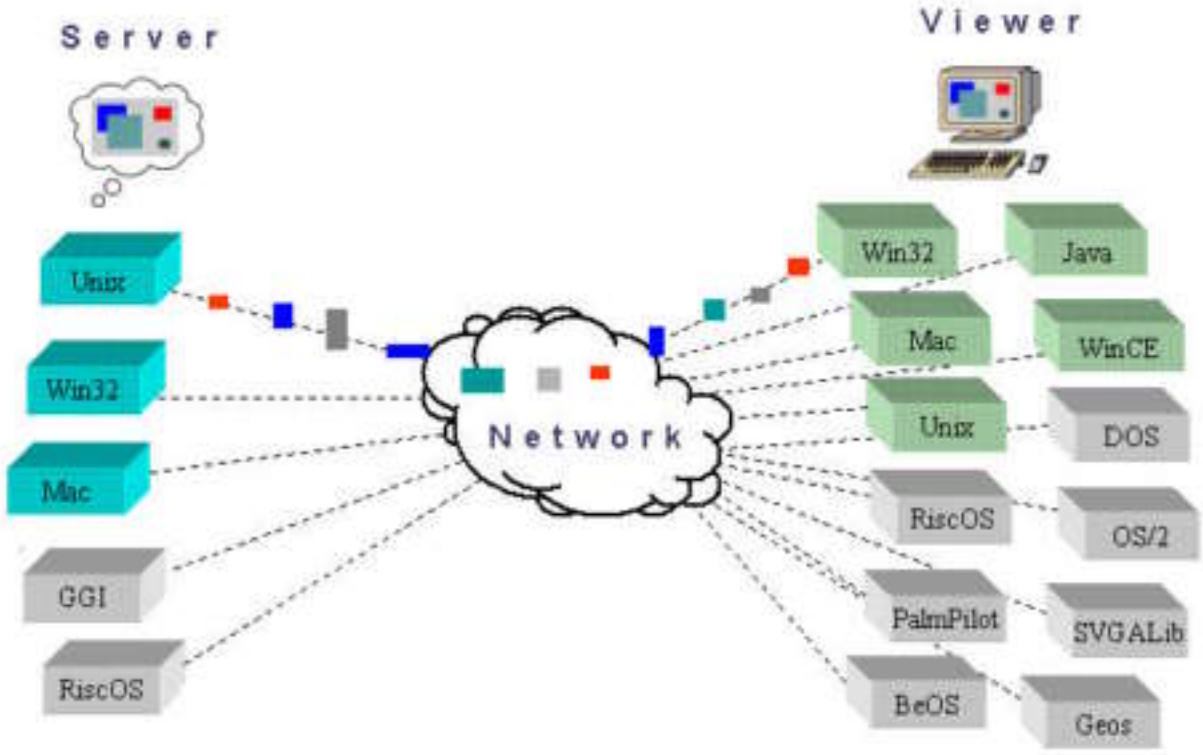


Figure 6. Virtual Network Computing (VNC) Diagram (Courtesy of RealVNC)

3. 218.16.124.131 (Explanation found after #5 below)

Final results obtained from whois.apnic.net.
Results:

% [whois.apnic.net node-2]
% Whois data copyright terms http://www.apnic.net/db/dbcopyright.html

inetnum: 218.13.0.0 - 218.18.255.255
netname: CHINANET-GD
descr: CHINANET Guangdong province network
descr: Data Communication Division
descr: China Telecom
country: CN
admin-c: CH93-AP
tech-c: WM12-AP
mnt-by: MAINT-CHINANET
mnt-lower: MAINT-CHINANET-GD
changed: hostmaster@ns.chinanet.cn.net 20010528
status: ALLOCATED PORTABLE
source: APNIC
person: Chinanet Hostmaster

address: No.31 ,jingrong street,beijing
address: 100032
country: CN
phone: +86-10-66027112
fax-no: +86-10-66027334
e-mail: hostmaster@ns.chinanet.cn.net
e-mail: anti-spam@ns.chinanet.cn.net
nic-hdl: CH93-AP
mnt-by: MAINT-CHINANET
changed: hostmaster@ns.chinanet.cn.net 20021016
source: APNIC

person: WU MIAN
address: NO.1,RO.DONGYUANHENG,YUEXIUNAN,GUANGZHOU
country: CN
phone: +086-20-83877223
fax-no: +86-20-83877223
e-mail: ipadm@gddc.com.cn
nic-hdl: WM12-AP
mnt-by: MAINT-CHINANET-GD
changed: ipadm@gddc.com.cn 20010820
source: APNIC

4. 211.91.144.72 (Explanation found after #5 below)

Final results obtained from whois.apnic.net.
Results:

% [whois.apnic.net node-2]
% Whois data copyright terms http://www.apnic.net/db/dbcopyright.html

inetnum: 211.90.0.0 - 211.91.255.255
netname: UNICOM
country: CN
descr: China United Telecommunications Corporation
admin-c: UCH1-AP
tech-c: UC6-AP
status: ALLOCATED PORTABLE
changed: ipas@cnnic.net.cn 20020917
mnt-by: MAINT-CNNIC-AP
source: APNIC

role: Unicom China Hostmaster
address: 911 Room,Xin Tong Center,No.8 Beijing Railway Station
address: East Avenue, Beijing,PRC.
country: CN
phone: +86-10-6527-8866

fax-no: +86-10-6526-0124
e-mail: ip_address@cnuninet.com
admin-c: RX9-AP
tech-c: RX9-AP
nic-hdl: UCH1-AP
notify: ip_address@cnuninet.com
mnt-by: MAINT-CN-CNNIC-UNICOM
changed: hostmaster@apnic.net 20010820
source: APNIC

person: Unicom China
address: 911 Room,Xin Tong Center,No.8 Beijing Railway Station
address: East Avenue, Beijing,PRC.
country: CN
phone: +86-10-6527-8866
fax-no: +86-10-6526-0124
e-mail: ip_address@cnuninet.com
nic-hdl: UC6-AP
mnt-by: MAINT-CNNIC-AP
changed: ip_address@cnuninet.com 20010521
changed: hostmaster@apnic.net 20010820
source: APNIC

5. 202.114.103.130

Final results obtained from whois.apnic.net.
Results:
% [whois.apnic.net node-1]
% Whois data copyright terms http://www.apnic.net/db/dbcopyright.html

inetnum: 202.114.96.0 - 202.114.111.255
netname: WUHEE-CN
descr: ~{Nd::K.@{K.5g4sQ'~}
descr: Wuhan University of Hydrualic & Electric Engineering
descr: Hubei, Wuhan
country: CN
admin-c: XT2-CN
tech-c: JL1-CN
tech-c: CER-AP
remarks: origin AS4538
changed: hm-changed@net.edu.cn 19951225
mnt-by: MAINT-CERNET-AP
status: ASSIGNED NON-PORTABLE
source: APNIC

role: CERNET Helpdesk
address: Room 224, Main Building

address: Tsinghua University
address: Beijing 100084, China
country: CN
phone: +86-10-6278-4049
fax-no: +86-10-6278-5933
e-mail: cernet-helpdesk-ip@net.edu.cn
trouble: abuse@net.edu.cn
admin-c: XL1-CN
tech-c: SZ2-AP
nic-hdl: CER-AP
remarks: Point of Contact for admin-c
mnt-by: MAINT-CERNET-AP
changed: cernet-helpdesk-ip@net.edu.cn 20010903
source: APNIC

person: Xuzhang Tang
address: ~{Nd::K.@{K.5g4sQ'~}
address: Computer Center
address: Wuhan University of Hydrualic & Electric Engineering
address: Hubei,Wuhan 430072
address: China
phone: +86-27-7884555ext.2120
fax-no: +86-27-7884496
e-mail: xpwang@hustcc.whnet.edu.cn
nic-hdl: XT2-CN
notify: address-allocation-staff@cernic.net
mnt-by: MAINT-NULL
changed: szhu@cernic.net 19951225
source: APNIC

person: June Li
address: ~{Nd::K.@{K.5g4sQ'~}
address: Computer Center
address: Wuhan University of Hydrualic & Electric Engineering
address: Hubei,Wuhan 430072
address: China
phone: +86-27-7868613
fax-no: +86-27-7884496
e-mail: xpwang@hustcc.whnet.edu.cn
nic-hdl: JL1-CN
notify: address-allocation-staff@cernic.net
mnt-by: MAINT-NULL
changed: szhu@cernic.net 19951225
source: APNIC

External IP addresses 218.16.124.131, 211.91.144.72 and 202.114.103.130 were all involved with alert (TCP SRC and DST Outside Network). These addresses were destination addresses, however I believe they initially elicited some stimulus that is causing an undefined response from University machines. Thus, they could be considered source entities. As for the stimulus these machines sent, I'm just not sure. I analyzed these machines as part of my analysis on the (TCP SRC and DST Outside Network) alert, but was unable to come up with a valid conclusion as to what was causing the return traffic to these machines. It is my inability to fully conclude what was going on that caused me to include these addresses in my "External Lookups". These are addresses that I would tag as "Addresses to Watch" if I were monitoring the University's network on a regular basis.

Analysis Process

1. Process For Alert Logs

The alert logs were first concatenated into one single file for ease of use and added log continuity. I then comma delimited the file with a Perl Script borrowed from Tod Beardsley's GCIA Practical. I also utilized a second Perl script borrowed from Tod Beardsley's GCIA Practical to generate a useful report on the data. Some of the reports created include: List of alerts by alert count, top source/destination talkers, top source/destination ports, machine relationships, etc.

The comma delimited alert logs were imported into an Access database. This allowed me to provide alert counts, as well as sort the logs by IP address, Port, Time, etc. During the import process, 5,352 logs were lost due to probable improper log format.

2. Process For Scan Logs

I also comma separated the scan logs with the aforementioned Perl script. The scan logs were then loaded into an Access database individually by date. However, I loaded each file into the same database table. Thus, I ended up with one huge table that I used to manipulate all the scan data at the same time. A total of 7,131 log entries were lost during the import process, likely due to incorrectly formatted entries.

3. Process For OOS Logs

Te OOS files were simply concatenated and imported into an Excel spreadsheet. The spreadsheet was then used to organize the data for analysis and extrapolation. All analysis of the OOS logs was done manually.

4. Equipment Used

Compaq Laptop
ActivePerl 5.8.1.807
Snort v.2.0.2

Microsoft Access 2000
Microsoft Excel 2000
Microsoft Word 2000

Appendix A - Perl Scripts

1) Perl script used to pull Snort scan preprocessor data from the alert file and format the file into .csv for parsing into database. (Courtesy of Tod Beardsley)

```
#!/cygdrive/c/Perl/bin/perl.exe -w

# Name: csv.pl

# Reads in a Snort -A Fast style alert log which for some
# reason wasn't generated as CSV, and make it as such.
#
# Usage: csv.pl infile [outfile]

unless ($ARGV[0]) {
  print "Need an input file!\n";
  die "(Hint: go to http://www.research.umbc.edu/~andy and get one)\n";
}

unless ($ARGV[1]) {
  $outfile = "$ARGV[0].csv";
} else {
  $outfile = "$ARGV[1]";
}

open(INFILE,"$ARGV[0]") || die "Can't open $ARGV[0] for reading!\n";
open(OUTFILE,">$outfile") || die "Can't open $ARGV[1] for writing!\n";

print "Transforming $ARGV[0] into $outfile.\n";
print "Just a moment.";

@calendar=qw(Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec);

while (<INFILE>) {
  next unless /(\w{1,3}\.){2}(\d{1,3}\.\d{1,3})/;       # Skip lines missing IPv4 IPs.
  next if /spp_portscan/;                               # Skip portscan notifications.
  chomp;
  if (/ \[\*\*\] /) {                                   # Alert report.

   ($date_and_time,$alert,$src_and_dst) = split(/\s+\[\*\*\]\s/);
   ($date,$time) = split(/-/,$date_and_time);
   ($month_number,$day) = split(/\//,$date);
   $month = $calendar[$month_number-1];
```

```
 ($src,$dst) = split(/\s-\>\s/,$src_and_dst);
 ($src_ip,$src_port) = split(/:/,$src);
 ($dst_ip,$dst_port) = split(/:/,$dst);
 $snort_entry="ALERT" ;

 } else {                                                    # Scan report.
 ($month,$day,$time,$src,$arrow,$dst,$alert,$flags) = split;
 undef $arrow;
 ($src_ip,$src_port) = split(/:/,$src);
 $alert = "$alert scan (Internally-based)" if $src_ip =~ /^MY\.NET/;
 $alert = "$alert scan (Externally-based)" unless $src_ip =~ /^MY\.NET/;
 ($dst_ip,$dst_port) = split(/:/,$dst);
 $snort_entry="SCAN" ;
}

 print OUTFILE "$snort_entry,";
 print OUTFILE "$month,$day,$time,$alert,";
 print OUTFILE "$src_ip,";
 print OUTFILE "$src_port" if $src_port;
 print OUTFILE "None" unless $src_port;
 print OUTFILE ",";
 print OUTFILE "$dst_ip";
 print OUTFILE ",";
 print OUTFILE "$dst_port" if $dst_port;
 print OUTFILE "," if $flags;
 print OUTFILE "None," unless $dst_port;
 print OUTFILE "$flags" if $flags;
 print OUTFILE "\n";

   $happydots++;

   print "." if $happydots % 100 == 0; # if $happydots == 100;

   print "Just a moment." if $happydots % 46600 == 0;
}
```

2) Perl script used to generate a test summary of all alerts. (Courtesy of Tod Beardsley)

```
#!/cygdrive/c/Perl/bin/perl.exe -w
#
# Initialise variables
$HOME_NET="MY.NET";
$SUM_EXT="alert_sum.csv";
@reports1 = (
      [ "uniq_ext_srcip.txt", "external sources", "ext_src_ip_ct" ],
      [ "uniq_ext_dstip.txt", "external destinations", "ext_dst_ip_ct" ],
```

```
        [ "uniq_ext_src_port.txt", "external source ports", "ext_src_port_ct" ],
        [ "uniq_ext_dst_port.txt", "external destination ports", "ext_dst_port_ct" ],
        [ "uniq_int_srcip.txt", "internal sources", "int_src_ip_ct" ],
        [ "uniq_int_dstip.txt", "internal destinations", "int_dst_ip_ct" ],
        [ "uniq_int_src_port.txt", "internal source ports", "int_src_port_ct" ],
        [ "uniq_int_dst_port.txt", "internal destination ports", "int_dst_port_ct" ]
);
@reports2 = (
        [ "uniq_in_src_dst_ip.txt", "inbound src/dst pairs", "inbound_pair_ct" ],
        [ "uniq_in_src_dst_port.txt", "inbound src/dst port pairs", "inbound_port_pair_ct" ],
        [ "uniq_out_src_dst_ip.txt", "outbound src/dst pairs", "outbound_pair_ct" ],
        [ "uniq_out_src_dst_port.txt", "outbound src/dst port pairs", "outbound_port_pair_ct" ],
       [  "uniq_int_src_dst_ip.txt", "internal src/dst pairs", "internal_pair_ct" ],
        [ "uniq_int_src_dst_port.txt", "internal src/dst port pairs", "internal_port_pair_ct" ],
        [ "uniq_ext_src_dst_ip.txt", "external src/dst pairs", "external_pair_ct" ],
        [ "uniq_ext_src_dst_port.txt", "external src/dst port pairs", "external_port_pair_ct" ]
);
$sum_fname="summary.csv";
$display_ct=0;
@display_char=('-','\\','|','/','-');

# Check command-line arguments exist
&usage unless ($out_base=$ARGV[1]);
&usage unless ($alertsfile=$ARGV[0]);

# attempt to open files
open(ALERTSFILE,"$alertsfile") || die "ERROR! - Can't open alerts-file:$alertsfile $!\n";
open(SUMFILE,"|sort -t ',' -k 2,2rn >$out_base-$sum_fname") || die
            "ERROR! - Can't open output-file:$out_base$SUM_EXT $!\n";
for $i ( 0 .. $#reports1) {
        $fname = $reports1[$i][0];
        &open_output_file($fname, '2', $out_base, $fname);
}
for $i ( 0 .. $#reports2) {
        $fname = $reports2[$i][0];
        &open_output_file($fname, '3', $out_base, $fname);
}
# read alerts file and do sums
print "Reading alert data into memory.....";
while (<ALERTSFILE>) {
        &progress; &process_alertsfile;
}
print "\b\b\b   \nCounting unique external destinations.....\n";
foreach $key (keys(%ext_dst_by_alert_ct)) {
        ($ip_key,$alert_key) = split(/\@/,$key);
        $uniq_ext_dst_by_alert{$alert_key}++; $uniq_ext_dst_ct++;
```

```
}
print "Counting unique external sources.....\n";
foreach $key (keys(%ext_src_by_alert_ct)) {
      ($ip_key,$alert_key) = split(/\@/,$key);
       $uniq_ext_src_by_alert{$alert_key}++; $uniq_ext_src_ct++;
}
print "Counting unique internal destinations.....\n";
foreach $key (keys(%int_dst_by_alert_ct)) {
      ($ip_key,$alert_key) = split(/\@/,$key);
      $uniq_int_dst_by_alert{$alert_key}++; $uniq_int_dst_ct++;
}
print "Counting unique internal sources.....\n";
foreach $key (keys(%int_src_by_alert_ct)) {
      ($ip_key,$alert_key) = split(/\@/,$key);
      $uniq_int_src_by_alert{$alert_key}++; $uniq_int_src_ct++;
}
print "Creating alert summary.....\n";
foreach $alert_name (sort keys(%alert_ct)) {
      &progress; &sum_each_alert;
}
print "\b\b\b   \n\nTotal no. of alerts processed: ",$total_alerts_ct,"\n";
print SUMFILE "Alert name,# alerts,# Ext Src,# Int Dst,# Int Src,# Ext Dst,";
print SUMFILE "# Inbound,# Outbound,# I->I,# E->E\n";
print SUMFILE "Totals:,$total_alerts_ct,$uniq_ext_src_ct,$uniq_int_dst_ct,";
print SUMFILE "$uniq_int_src_ct,$uniq_ext_dst_ct,$inbound_total_ct,";
print SUMFILE "$outbound_total_ct,$internal_total_ct,$external_total_ct\n";

for $i ( 0 .. $#reports1) {    # Produce unique totals reports
      print "\b\b\b   \nCreating unique $reports1[$i][1] file.....";
      foreach $key (keys(%{$reports1[$i][2]})) {
            &progress;
            print {$reports1[$i][0]} "$key","\t","${$reports1[$i][2]}{$key}\n";
      }
}
for $i ( 0 .. $#reports2) {    # Produce relationship reports
      print "\b\b\b   \nCreating unique $reports2[$i][1] file.....";
     fo reach $key (keys(%{$reports2[$i][2]})) {
            &progress; ($keya, $keyb) = split(/\@/,$key);
            print {$reports2[$i][0]} "$keya", "\t", $keyb, "\t", "${$reports2[$i][2]}{$key}\n";
      }
}
print "\b\b\b   \nFinished!\n\n";

# Subroutines
sub usage {
      die "\nUsage: summary.pl alert-file output-file-basename\n\n";
```

```
}
sub process_alertsfile {
      chomp;
      ($timestamp,$alert,$srcip,$srcport,$dstip,$dstport) = split(/\,/,$_);
      $alert_ct{"$alert"}++;
      $total_alerts_ct++;

      if($srcip =~ "^$HOME_NET") {    # Internal Src
            $int_src_ip_ct{$srcip}++;               # Count uniq source IPs
            $int_src_by_alert_ct{"$srcip"."@"."$alert"}++; # As above, by alert
            $in t_src_port_ct{$srcport}++;           # Count uniq src ports

            if($dstip =~ "^$HOME_NET") {    # Internal Src / Internal Dst
                  $int_dst_ip_ct{$dstip}++;         # Count uniq dest IPs
                  $int_dst_by_alert_ct{"$dstip"."@"."$alert"}++; # As above, by alert
                  $int_dst_port_ct{$dstport}++;      # Count uniq dest ports
                  $internal_total_ct++;             # Count tot no. of Int-Int alerts
                  $internal_ct{$alert}++;           # As above, by alert
                  $internal_pair_ct{"$srcip"."@"."$dstip"}++; # Count Src/dst pairs
                  $internal_port_ pair_ct{"$srcport"."@"."$dstport"}++; # Count port pairs
            }
            elsif($dstip !~ "^$HOME_NET") {        # Internal Src / External Dst
                  $ext_dst_ip_ct{$dstip}++;
                  $ext_dst_by_alert_ct{"$dstip"."@"."$alert"}++;
                  $ext_dst_port_ct{$dstport}++;
                  $outbound_total_ct++;
                  $outbound_ct{$alert}++;
                  $outbound_pair_ct{"$srcip"."@"."$dstip"}++;
                  $outbound_port_pair_ct{"$srcport"."@"."$dstport"}++;
            }
      }
      elsif($srcip !~ "^$HOME_NET") {        # External Src
            $ext_src_ip_ct{$srcip}++;
            $ext_src_by_alert_ct{"$srcip"."@"."$alert"}++;
            $ext_src_port_ct{$srcport}++;
            if( $dstip !~ "^$HOME_NET") {    # External Src / External Dst
                  $ext_dst_ip_ct{$dstip}++;
                  $ext_dst_by_alert_ct{"$dstip"."@"."$alert"}++;
                  $ext_dst_port_ct{$dstport}++;
                  $external_total_ct++;
                  $external_ct{$alert}++;
                  $external_pair_ct{"$srcip"."@"."$dstip"}++;
                  $external_port_pair_ct{"$srcport"."@"."$dstport"}++;
            }
            elsif($dstip =~ "^$HOME_NET") {   # External Src / Internal Dst
                  $int_dst_ip_ct{$dstip}++;
```

```
                $int_dst_by_alert_ct{$dstip."@".$alert}++;
                $int_dst_port_ct{$dstport}++;
               $i  nboun d_total_ct++;
                $inbound_ct{$alert}++;
                $inbound_pair_ct{"$srcip"."@"."$dstip"}++;
                $inbound_port_pair_ct{"$srcport"."@"."$dstport"}++;
            }
            else {   # Th is should never happen - Something just has to be broken!
                 die "ERROR! - Unexpect condition - cannot continue!\n";
            }
        }
}
sub sum_each_alert {
        print SUMFILE $alert_name,",";
        print SUMFILE $alert_ct{$alert_name},",";
        print SUMFILE $uniq_ext_src_by_alert{$alert_name},",";
        print SUMFILE $uniq_int_dst_by_alert{$alert_name},",";
        print SUMFILE $uniq_int_src_by_alert{$alert_name},",";
        print SUMFILE $uniq_ext_d st_by_alert{$alert_name},",";
        print SUMFILE $inbound_ ct{$alert_name},",";
        print SUMFILE $outbound_ct{$alert_name},",";
        print SUMFILE $internal_ct{$alert_name},",";
        print SUMFILE $external_ct{$alert_name},"\n";
}
sub progress {
        $lines++;
        if ($lines % 50 == 0) {
            if ($display_ct == @display_char) {
                $display_ct=0;
            }
            $char=@display_char;

            print "\b\b",$display_char[$display_ct]," ";
            $display_ct++;
        }
}
sub open_output_file {
        local($handle, $sortkey, $basename, $fname) = @_;
        open($handle,"| sort -rnk $sortkey,$sortkey >$basename-$fname") || die
            "ERROR! - Can't open output-file: $basenam e-$fname $!\n";
}
```

References

Cahoon, Brian. “GIAC Certified Intrusion Analyst (GCIA) Version 3.3.” January 6, 2002. http://www.giac.org/practical/Brian_Cahoon_GCIA.pdf. (11 Nov 2003).

Carnegie Mellon Software Engineering Institute, CERT® Coordination Center. “CERT Incident No IN-2000-10.” September 15, 2000. http://www.cert.org/incident_notes/IN-2000-10.html. (12 Nov 2003).

Carnegie Mellon Software Engineering Institute, CERT® Coordination Center. “CERT Incident No IN-99-07.” January 15, 2001. http://www.cert.org/incident_notes/IN-99-07.html. (12 Nov 2003).

Carnegie Mellon Software Engineering Institute, CERT® Coordination Center. “CERT® Advisory CA-2001-19 “Code Red” Worm Exploiting Buffer Overflow in IIS Indexing Service DLL.” January 17, 2002. http://www.cert.org/advisories/CA-2001-19.html. (7 Nov 2003).

Carnegie Mellon Software Engineering Institute, CERT® Coordination Center. “CERT® Advisory CA-2001-13 Buffer Overflow In IIS Indexing Service DLL.” January 17, 2002. http://www.cert.org/advisories/CA-2001-12.html. (7 Nov 2003).

Caswell, Brian and Roesch, Marty. “Snort Signature Database.” Wed, Nov 12 2003. http://www.snort.org/snort-db/sid.html?sid=251. (12 Nov 2003).

Cisco. “Cisco Security Advisory: “Code Red” Worm – Customer Impact.” 2001 November 01. http://www.cisco.com/warp/public/707cisco-code-red-worm-pub.shtml. (7 Nov 2003).

Cisco. “Cisco Security Advisory: “Multiple Vulnerabilities in CBOS.” 2001 August 01. http://www.cisco.com/warp/public/707/CBOS-multiple.shtml. (7 Nov 2003).

Common Vulnerabilities and Exposures. “CAN-2000-0138.” http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0138. (12 Nov 2003).

Common Vulnerabilities and Exposures. “CVE-2001-0500.” 20020309. http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0500. (7 Nov 2003).

Dshield.org. Distributed Intrusion Detection Center. http://www.dshield.org. (13 Nov 2003).

Dittrich, David. “The Tribe Flood Network Distributed Denial of Service Attack Tool.” October 21, 1999. http://staff.washington.edu/dittrich/misc/tfn.analysis. (11 Nov 2003).

eEye® Digital Security. ".ida "Code Red" Worm." http://www.eeye.com/html/Research/Advisories/AL20010717.html. (7 Nov 2003).

GFI.com. "Virus Alert: Nothing Romantic About New Romeo and Juliet Virus." 17 November 2000. http://www.gfi.com/news/en/romeoalert.htm. (14 Nov 2003).

http://www.giac.org/practical/Steve_Lukacs_GCIA.doc. (13 Nov 2003).

http://www.giac.org/practical/Miika_Turkia_GCIA.html. (13 Nov 2003).

http://cert.uni-stuttgart.de/archive/intrusions/2003/02/msg00055.html. (8 Nov 2003).

http://cert.uni-stuttgart.de/archive/intrusions/2002/08/msg00106.html. (8 Nov 2003).

Incidents.org. http://www.incidents.org/logs/Raw/2002.4.16. (15 Oct 2003).

Internet Engineering Task Force. "RFC 792, Internet Control Message Protocol." September 1981. http://www.ieft.org/rfc/rfc0792.txt. (11 Nov 2003).

Japan Network Information Center (JPNIC). http://www.nic.ad.jp/. (12 Nov 2003).

Johnson, Rick. "Linux Security-Tribal Flood Network." September 19, 2000. http://www.rickjohnson.org/writing/itworld/msg00043.html. (12 Nov 2003).

Leuning, Erich. "Romeo and Juliet Bug Spreads on Outlook." November 21, 2000. http://news.com.com/2100-1023_3-248938.html. (14 Nov 2003).

Microsoft.com. "Microsoft Security Bulletin MS01-033." November 11, 2003. http://www.microsoft.com/technet/security/bulletin/MS01-033.asp?frame=true. (7 Nov 2003).

Microsoft.com. "Microsoft Security Bulletin (MS99-042)." November 4, 1999. http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms99-042.asp. (14 Nov 2003).

Microsoft.com. "Microsoft Security Bulletin (MS00-046)." July 20, 2000. http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/MS00-046.asp. (14 Nov 2003)

Microsoft.com. "Microsoft Security Bulletin (MS99-032)." March 21, 2003. (http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms99-032.asp. (14 Nov 2003).

Microsoft.com. "Microsoft Security Bulletin (MS00-037)." June 02, 2000. (http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms00-037.asp. (14 Nov 2003).

Microsoft.com. “Security Update.” 18 June 2001. http://www.microsoft.com/Windows2000/downloads/critical/q300972/download.asp. (8 Nov 2003).

Naidu, Krishni. “Firewall Checklist.” http://www.sans.org/score/checklists/FirewallChecklist.pdf. (13 Nov 2003).

Network Associates. “W32/BleBla.a@MM.” http://vil.nai.com/vil/content/v_98894.htm. (14 Nov 2003).

PCWorld.com. “The Dawn of Orchestrated Attacks.” http://www.pcworld.com/news/article/0,aid,15199,pg,2,00.asp. (12 Nov 2003).

Security Focus-Bugtraq. “MS Index Server and Indexing Service ISAPI Extension Buffer Overflow Vulnerability.” Aug 10, 2001. http://www.securityfocus.com/bid/2880. (7 Nov 2003).

Snort.org. http://www.Snort.org. Snort 2.0.2 Download For WIN32. (14 Oct 2003).

Snort.org. “Snort Signature Database.” http://www.snort.org/snort-db/sid.html?sid=1322. (13 Nov 2003).

Snort.org. “Snort Signature Database.” http://www.snort.org/snort-db/sid.html?sid=251. (13 Nov 2003).

Snort.org. “Snort Signature Database.” http://www.snort.org/snort-db/sid.html?sid=1242. (13 Nov 2003).

Snort.org. “Snort Signature Database.” http://www.snort.org/snort-db/sid.html?sid=1243. (13 Nov 2003).

Snort.org. “Snort Signature Database.” http://www.snort.org/snort-db/sid.html?sid=726. (14 Nov 2003).

Szor, Peter. “W32.BleBla.worm.” http://securityresponse.symantec.com/avcenter/venc/data/w32.blebla.worm.html. (14 Nov 2003).

Uimonen, Terho. “Low-Risk Internet Worm Making the Rounds.” November 22, 2000. http://archive.infoworld.com/articles/hn/xml/00/11/22/001122hnworm.xml?p=br&s=2. (14 Nov 2003).

Viruslist.com. “I-Worm.Blebla (a.k.a. Verona and Romeo&Juliet).” http://www.viruslist.com/eng/viruslist.html?id=4116. (14 Nov 2003).

Whitehats.com “DDOS-TFN-CLIENT-COMMAND-LE.” 2001. http://www.whitehats.com/info/IDS183. (12 Nov 2003).

Whitehats.com. “DDOS-TFN-CLIENT-COMMAND-LE.” 2001. http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids183&view=protocol (12 Nov 2003).

Whitehats.com. “DDOS-TFN-CLIENT-COMMAND-LE.” 2001. http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids183&view=research (12 Nov 2003).

Whois.sc. http://www.whois.sc/216.111.190.2. (14 Nov 2003).

Assignment 3: Analyze This

Adams, Samuel. “Fun With Intrusion Detection, SANS GIAC GCIA v3.3.” 23 June 2003. http://www.giac.org/practical/Samuel_Adams_GCIA.pdf. (23 Nov 2003).

Antoine, Vanessa and many others. “National Security Agency Router Security Configuration Guide.” September 27, 2002. nsa2.www.conxion.com/cisco/guides/cis-2.pdf. (24 Nov 2003).

Beardsley, Tod. “Intrusion Detection and Analysis: Theory, Techniques and Tools.” May 8, 2002. http://www.giac.org/practical/Tod_Beardsley_GCIA.doc. (17 Nov 2003).

Bindview Razor. “Object Enumeration in Novell Environments.” 11/08/00. http://razor.bindview.com/publish/advisories/adv_novellleak.html. (23 Nov 2003).

Blackcode.com. “Trojan Library: Details of Bad Blood.” http://www.blackcode.com/trojans/details.php?id=134. (24 Nov 2003).

Carnegie Mellon Software Engineering Institute, CERT® Coordination Center. “CERT Incident Note IN-2001-01.” January 18, 2001. http://www.cert.org/incident_notes/IN-2001-01.html. (24 Nov 2003).

Chapman, Todd. “SANS GIAC Level Two – Intrusion Detection In Depth, SANS GCIA Practical Assignment, Version 3.0.” http://www.giac.org/practical/Todd_Chapman_GCIA.doc. (24 Nov 2003).

Common Vulnerabilities and Exposures. “CVE-2000-0232.” http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0232. (24 Nov 2003).

Common Vulnerabilities and Exposures. “CVE-2000-0839.” http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0839. (24 Nov 2003).

Common Vulnerabilities and Exposures. “CVE-2000-1208.” http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0839. (24 Nov 2003).

Common Vulnerabilities and Exposures. “CVE-2001-0906.” http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0839. (24 Nov 2003).

Common Vulnerabilities and Exposures. “CVE-1999-0032.” http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0839. (24 Nov 2003).

Common Vulnerabilities and Exposures. “CVE-1999-0335.” http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0839. (24 Nov 2003).

Common Vulnerabilities and Exposures. “CVE-1999-1102.” http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0839. (24 Nov 2003).

Common Vulnerabilities and Exposures. “CVE-2003-0144.” http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0839. (24 Nov 2003).

Dshield.org. “Port Report.” http://www.dshield.org/port_report.php?port=3383&recax=1&tarax=2&srcax=2&percent=N&days=70&Redraw=Submit. (30 Nov 2003).

Dshield.org. “Port Report.” http://www.dshield.org/port_report.php?port=22321&recax=1&tarax=2&srcax=2&percent=N&days=70&Redraw=Submit. (30 Nov 2003).

Dshield.org. “Port Report.” http://www.dshield.org/port_report.php?port=7674&recax=1&tarax=2&srcax=2&percent=N&days=70&Redraw=Submit. (30 Nov 2003).

Dshield.org. “Port Report.” http://www.dshield.org/port_report.php?port=4673&recax=1&tarax=2&srcax=2&percent=N&days=70&Redraw=Submit. (30 Nov 2003).

Geektools.com. ”Whois Search.” http://www.geektools.com. (20 Nov 2003).

Hackworth, Aaron. “SANS GIAC Level II-Intrusion Detection In Depth.” May 20, 2002. http://www.giac.org/practical/Aaron_Hackworth_GCIA.pdf. (23 Nov 2003).

Hally, John. “SANS Training & GIAC Certification, Intrusion Detection In Depth, GCIA Practical Assignment, Version 3.1.” 7/13/02. www.giac.org/practical/GCIA/John_Hally_GCIA.pdf. (24 Nov 2003).

http://www.sophos.com/virusinfo/analyses/w32opaserva.html. “W32 OpaServ -A.” (23 Nov 2003).

Hyperdictionary.com. “Netware Core Protocol.” http://www.hyperdictionary.com/computing/netware+core+protocol (23 Nov 2003).

IANA.org. “Port Numbers.” http://www.iana.org/assignments/port-numbers. (23 Nov 2003)

Insecure.org Mailing List. Mar 11, 2000. http://seclists.org/. (23 Nov 2003).

Internet Storm Center (Ken). “User Comment Port 137.” http://isc.incidents.org/show_comment.html?id=85. (23 Nov 2003).

Internet Storm Center. “Port Reports.” http://isc.incidents.org/port_details.html?port=137. (23 Nov 2003).

Knowles, Douglas, Perriot, Frederick, Szor, Peter. “W32.Blaster.Worm.” October 8, 2003. http://securityresponse.symantec.com/avcenter/venc/data/w32.blaster.worm.html. (28 Nov 2003).

Kuethe, Chris. http://www.giac.org/practical/chris_kuethe_gcia.html. (23 Nov 2003).

Maymounkov, Petar. “Kadlemia.” 2002. http://kademlia.scs.cs.nyu.edu/. (30 Nov 2003).

Myung-Sup Kim, Hun-Jung Kang, and James W. Hong. “Towards Peer-To-Peer Traffic Analysis Using Flow”. http://dpnm.postech.ac.kr/papers/DSOM/03/P2P/p2p-traffic-analysis.pdf. (30 Nov 2003).

Novell Netware. “Netware 6.0 Support Pack 2.” 11 Sep 2002. http://www.novell.com/documentation/lg/nw42/index.html?cncptenu/data/hxm2lg67.html (23 Nov 2003).

Oborn, David. “SANS GCIA Practical Assignment.” http://www.giac.org/practical/David_Oborn_GCIA.html. (23 Nov 2003).

Pederson, Stephen. “Intrusion Detection In Depth GCIA Practical Assignment, Version 2.9.” May 2001. http://www.giac.org/practical/Stephen_Pedersen_GCIA.DOC. (24 Nov 2003).

RealVNC.com. “What is VNC?” 2002. http://www.realvnc.com/what.html (26 Nov 2003).

Seifried, Kurt. “Port 5901 TCP, UDP.” 6/13/2002. http://www.seifried.org/security/ports/5000/5901.html (26 Nov 2003).

Snort.org. Rules database. http://www.snort.org/snort-db. (2 Nov 2003)

Snort.org. “Snort Signature Database.” http://www.snort.org/snort-db/sid.html?sid=648. (23 Nov 2003).

Snort.org. “Snort Signature Database.” http://www.snort.org/snort-db/sid.html?sid=1394. (23 Nov 2003).

TrendMicro. “Worm Bugbear.A.” http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=WORM_BUGBEAR.A. (23 Nov 2003).

Von Braun, Joakim. “Intrusion Detection FAQ: What Port Numbers Do Well-Known Trojan Horses Use.” http://www.sans.org/resources/idfaq/oddports.php. (24 Nov 2003).

Whitehats.com. IDS181 “Shellcode-X86-NOPS.” http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids181&view=event. (23 Nov 2003).

Whois.nic.or.kr. http://whois.nic.or.kr/english/index.html. (26 Nov 2003).

Pratte, Sebastien. “GIAC Certifed Intrusion Analyst (GCIA), Practical Assignment, Version 3.2.” August 12, 2002. www.giac.org/practical/GCIA/Sebastien_Pratte_GCIA.pdf. (30 Nov 2003).