GIAC
CERTIFICATIONS

Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

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

Interested in learning more?

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

SANS Intrusion Detection in Depth
GCIA Practical Assignment
Version 3.0

CDI EAST, Washington D.C.
December 2001

© SANS Institute 2000 - 2002, Author retains full rights.

Prepared By:
Angela D. Orebaugh, GSEC, GCFW, GCIH
March 11, 2002

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

EXECUTIVE SUMMARY

This document presents the Intrusion Detection In Depth GCIA Practical Assignment Version 3.0. Its purpose is to demonstrate my mastery of the course material and to assist in improving the state of practice of information security. The practical consists of the following three related assignments: Describe the State of Intrusion Detection, Network Detects, and the "Analyze This" Scenario.

The first assignment consists of a white paper on an intrusion technology or challenge. I have chosen to discuss Intrusion Detection System (IDS) Testing. I discuss the need for IDS testing and the testing methodology used in both the pre-deployment phase and the post-deployment phase.

The second assignment consists of five network detects with analysis. The analysis includes ten predefined items to be discussed. These items discuss detect attributes such as source, description, correlations, severity, defense recommendation, etc. Each analysis also includes one multiple-choice question relating to the detect.

The third assignment consists of a security audit scenario for a university. Data is provided in the form of ASCII text files that are output from a Snort intrusion detection system. The assignment consists of analyzing five consecutive days of recent data. The analysis includes gathering and correlating the data to provide evidence of compromised systems, network problems, or conclusions to specific alerts. Various aspects of the data are discussed in terms of scans, alerts, and Out of Spec (OOS) data. This includes a description of prioritized detects, lists of top talkers, researched information on external source addresses, correlations with outside references, critical activity, defensive recommendations, and a description of the analysis process.

© SANS Institute 2000 - 2002 Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

TABLE OF CONTENTS

Page

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

LIST OF TABLES

LIST OF FIGURES

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

1. ASSIGNMENT 1 – DESCRIBE THE STATE OF INTRUSION DETECTION

IDS TESTING

1.1 INTRODUCTION

Today's networks are under constant attack from threats ranging from the basic computer user experimenting with hacks easily available from the Internet to the technically elite hacker trying to exploit security holes for their own benefit. Protecting against these threats involves using various and complex technologies to detect attacks and protect the network and its assets from these attacks. To defend against the well-publicized threat of Internet based attacks against computer networks, many organizations today have deployed Intrusion Detection Systems (IDS). Intrusion detection systems are dedicated components of the host and network that constantly monitor for attacks. These systems provide highly sensitive monitoring capabilities and are able to alert a central management station when they see attacks. When an attack is detected the IDS can initiate various responses based on where the IDS is located. A network IDS can typically reset a network connection or reconfigure a firewall thus preventing continued attack. A host based IDS can take a more active role by preventing the attack from ever reaching the target device.

Intrusion detection systems offer great benefits. They act as burglar alarm systems for computer networks, warning and defending against attacks. But as with many advanced and specialized technologies, intrusion detection systems can sometimes be complex to configure and test. Since IDSs are highly sensitive, they have a tendency to issue a large number of false positives, which are normal and harmless network traffic that can be mistaken for attacks that detract from the systems effectiveness. With the constant release of new threats and attacks it is critical to confirm that the intrusion detection system is protecting against those threats.

A recent Information Security Magazine survey states that:

> 85% of companies surveyed use Intrusion Detection Systems
> 95% of companies surveyed state that the IDS is important
> Only 20% of companies surveyed are confident that their IDS is protecting their mission critical systems.
> 75% of companies surveyed want to increase attack analysis and reduce false positives
>
> www.infosecuritymag.com/articles/august01/cover.shtml

Currently there is no easy way of validating that an IDS is properly detecting and defending against attacks. This document describes some popular testing methodologies used to test intrusion detection systems. The goals of IDS testing are the following:

- Ensuring that the IDS is configured correctly and detecting and alerting properly.
- Determine the load that the IDS can handle and still properly detect attacks.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

- Check to make sure that the IDS vendor is keeping up with the latest threats.
- Ensuring that any custom signatures are detecting and alerting properly.

IDS systems should be tested on a regular basis once deployed, especially in the following circumstances:

- The discovery of new threats.
- After applying security updates.
- After applying new or updated signatures.
- After applying any user defined or customized rules.
- If the system has been rebooted or taken off line for any reason.

It is also a good security practice to perform IDS "fire drills" to periodically test the IDS, firewalls, and virus software.

1.2 TESTING METHODOLOGY AND TOOLS

Intrusion Detection System testing consists of pre-deployment performance testing and post-deployment assessment testing. The pre-deployment tests focus on attack detection, reporting capabilities, stress testing, and packet reassembly. The goal is to measure how accurately the system detected attacks, how well it conveyed such activity to the user, how much it can handle before it fails, and how well it handles fragmentation. Post deployment tests focus on new attack detection, new signature accuracy, and auditing. The goal is to ensure that new attacks and new signatures are being reported accurately and to ensure that the system is still performing to standard.

In order to perform IDS testing, an assortment of attack tools, utilities, and traffic generation devices, are needed. There are a vast assortment of exploit scripts and reconnaissance tools available in the security community. For more information on these tools, see Packet Storm Internet Security Solutions website at http://packetstorm.securify.org. Testing should combine scanning and exploiting techniques in series just as a real intruder would. Extreme caution should be used in the choice and implementation of exploit tools, especially in the post-deployment phase. These tools contain malicious code that could affect production network traffic and services. Post-deployment assessment testing should be performed on an isolated network segment used solely for this purpose.

One general note to make is that IDS testing is not the same thing as vulnerability testing. Generally vulnerability scanners do not launch actual attacks, instead they come to a conclusion if a system is vulnerable by looking at a number of other things, such as the operating system type, the current patches applied, etc. This generally leads to a high number of incorrect findings and misreportings, and does not truly test the ability of the IDS to detect actual attacks.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

1.2.1 Pre-Deployment Performance Testing

Performance is critical because of the high amount of bandwidth that most sites sustain. The IDS shouldn't miss any potential events because of the performance limitations of the IDS infrastructure. The definition of IDS performance is the ability for an IDS infrastructure to consistently detect X number of attacks within a given bandwidth utilization. Most IDSs look at each packet and determine whether it is part of an attack. IDS software can take many approaches to accomplish this. The issue with performance is that with high levels of bandwidth, the IDS should continually and consistently look at every IP packet and respond accordingly.

It is important to distinguish between background traffic and attack traffic in performance testing. A combination of background traffic, with attack traffic injected at certain intervals is needed. The largest challenges have been with generating legitimate background traffic, not attack traffic. After finding breaking points, it will probably only take 5-10 attacks to determine whether the IDS is failing of not.

Good traffic generators can operate above Layer 3. They can also perform basic things like the TCP 3-way handshake, TCP Initial Sequence Numbers (ISN) other than 1, and adjustable session lengths. This is important when trying to generate real world background traffic. There is a big difference between real HTTP-compliant web traffic versus a few constructed packets with port numbers set at 80 and fillers in the payloads. CAW Networks web load-testing software and NetIQ's Chariot are good traffic generators. The more accurate the background traffic (i.e. SMTP, FTP, HTTP, IMAP, SSH, etc,) the more legitimate the tests are going to be and the more accurate the true failing point of an IDS will be.

Legitimate background traffic is also important due to the fact that not all IDSs are designed the same. Most IDSs are based on either a general "packet grep" model (SNORT, Dragon, etc.) or a more protocol-aware approach (SNP, BlackICE, etc.) Many of the packets greppers are also building in protocol preprocessors. An IDS that is checking protocol compliance and one that is just packet grepping are going to be affected differently by different types of traffic.

Another point to consider in performance testing is to use real exploit code when testing the IDS. Most scanners rely on banner checking and other characteristics that don't look the same as exploit code to the IDS. When possible, use real exploit code against actual vulnerable machines. Once again, remember that these exploits contain malicious code and should be handled with caution.

One final point in performance testing is to conduct substantial testing of the systems ability to reassemble fragmented packets. The problem with fragmentation is an extremely important issue. By simply fragmenting the packets that carry attacks, many intrusion detection systems can be evaded. Also test the reassembly at higher speeds to measure fragment reassembly thresholds.

1.2.1.1 Testing Environment

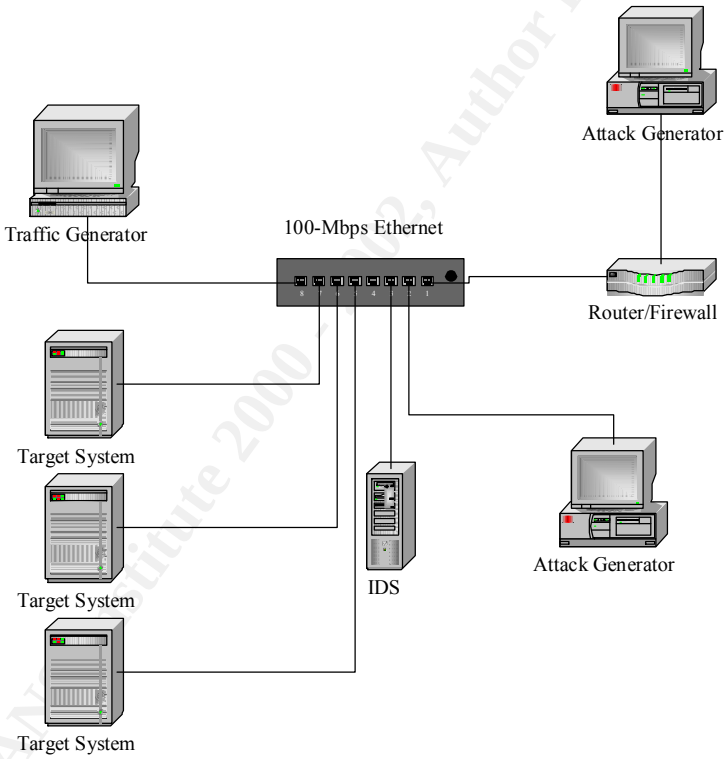
Intrusion detection systems should be performance tested before being deployed to ensure that they function properly and can handle the traffic load of the particular network in which they are

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

being deployed. This testing occurs in a closed and controlled lab environment. There are many ways to configure a testing environment, but the basic elements consist of an attack generator, target systems, a traffic generator, and the IDS to be tested, as depicted in Figure 1-1.

Traffic should be generated that is similar to the actual traffic that will be seen on the network. The network traffic generator should be configured to generate a mix of traffic comparable to the networks general level of activity. This could be a combination of things such as HTTP, Telnet, ICMP, etc. The best way to do this is to take a sample of the network traffic and configure the traffic generator accordingly. The IDS signature rulebase used should also be the actual rulebase to be used in production. Dedicated machines need to be assigned as attack generators or target devices.

Figure 1-1 Pre-Deployment IDS Testing Lab



A good testing architecture is to place the target servers on a 100-Mbps shared Ethernet segment with the IDS and a traffic-generating machine such as FileMetric, Chariot, or Webload. An assortment of BSD, IBM AIX, Linux, Windows 9x/NT/2000, and Solaris can be used as target servers. Attack from within the same segment or another segment outside a router or firewall.

1.2.1.2 Testing Scenarios

The pre-deployment testing scenarios consist of baseline testing, load testing, and fragmentation testing. The objective is to launch attacks against the target systems while injecting increasing

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

amounts of network traffic. At some point the IDS will no longer be able to effectively and consistently detect the attacks. That limit, measured in megabits per second, is what is important.

Step 1

Baseline testing involves launching the attacks at the target systems with no additional network traffic. Repeat this at least three times to get a baseline to ensure that the IDS will catch these attacks. This baseline testing will ensure that the intrusion detection system detects all of the attacks that it claims.

Step 2

Load testing introduces varying levels of traffic using a traffic generator such as Chariot. Increase the network traffic in 3M to 5Mbit/sec increments and repeat the test, launching the attacks at least three times at each traffic level. This will determine the load when frames begin to drop and the load when signatures begin to fail. Do this until the IDS is no longer consistently responding to the attacks.

Step 3

Fragmentation testing uses fragrouter to validate the IDSs ability to reassemble packets. Using this with the traffic generator can determine the load when fragmentation assembly begins to fail. Repeat this test at least three times at each level during fragmentation.

1.2.2 Post-deployment Assessment Testing

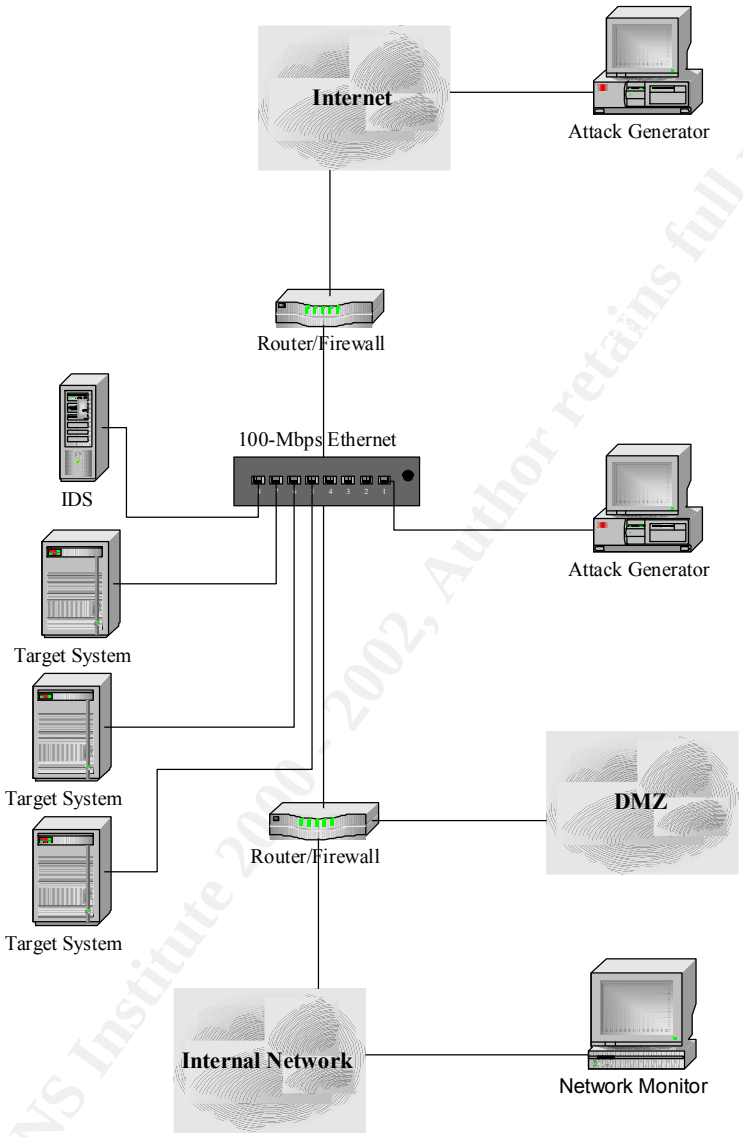
Assessment testing is important to ensure that a production IDS is performing as it should as changes occur. Once deployed, IDSs should be testing on a regular basis such as weekly or monthly. Assessment testing should also be performed to test new signatures and responses to new attacks. A lot of organizations have a habit of installing an IDS and forgetting about it. Maintaining an IDS is a daily job. This includes reviewing logs, keeping up to date on new attacks, updating signatures, and testing the system. It is also a good security practice to perform IDS "fire drills" to periodically test the IDS, firewalls, and virus software.

1.2.2.1 Testing Environment

The testing environment for post-deployment assessment testing is the current production environment. This needs to be kept in mind when generating attacks and exploits. These tools contain malicious code that could affect production network traffic and services. Post-deployment assessment testing should be performed on and isolated network segment, that contains the target machines, used solely for this purpose. Traffic generators will not likely be needed due to the presence of real life daily traffic. The intention of post-deployment assessment testing is not to "break" the IDS, but to ensure that is working correctly. When performing post-deployment IDS testing the network should be closely monitored for performance problems. If the network begins to degrade or lose services the testing should be stopped immediately. The basic testing environment for post-deployment testing includes the attack generator, the target machines on an isolated network, and the IDS to be tested, as depicted in Figure 1-2.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Figure 1-2 Post-Deployment IDS Testing



© SANS Institute 2000 - 2002, Author retains full rights.

The attack tools used in post-deployment assessment testing are important. As stated earlier many exploit scripts and reconnaissance tools available in the security community contain malicious cold that can bring down the network or its services. While these tools can still be used, there is another option by Blade Software known as IDS Informer. IDS Informer is a pure intrusion detection testing solution that utilizes the unique Simulated Attacks For Evaluation (S.A.F.E.) process to launch real but harmless attacks at IDS systems. IDS Informer contains over 320 of the latest attack and is updated on a regular basis to ensure that the latest attacks are available.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

1.2.2.2 Testing Scenarios

The post-deployment testing scenario tests that the IDS is working as anticipated, is monitoring the network correctly, and is picking up the latest attacks and responding in the correct way.

Step 1

Launch a predefined set of attacks from an Attack Group to determine that they IDS is detecting and alerting properly.

Step 2

Launch any new attacks to check new signatures that have been added. Ensure that the IDS detects and alerts on these new attacks properly.

Step 3

IDS Informer provides an Attack Log to fully log all attacks that are run. This log should be printed or saved to a text file for comparison to the IDS logs to aid in validation that the IDS detected the attacks. Any attacks that were not picked up by the IDS should be investigated further.

1.2.3 Testing Tools

The following list of tools can be used for IDS testing:

Chaff: A Perl script to test basic vulnerabilities in FTP and Web servers. It was designed to set off IDS alarms and can be used extensively in fragmentation testing.

Fragrouter: A program that fragments packets to evade IDSs. This is good for testing some of the anti-evasion components of the IDS.

Hailstorm:

IDS Informer: An intrusion detection testing solution that utilizes the unique S.A.F.E. process to launch real but harmless attacks at IDSs.

Nemesis: A program that performs generating and spoofing of various packets.

Nessus: A program that will trigger scanning alarms.

Nmap: A program that is an advanced port scanner. Nmap provides slow scanning in order to attempt evading detection.

Sneeze: A program to test Snort alarm and logging capability.

Snot: A program to test IDS robustness, as well as alarm and logging capability.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Sscan: An older, very chatty script that probes and reports on a wide range of system services and version numbers. It also performs some CGI querying.

Stick: A program that tests IDS robustness, as well as alarm and logging capability.

Tcpreplay: A program that replays real traffic in which to hide attacks.

Wacky-scan: A Perl script that mutates Nmap's scanning behavior. Most port scanning is performed vertically, that is, one host is scanned for X ports, the next is scanned for X ports and so on. Wacky-scan performs horizontal scanning and introduces a greater amount of delay between reconnaissance packets.

Whisker: A CGI vulnerability scanner that includes some anti-IDS checking abilities. Whisker looks for a defined set of CGI programs known to have security flaws. It has been known to sneak past IDSs.

Exploits and DoS attacks: LAND, Teardrop, BIND exploits, WU-FTPD exploits, IMAP/POP exploits, etc.

1.3 CONCLUSION

One of the biggest advances in information security has been the development and implementation of intrusion detection systems. IDSs are constantly monitoring the network for attacks and alerting and responding accordingly. However, rapid changing technology and new hacking techniques make it necessary to continually update and test the IDS.

An IDS should be tested in the pre-deployment phase to validate vendor claims on attack detection and to ensure that the product will perform as needed during normal network conditions. Performance testing will baseline the system, load test the system, and test packet reassembly. The tests should be run with the ruleset that will be used in production and with traffic similar to the networks general level of activity. The results should conclude whether the IDS will function well in the production environment.

Once an IDS is deployed it should have an assessment test performed on a regular basis. This can be a "fire-drill" exercise to validate signatures, test user-defined actions, prove connectivity and monitoring capabilities, and to test effectiveness against the latest attacks.

With the constant release of new threats and attacks it is critical to confirm that the intrusion detection system is protecting against those threats, which were the cause for the installation of the IDS in the first place. IDS testing allows assurance that the system is working as desired and the assurance that security funds were well warranted.

1.4 REFERENCES

Blade Software. http://www.blade-software.com.

The Ideahampster Organization. Tools – Intrusion Detection System (IDS) Testing. January 29, 2002. http://www.ideahampster.org/tools/ids.shtml.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

SC Info Security Magazine. Product Review: Blade IDS Informer. March 2002.

Shipley, Greg. Network Computing. Intrusion Detection, Take Two. November 15, 1999. http://www.networkcomputing.com/1023/1023f19.html.

Shipley, Greg. Re: Generating Traffic to Stress Test IDS. SecurityFocus Online. January 24, 2002. http://online.securityfocus.com/archive/96/252328.

Thurman, Mathias. Computerworld. Testing Intrusion-detection systems. May 22, 2001. http://www.linuxworld.com.au/article.php3?aid=166&tid=4.

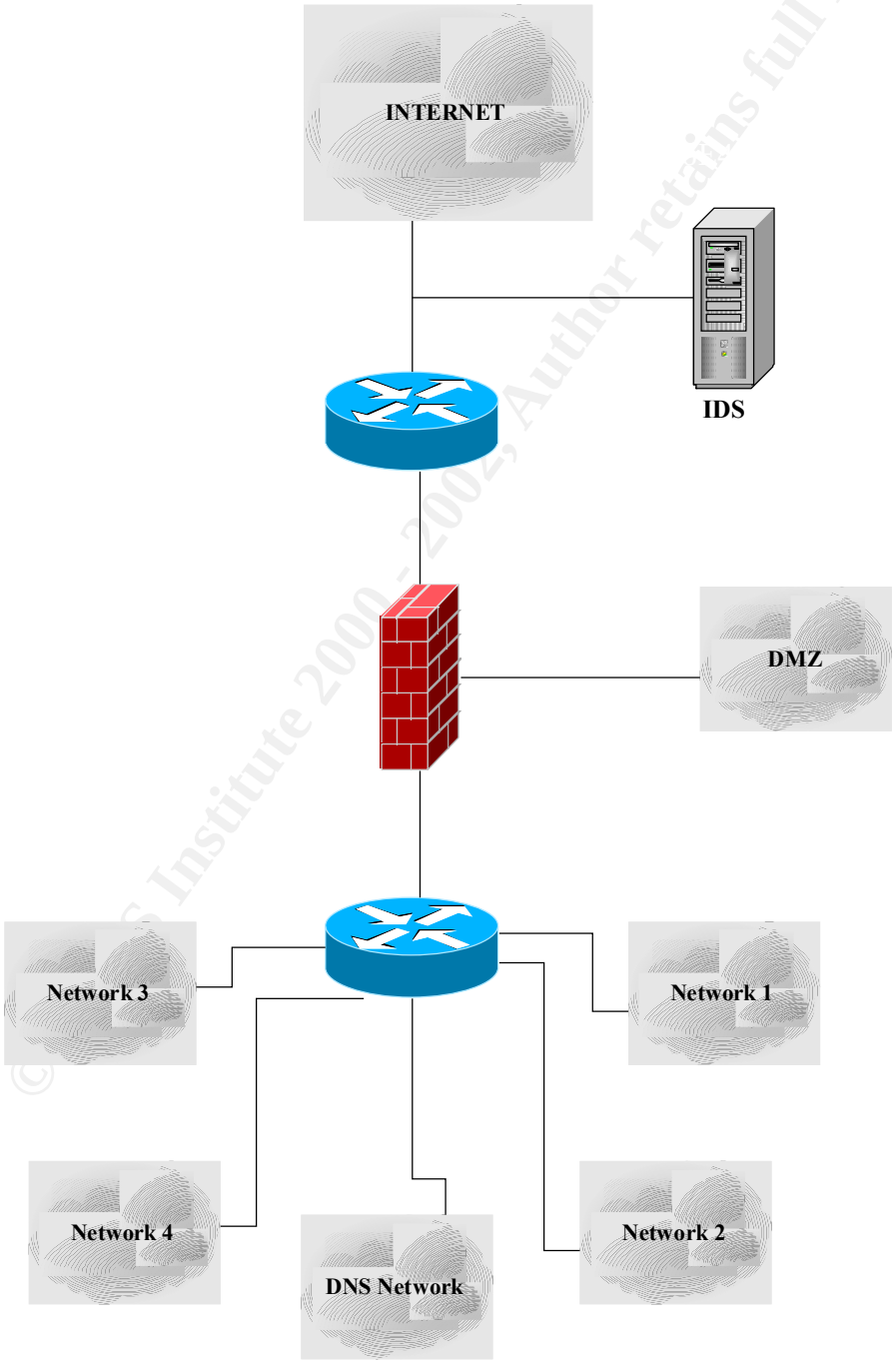
© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

2. ASSIGNMENT 2 – NETWORK DETECTS

Figure 2-1 represents the architecture of the network where all of the detects for this portion of the practical were made:

Figure 2-1 Network Architecture Diagram



© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Table 2-1 represents a general Snort Alert Log entry and the meaning of its contents:

```
[**] SCAN-SYN FIN [**]
10/25-20:33:38.568567 63.78.46.199:21 -> my.net.109:21
TCP TTL:26 TOS:0x0 ID:39426
**SF**** Seq: 0x76F7894 Ack: 0x59E55EAE Win: 0x404
```

Meaning	Snort Information
Snort signature	[**] SCAN-SYN FIN [**]
Date/Time group	10/25-20:33:38.568567
Source Address and port (21)	63.78.46.199:21
Direction operator	->
Destination address and port (21)	my.net.109:21
Protocol and Time to Live (TTL)	TCP TTL:26
Type of Service (TOS)	TOS:0x0
Packet ID in binary	ID:39426
TCP flags set	**SF****
Sequence # in Hex	Seq: 0x76F7894
Acknowledgement # in Hex	Ack: 0x59E55EAE
Windows size in Hex	Win: 0x404

Table 2-1 - Snort Alert Log File

*Table provided by SANS "Introduction to Log File Analysis" 2001.

2.1 DETECT 1 – CODERED V2 SCAN

```
[**] [1:1256:2] WEB-IIS CodeRed v2 root.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
03/14-04:21:03.329961 12.230.129.141:4635 -> MY.NET.101.230:80
TCP TTL:109 TOS:0x0 ID:9043 IpLen:20 DgmLen:112 DF
***AP*** Seq: 0xD6CE72DB  Ack: 0xC5FFACB7  Win: 0x4470  TcpLen: 20

[**] [1:1002:2] WEB-IIS cmd.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
03/14-04:21:05.837076 12.230.129.141:4754 -> MY.NET.101.230:80
TCP TTL:109 TOS:0x0 ID:9439 IpLen:20 DgmLen:120 DF
***AP*** Seq: 0xD7306505  Ack: 0xC6097006  Win: 0x4470  TcpLen: 20
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

1. **Source of Trace:**

   This trace was detected on the network depicted in Figure 2-1.

2. **Detect was generated by:**

   This detect was generated by Snort 1.8-WIN32 (Build 88).

   The signatures that generated these alerts are:

   alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB-IIS CodeRed v2 root.exe access"; flags: A+; uricontent:"scripts/root.exe?"; nocase; classtype:web-application-attack; sid: 1256; rev:2;)

   alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB-IIS cmd.exe access"; flags: A+; content:"cmd.exe"; nocase; classtype:web-application-attack; sid:1002; rev:2;)

3. **Probability the source address was spoofed:**

   The probability that the source address was spoofed is low due to the fact that the packet that caused this alert is normally part of an established TCP session.

   An American Registry for Internet Numbers (ARIN) lookup on the source IP address returns the following:

   AT&T ITS (NET-ATT)
   200 Laurel Avenue South
   Middletown, NJ 07748
   US
   Netname: ATT
   Netblock: 12.0.0.0 - 12.255.255.255
   Maintainer: ATTW
   Coordinator:
   Kostick, Deirdre (DK71-ARIN) help@IP.ATT.NET
   (888)613-6330
   Domain System inverse mapping provided by:
   DBRU.BR.NS.ELS-GMS.ATT.NET 199.191.128.106
   DMTU.MT.NS.ELS-GMS.ATT.NET 12.127.16.70
   CBRU.BR.NS.ELS-GMS.ATT.NET 199.191.128.105
   CMTU.MT.NS.ELS-GMS.ATT.NET 12.127.16.69
   Record last updated on 06-Nov-2000.
   Database last updated on 17-Mar-2002 19:57:32 EDT.

4. **Description of attack:**

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

The CodeRed worm attempts to connect to TCP port 80 on a randomly chosen host assuming that a web server will be found. Upon a successful connection to port 80, the attacking host sends a crafted HTTP GET request to the target, attempting to exploit a buffer overflow in the Indexing Service described in CERT advisory CA-2001-13. The CodeRed worm is self-replicating malicious code that exploits a known vulnerability in Microsoft IIS servers. The CA-2001-13 vulnerability has been assigned the identifier CVE-2001-0500 by the Common Vulnerabilities and Exposures (CVE) group.

5. Attack mechanism:

Once the connection is made on port 80, the HTTP GET request is sent to attempt the buffer overflow exploit. Results depend on the type of host that received the request:

- Microsoft IIS 4.0 and 5.0 servers with Indexing Service installed will almost certainly be compromised by the CodeRed worm.
- Unpatched Cisco 600-series DSL routers will process the HTTP request thereby triggering an unrelated vulnerability, which causes the router to stop forwarding packets. http://www.cisco.com/warp/public/707/cisco-code-red-worm-pub.shtml
- Systems not running Microsoft IIS, but with an HTTP server listening on TCP port 80 will probably accept the HTTP request, return an "HTTP 400 Bad Request" message, and potentially log this request in an access log.

If the exploit is successful, the worm begins executing on the target host. In the earlier variant of the worm, target hosts with a default language of English experienced the following defacement on all pages requested from the server:

HELLO! Welcome to http://www.worm.com! Hacked By Chinese!

Servers configured with a language that is not English and those infected with the later variant will not experience any change in the server content.

Other worm activity on a compromised machine is time sensitive; different activity occurs based on the day of the month of the system clock.

- Day 1 - 19: The infected host will attempt to connect to TCP port 80 of randomly chosen IP addresses in order to further propagate the worm.
- Day 20 - 27: A packet-flooding denial of service attack will be launched against a particular fixed IP address
- Day 28 - End of the month: The worm "sleeps". No active connections or denial of service

The most damaging property of this new worm is that it creates a back door on an infected server, leaving the system wide open to any attacker.

The worm copies %windir%\CMD.EXE to the following locations:

c:\inetpub\scripts\root.exe

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

c:\progra~1\common~1\system\MSADC\root.exe
d:\inetpub\scripts\root.exe
d:\progra~1\common~1\system\MSADC\root.exe

Given that the \scripts and \MSADC virtual folders have execute permission by default, moving a copy CMD.EXE to these externally accessible locations provides a means for a remote attacker to execute arbitrary commands on the compromised server. Microsoft IIS will pass commands to root.exe for execution when the server is presented with a request such as (where ARBITRARY_COMMAND is any command):

http://IpAddress/c/inetpub/scripts/root.exe?/c+ARBITRARY_COMMAND.

6. **Correlations:**

The following resources were used to gather information and correlate findings on the CodeRed worm:

http://www.snort.org/snort-db/sid.html?id=1256
http://www.cert.org/advisories/CA-2001-19.html
http://www.cert.org/advisories/CA-2001-13.html
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2001-0500
http://www.incidents.org/react/code_redII.php

7. **Evidence of active targeting:**

There is no evidence of active targeting since the CodeRed worm is self-replicating and chooses its targets randomly.

8. **Severity:**

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

Criticality: 3 (Windows server)

Lethality: 4 (could change web information and be used to scan other hosts)

System countermeasures: 5 (modern OS, patches applied)

Network countermeasures: 5 (restrictive firewall)

Severity = (3 + 4) – (5 + 5) = **-3**

9. **Defensive recommendation:**

The target machine had all patches up to date to defend against this attack.

For Windows NT 4.0:

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

http://www.microsoft.com/Downloads/Release.asp?ReleaseID=30833

For Windows 2000 Professional, Server, and Advanced Server:

http://www.microsoft.com/Downloads/Release.asp?ReleaseID=30800

10. **Multiple choice test question:**

The CodeRed worm creates a backdoor by copying what file to root.exe?

a. Nbstat.exe
b. Find.exe
c. Net.exe
d. Cmd.exe

Answer: D

2.2 DETECT 2 – DNS NAMED VERSION PROBE

[**] [1:257:1] DNS named version attempt [**]
[Classification: Attempted Information Leak] [Priority: 2]
03/18-20:04:34.824077 203.155.236.45:4991 -> MY.NET.101.214:53
UDP TTL:47 TOS:0x0 ID:1806 IpLen:20 DgmLen:58
Len: 38
[Xref => http://www.whitehats.com/info/IDS278]

1. **Source of Trace:**

This trace was detected on the network depicted in Figure 2-1.

2. **Detect was generated by:**

This detect was generated by Snort 1.8-WIN32 (Build 88).

The signature that generated this alert is:

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

3. **Probability the source address was spoofed:**

Since this alert was caused by a UDP packet, the source IP address could be easily forged. However, the attacker likely wants a response to these packets, so the source IP address is probably not spoofed.

An Asia Pacific Network Information Center (APNIC) lookup on the source IP address returns the following:

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
inetnum         203.155.192.0 - 203.155.255.255
        netname         COMNET-TH
        descr           KSC Commercial Internet Co. Ltd.
        descr           2/4 Samaggi Insurance Tower 10th Fl.,
        descr           Viphavadee-Rangsit RD
        descr           Thungsonghong, Laksi
        descr           Bangkok 10210
        country         TH
        admin-c         CW246-AP, inverse
        tech-c          TO94-ORG, inverse
        remarks         service provider
        mnt-by          APNIC-HM, inverse
        mnt-lower       KSC-ADMIN, inverse
        changed         hostmaster@apnic.net 19990218
        changed         hostmaster@apnic.net 20011016
        source          APNIC
        person          Craig White, inverse
        address         KSC Commercial Internet Co.,Ltd.
        address         2/4 Samaggi Insurance Tower 10th Fl., Viphavadee-Rangsit Rd.,
        address         Thungsonghong, Laksi
        address         Bangkok 10210
        country         TH
        phone           +66-2-9797777 ext. 7071
        e-mail          cwhite@ksc.net, inverse
        nic-hdl         CW246-AP, inverse
        mnt-by          KSC-ADMIN, inverse
        changed         netadmin@ns.ksc.co.th 20011012
        source          APNIC
        person          Technical Operation Center, inverse
        address         KSC Commercial Internet Co.,Ltd.
        address         Operation Department
        address         2/4 Samaggi Insurance Tower 10th Fl., Viphavadee-Rangsit Rd.,
        address         Thungsonghong, Laksi
        address         Bangkok 10210
        country         TH
        phone           +66-2-9797777 ext. 8428
        e-mail          netadmin@ns.ksc.co.th, inverse
        nic-hdl         TO94-ORG, inverse
        mnt-by          KSC-ADMIN, inverse
        changed         admin@ns.ksc.co.th 20011012
        source          APNIC
```

4. **Description of attack:**

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

This trace shows a UDP packet that is destined for port 53, which is DNS. This alert indicates that a remote user has attempted to determine the version of BIND running on a name server. This probe has been assigned the identifier CVE-1999-0009 by the Common Vulnerabilities and Exposures (CVE) group

5. **Attack mechanism:**

This is often a pre-attack probe used to locate vulnerable servers running the named service. There are several known vulnerabilities in BIND that the attacker may target once the version information is known. The following packet trace shows a BIND version query:

```
05/16-11:26:03.812022 attacker:1254 -> target:53
UDP TTL:64 TOS:0x0 ID:15222
Len: 38
AC CB 01 80 00 01 00 00 00 00 00 00 07 76 65 72  ..............ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03        sion.bind.....
```

The signature to detect this alert looks at the contents of byte 12 for "|07|version" or "|04|bind".

6. **Correlations:**

The following resources were used to gather information and correlate findings on BIND vulnerabilities and exploits:

http://activeworx.com/info/IDS278/
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0009
http://online.securityfocus.com/bid/134
http://www.cert.org/advisories/CA-98.05.bind_problems.html
http://www.isc.org/products/BIND/

7. **Evidence of active targeting:**

There is evidence of active targeting of the DNS service on this machine. The attacker wanted to determine the version of BIND running on this name server.

8. **Severity:**

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

Criticality: 5 (DNS)

Lethality: 5 (could disrupt name services or gain root access)

System countermeasures: 5 (modern OS, patches applied)

Network countermeasures: 5 (restrictive firewall)

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Severity = (5 + 5) – (5 + 5) = **0**

9. **Defensive recommendation:**

The server is running the latest version of BIND and the latest patches are applied.

10. **Multiple choice test question:**

What is the significance of the 12-byte offset in the following trace?

```
05/16-11:26:03.812022 attacker:1254 -> target:53
UDP TTL:64 TOS:0x0 ID:15222
Len: 38
AC CB 01 80 00 01 00 00 00 00 00 07 76 65 72  ............ver
73 69 6F 6E 04 62 69 6E 64 00 00 10 00 03     sion.bind.....
```

a. |07|version
b. |07|bind
c. |00|version
d. |00|bind

Answer: A

2.3 DETECT 3 – BAD TRAFFIC IP RESERVED BIT SET

```
[**] [1:523:3] BAD TRAFFIC ip reserved bit set [**]
[Classification: Misc activity] [Priority: 3]
03/19-00:27:14.808812 192.168.0.100 -> MY.NET.101.226
TCP TTL:234 TOS:0x0 ID:0 IpLen:20 DgmLen:40 RB
Frag Offset: 0x864   Frag Size: 0x14
```

1. **Source of Trace:**

This trace was detected on the network depicted in Figure 2-1.

2. **Detect was generated by:**

This detect was generated by Snort 1.8-WIN32 (Build 88).

The signature that generated this alert is:

```
alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"BAD TRAFFIC ip
reserved bit set"; fragbits:R; sid:523;  classtype:misc-activity; rev:3;)
```

3. **Probability the source address was spoofed:**

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

It is very probable that the source address was spoofed because 192.168.0.0 – 192.168.255.255 are reserved nonroutable addresses. Because of concerns over the growing shortage of IP addresses, there is a special set of IP addresses that have been set aside by the Internet Assigned Numbers Authority (IANA) for private networks. These addresses will not be assigned to any system connected to the Internet.

4. **Description of attack:**

This detect is a TCP packet with both of the Reserved flags set. The packet was also fragmented and generated from a spoofed IP address. This is unexpected traffic not usually seen through normal activity.

5. **Attack mechanism:**

The reserved bits are unused and should be set to 0 (False). Reconnaissance of host information may involve setting these bits to 1 (True) in order to see how target hosts react. This may give clues to identifying the Operating System and Version of target systems. Different IP stacks will respond to traffic in different ways allowing the intruder to use this information to identify hosts on the network. Figure 2-3 shows the format of a TCP header. Notice the reserved 6-bit portion highlighted in yellow.

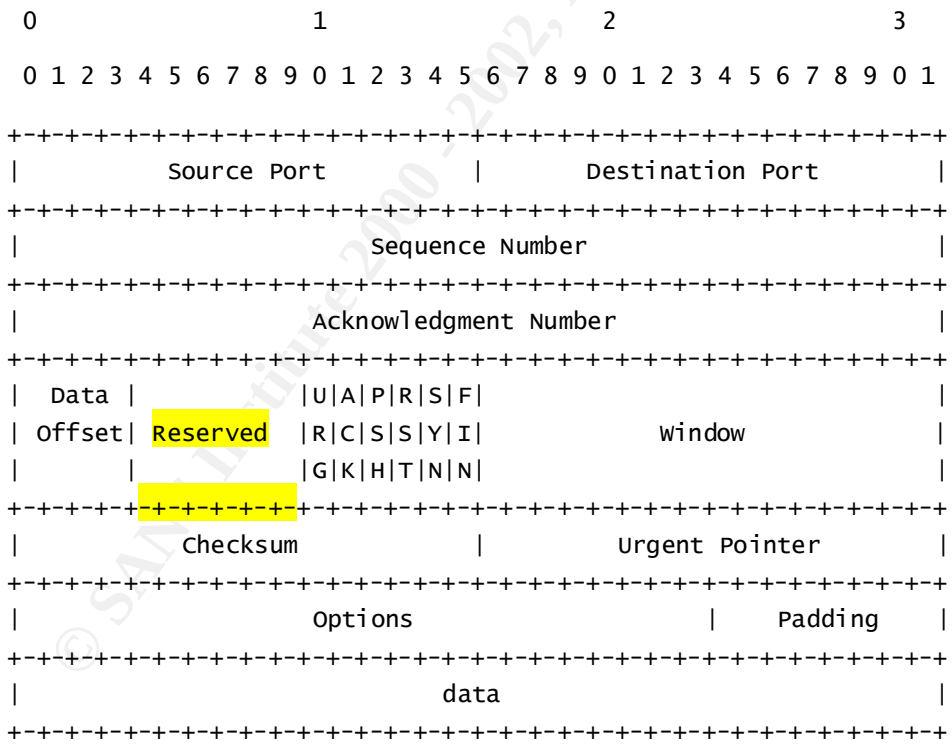


Figure 2-2 TCP Header Format

Note that one tick mark represents one bit position.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

The intruder needs to be able to craft and insert the bad packets onto the network. Several public tools exist to create packets matching this profile.

This traffic should not appear in normal traffic, making the likeliness of false positives very low. Non-RFC IP stacks may generate false positives.

6. **Correlations:**

The following resources were used to gather information and correlate findings on TCP packet vulnerabilities and exploits:

http://www.snort.org/snort-db/sid.html?id=523
http://www.freesoft.org/CIE/Course/Section4/8.htm

7. **Evidence of active targeting:**

It is probable that this detect is evidence of active targeting. It did not target any other machines on my network as a general scan would.

8. **Severity:**

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

Criticality: 2 (Windows host)

Lethality: 2 (could identify operating system to look for vulnerabilities and exploits)

System countermeasures: 5 (modern OS, patches applied)

Network countermeasures: 5 (restrictive firewall)

Severity = (2 + 2) – (5 + 5) = **-6**

9. **Defensive recommendation:**

Drop bad traffic of this type at the border routers.

10. **Multiple choice test question:**

Setting the reserved bits in the TCP header of a packet could allow the attacker to:

a. Gain root level access on the target host.
b. Cause the target host to reboot.
c. Exploit a well-known CGI vulnerability.
d. Identify the operating system and version of the target host.

Answer: D

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

2.4 DETECT 4 – SHELLCODE X86 NOOP

[**] [1:648:4] SHELLCODE x86 NOOP [**]
[Classification: Executable code was detected] [Priority: 1]
03/17-18:01:40.053281 128.11.183.67:7115 -> MY.NET.101.17:13830
TCP TTL:47 TOS:0x10 ID:32368 IpLen:20 DgmLen:1500 DF
A* Seq: 0xAE6D9F05 Ack: 0xAE20C858 Win: 0x60F4 TcpLen: 20
[Xref => http://www.whitehats.com/info/IDS181]

1. **Source of Trace:**

   This trace was detected on the network depicted in Figure 2-1.

2. **Detect was generated by:**

   This detect was generated by Snort 1.8-WIN32 (Build 88).

   The signature that generated this alert is:

   alert ip $EXTERNAL_NET any -> $HOME_NET any (msg:"SHELLCODE x86 NOOP"; content: "|90 90 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:4;)

3. **Probability the source address was spoofed:**

   The probability that the source address was spoofed is low due to the fact that the packet that caused this alert is normally part of an established TCP session.

   An ARIN lookup on the source IP address returns the following:

   GENUITY (NET-GNTY-128-11)
   3 Van de Graaff Dr.
   Burlington, MA 01803
   US
   Netname: GNTY-128-11
   Netblock: 128.11.0.0 - 128.11.255.255
   Maintainer: GNTY
   Coordinator:
   Soulia, Cindy (CS15-ARIN) csoulia@genuity.net
   800-632-7638
   Domain System inverse mapping provided by:
   NS1.GENUITY.NET 207.240.5.60
   NS2.GENUITY.NET 207.240.5.61
   NIC.NEAR.NET 192.52.71.4
   NS1.BARRNET.NET 131.119.245.5
   Record last updated on 24-Sep-2001.
   Database last updated on 17-Mar-2002 19:57:32 EDT.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

4. Description of attack:

This alert indicates that a string of the character 0x90 was detected. It indicates the presence of the NOP operation in x86 machine code. Many remote buffer overflow exploits send a series of NOP (no-operation) bytes to pad their chances of successful exploitation.

This alert is specific to a vulnerability that may have been caused by any of several possible exploits. Signatures used to detect this event are specific and consider the packet payload, in this case the following:

Contents: "|90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90|"

Since the packet offset is zero, begin looking for this content string in the start of the packet data.

5. Attack mechanism:

The attack works by completing the three-way handshake, then sending data, which is padded with a large number of NOP bytes, and attack data. The buffer size is exceeded, and the attacker's information is then processed

There are reported incidents where legitimate traffic may cause an intrusion detection system to raise "false positive" alerts for this event. Since all network traffic is watched, it is possible this sequence may occur in any legitimate binary file transmission, such as GIFs and JPGs. This can be confirmed by looking at the packet trace generated by this alert.

The following trace was provided by www.whitehats.com and shows NOPs used in the remote Netscape FastTrack exploit for UnixWare running on x86.

```
12/31-12:28:59.998897 source:2389 -> target:457
TCP TTL:64 TOS:0x0 ID:19342  DF
***PA* Seq: 0xBE085F0F   Ack: 0x1ABC3DF2   Win: 0x7D78
TCP Options => NOP NOP TS: 73313996 0
47 45 54 20 2F 90 90 90 90 90 90 90 90 90 90 90      GET /...........
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90      ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90      ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90      ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90      ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90      ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90      ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90      ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90      ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90      ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90      ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90      ................
90 90 EB 5F 9A FF FF FF FF 07 FF C3 5E 31 C0 89      ..._........^1..
46 9D 88 46 A2 31 C0 50 B0 8D E8 E5 FF FF FF 83      F..F.1.P........
C4 04 31 C0 50 B0 17 E8 D8 FF FF FF 83 C4 04 31      ..1.P..........1
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
C0 50 56 8B 1E F7 DB 89 F7 83 C7 10 57 89 3E 83  .PV.........W.>.
C7 08 88 47 FF 89 7E 04 83 C7 03 88 47 FF 89 7E  ...G..~.....G..~
08 01 DF 88 47 FF 89 46 0C B0 3B E8 A4 FF FF FF  ....G..F..;.....
83 C4 0C E8 A4 FF FF FF D3 FF FF FF FF FF FF FF  ................
FF FF FF FF FF FF FF FF 2F 62 69 6E 2F 73 68 FF  ......../bin/sh.
2D 63 FF 2F 75 73 72 2F 58 2F 62 69 6E 2F 78 74  -c./usr/X/bin/xt
65 72 6D 24 7B 49 46 53 7D 2D 64 69 73 70 6C 61  erm${IFS}-displa
79 24 7B 49 46 53 7D 75 6E 69 78 3A 30 2E 30 FF  y${IFS}unix:0.0.
30 61 FC BF 20 20 48 54 54 50 2F 31 2E 30 0D 0A  0a..  HTTP/1.0..
48 6F 73 74 3A 20 6C 6F 63 61 6C 68 6F 73 74 3A  Host: localhost:
34 35 37 0D 0A 41 63 63 65 70 74 3A 20 74 65 78  457..Accept: tex
74 2F 68 74 6D 6C 0D 0A 41 63 63 65 70 74 2D 45  t/html..Accept-E
6E 63 6F 64 69 6E 67 3A 20 67 7A 69 70 2C 20 63  ncoding: gzip, c
6F 6D 70 72 65 73 73 0D 0A 41 63 63 65 70 74 2D  ompress..Accept-
4C 61 6E 67 75 61 67 65 3A 20 65 6E 0D 0A 4E 65  Language: en..Ne
67 6F 74 69 61 74 65 3A 20 74 72 61 6E 73 0D 0A  gotiate: trans..
55 73 65 72 2D 41 67 65 6E 74 3A 20 78 6E 65 63  User-Agent: xnec
0D 0A 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 90  ................
90 90 90 90 90 90 90 90 90 90 90 90 90 90 90 EB  ................
5F 9A FF FF FF FF 07 FF C3 5E 31 C0 89 46 9D 88  _........^1..F..
46 A2 31 C0 50 B0 8D E8 E5 FF FF FF 83 C4 04 31  F.1.P..........1
C0 50 B0 17 E8 D8 FF FF FF 83 C4 04 31 C0 50 56  .P..........1.PV
8B 1E F7 DB 89 F7 83 C7 10 57 89 3E 83 C7 08 88  .........W.>....
47 FF 89 7E 04 83 C7 03 88 47 FF 89 7E 08 01 DF  G..~.....G..~...
88 47 FF 89 46 0C B0 3B E8 A4 FF FF FF 83 C4 0C  .G..F..;........
E8 A4 FF FF FF D3 FF FF FF FF FF FF FF FF FF FF  ................
FF FF FF FF FF 2F 62 69 6E 2F 73 68 FF 2D 63 FF  ...../bin/sh.-c.
2F 75 73 72 2F 58 2F 62 69 6E 2F 78 74 65 72 6D  /usr/X/bin/xterm
24 7B 49 46 53 7D 2D 64 69 73 70 6C 61 79 24 7B  ${IFS}-display${
49 46 53 7D 75 6E 69 78 3A 30 2E 30 FF 30 61 FC  IFS}unix:0.0.0a.
BF 0D 0A 0D 0A                                   .....
```

6. Correlations:

SANS Intrusion Detection in Depth
© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Max Vision originally detected this trace. He developed the signature based on exploit analysis and confirmed by packet trace evidence. His contact information is vision@whitehats.com.

More information on this vulnerability can be found at:

http://www.whitehats.com/info/IDS181 mirrored at
http://activeworx.com/info/IDS181/event.html

and

http://www.snort.org/snort-db/sid.html?id=648

7. **Evidence of active targeting:**

It is probable that this detect is evidence of active targeting. The source address attacked the same target address four different times. It did not target any other machines on my network as a general scan would.

8. **Severity:**

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

Criticality: 5 (DNS)

Lethality: 3 (could get user access)

System countermeasures: 5 (modern OS, patches applied)

Network countermeasures: 5 (restrictive firewall)

Severity = (5 + 3) – (5 + 5) = **-2**

9. **Defensive recommendation:**

The target machine had all patches up to date to defend against this attack. I recommend investigating the payload further to see what shell code is attempting to execute.

10. **Multiple choice test question:**

According to the following trace, what does the ***AP*** represent?

```
[**] [1:1256:2] WEB-IIS CodeRed v2 root.exe access [**]
[Classification: Web Application Attack] [Priority: 1]
03/14-04:21:03.329961 12.230.129.141:4635 -> MY.NET.101.230:80
TCP TTL:109 TOS:0x0 ID:9043 IpLen:20 DgmLen:112 DF
***AP*** Seq: 0xD6CE72DB  Ack: 0xC5FFACB7  Win: 0x4470  TcpLen: 20
```

a. The packet is requesting to begin a TCP connection.
b. The packet is sending data.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

c. The packet is aborting a connection.
d. The packet is gracefully terminating a connection.

Answer: B

2.5 DETECT 5 – BAD TRAFFIC DATA IN TCP SYN PACKET

```
[**] [1:526:3] BAD TRAFFIC data in TCP SYN packet [**]
[Classification: Misc activity] [Priority: 3]
03/18-17:22:44.079620 216.35.71.246:55486 -> MY.NET.101.17:53
TCP TTL:46 TOS:0x0 ID:1 IpLen:20 DgmLen:64
******S* Seq: 0xBD6CDE92  Ack: 0x8178F63B  Win: 0x800  TcpLen: 20

[**] [1:526:3] BAD TRAFFIC data in TCP SYN packet [**]
[Classification: Misc activity] [Priority: 3]
03/18-17:22:45.103737 216.35.71.246:55490 -> MY.NET.101.17:53
TCP TTL:46 TOS:0x0 ID:2 IpLen:20 DgmLen:64
******S* Seq: 0x5D687C70  Ack: 0x21ED60A6  Win: 0x800  TcpLen: 20
```

1. Source of Trace:

This trace was detected in the network depicted in Figure 2-1.

2. Detect was generated by:

This detect was generated by Snort 1.8-WIN32 (Build 88).

The signature that generated this alert is:

alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"BAD TRAFFIC data in TCP SYN packet"; flags:S; dsize:>6; sid:526; classtype:misc-activity; rev:3;)

3. Probability the source address was spoofed:

An ARIN lookup on the source IP address returns the following:

Exodus Commnications Inc. (NETBLK-ECI-7)
1605 Wyatt Dr. Santa Clara, CA
95054US
US
Netname: ECI-7
Netblock: 216.32.0.0 - 216.35.255.255
Maintainer: ECI
Coordinator:
Center, Network Control (NOC44-ARIN) ipaddressadmin@exodus.net
(888) 239-6387 (FAX) (888) 239-6387
Domain System inverse mapping provided by:
DNS01.EXODUS.NET 209.1.222.244
DNS02.EXODUS.NET 209.1.222.245

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

DNS03.EXODUS.NET 209.1.222.246
DNS04.EXODUS.NET 209.1.222.247
* Rwhois reassignment information for this block is available at:
* rwhois.exodus.net 4321
ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
Record last updated on 09-Mar-2000.
Database last updated on 18-Mar-2002 19:58:22 EDT.

4. **Description of attack:**

Data was sent during a request to establish a TCP connection (SYN) with a host. The target port is DNS port 53. This should not occur during normal TCP communications. Large numbers of alerts from single or distributed sources may indicate a Denial-of-Service (DoS) attack in progress, especially if source addresses are unverifiable (spoofed). The alert above was seen about 39 times over a 10-minute period.

5. **Attack mechanism:**

The alert triggers based on the detection of data content within a TCP packet with the SYN (synchronize) flag set, which is used as part of the TCP "three-way handshake" process to establish communications between two hosts.

A TCP "three-way handshake" essentially consists of the following:
A --> B TCP SYN 'my sequence number is X'
B --> A TCP ACK 'your sequence number is X'
B --> A TCP SYN 'my sequence number is Y'
A --> B TCP ACK 'your sequence number is Y'
Where A = Source Host and B = Destination Host.

Once this handshake process completes successfully, the two hosts have successfully negotiated a TCP session, and data exchange may occur. For alerts matching this signature, it means that data was detected in a SYN phase of the three-way handshake, and by definition, shouldn't be there because the two hosts haven't finished setting up a TCP session.

This type of alert can indicate a DoS attempt, using SYN Flooding, or system probing. Several public tools exist to create packets matching this profile, including automated and scripted DoS tools.

6. **Correlations:**

The following resources were used to gather information and correlate findings on TCP SYN packet vulnerabilities and exploits:

http://www.snort.org/snort-db/sid.html?id=526
http://www.cert.org/incident_notes/IN-99-07.html

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

7. **Evidence of active targeting:**

   A DoS attempt is usually the result of active targeting. The attacker wants to disable a particular host, network, or service.

8. **Severity:**

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

   Critcality: 5 (DNS)

   Lethality: 3 (could disrupt network communication and services)

   System countermeasures: 5 (modern OS, patches applied)

   Network countermeasures: 5 (restrictive firewall)

   Severity = (5 + 3) – (5 + 5) = **-2**

9. **Defensive recommendation:**

   The firewall and border router protected against this attack.

10. **Multiple choice test question:**

    According to the following trace, how do you know this attack is targeting DNS?

```
[**] [1:526:3] BAD TRAFFIC data in TCP SYN packet [**]
[Classification: Misc activity] [Priority: 3]
03/18-17:22:44.079620 216.35.71.246:55486 -> MY.NET.101.17:53
TCP TTL:46 TOS:0x0 ID:1 IpLen:20 DgmLen:64
******S* Seq: 0xBD6CDE92  Ack: 0x8178F63B  Win: 0x800  TcpLen: 20
```

a. 216.35.71.246:55486
b. MY.NET.101.17:53
c. Seq: 0xBD6CDE92
d. This cannot be determined

Answer: B

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

3. ASSIGNMENT 3 – "ANALYZE THIS" SCENARIO

3.1 OVERVIEW

This audit has been conducted at the request of SANS and is based on the Snort intrusion detection logs provided by the university. This audit is to provide a security assessment spanning over a consecutive 5-day period. The analysis identifies vulnerabilities, provides specific recommendations to improve security, and provides knowledge sharing and transfer.

The Snort intrusion detection logs include the following log types:

Alert Logs – Logs containing possible signature matches made by the Snort IDS.

Scan Logs – Logs containing network reconnaissance scans as identified by the Snort IDS.

OOS logs – Logs containing out of scope packet detail as captured by the Snort IDS.

All log files were provided in Standard output (ASCII).

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

Alert files: alert.020226.gz, alert.020227.gz, alert.020228.gz, alert.020301.gz, alert.020302.gz

Scan files: scans.020226.gz, scans.020227.gz, scans.020228.gz, scans.020301.gz, scans.020302.gz

OSS files: oos_Feb.26.2002.gz, oos_Feb.27.2002.gz, oos_Feb.28.2002.gz, oos_Mar.1.2002.gz, oos_Mar.2.2002.gz

There are two reasons that I chose the five-day span of Feb. 26 through Mar. 2. First, I wanted to analyze trend data that included data from both the workweek and the weekend. The data meets this requirement by spanning Tuesday through Saturday. Second, I wanted to analyze trend data that covered both the end of the month and the first of the month. I wanted to see if these two characteristics had any affect on the volume or type of detects.

Section 3.2 of the audit is a list of detects from the Alert logs. I give descriptions for the top 10 alerts. The top alerts contain Unix LPD, Windows IIS, NETBIOS, SNMP, reconnaissance, large UDP packets, chat, Red Worm and GCI detects. I also list the top scanning source and destination ports. It is shown that most scanning traffic relates to file sharing and chat. Next, I give an analysis of the OOS files that are reconnaissance detects. There are numerous machines sending crafted packets to mostly the KaZaA port 1214 on 7 targets.

Section 3.3 lists the top talkers. I list the top ten talkers in the Scan log and compare the results to the top ten scanners in the Alert log. The results are the same ten addresses. I also list the top ten talkers in the Alert log, excluding scanning.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Section 3.4 lists 5 external source addresses that should be investigated further for possible trojan and exploit attempts. The alerts provided indicate possible compromised systems on the internal network. Compromised systems can be used for attack purposes for DoS attacks as well as other intrusion attempts. I give detailed information on registration of these addresses for investigation.

Section 3.5 correlates my findings with websites and other student practicals. I list several other student practicals that contain the same types of detects and traffic analysis. I noticed a trend in the practicals that I read in terms of trojans, Red Worm, messaging, and file sharing activities. These types of activities were prevalent and numerous is most practicals. I also noticed that over time the number of alerts and scans have increased significantly.

Section 3.6 contains graphs with daily and hourly totals for scans and alerts. This information provides trend analysis on the rate of detects over time. Overall, detects decreased toward the end of the week, and were the lowest on Saturday. Detects seemed to follow a trend with the typical workday, increasing around 8 a.m. and decreasing at 6 p.m. This could indicate false positives occurring within normal network traffic.

Section 3.7 lists information on some internal machines that are possibly compromised by the SubSeven 2.0 trojan. Five machines look as if they have been compromised by another internal machine. All of these systems should be investigated further and the trojan removed if found.

Section 3.8 lists some defense recommendation based on the performed audit. I list recommendations concerning security policy, perimeter defense, IDS performance, host security, and security awareness.

Section 3.9 details the steps and tools used to analyze the data. I imported all of the log data to a Microsoft Access database and ran queries to analyze the data. I then used Microsoft Excel to create the tables and charts.

This audit is a brief overview of the most critical issues in these logs from a security perspective. An additional in-depth analysis is recommended over an extended period of time. The volume of data in the logs can provide useful insight into potential security and performance issues within the MY.NET network.

3.2 LIST OF DETECTS

Over the five day period there were 3,110,757 total scan log entries. There were 550,344 alert log entries excluding scan information. The scan information in the alert logs consisted of "portscan detected", "portscan status", and "end of portscan" information. This portscan information alone resulted in 977,959 entries in the alert file. The OOS files consisted of 41 entries total for the five days. Table 3-1 outlines the alerts collected from Feb. 26 – Mar. 2 2002. The percentage represents the percent of actual alerts, excluding scan information.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Table 3-1 - Alert Totals for Five Day Period

Alert Type	Count	Percent of Total
connect to 515 from inside	207517	37.71%
spp_http_decode: IIS Unicode attack detected	87301	15.86%
SMB Name Wildcard	65052	11.82%
SNMP public access	46914	8.52%
MISC Large UDP Packet	41614	7.56%
ICMP Echo Request L3retriever Ping	31925	5.80%
INFO MSN IM Chat data	19601	3.56%
High port 65535 udp - possible Red Worm - traffic	11725	2.13%
spp_http_decode: CGI Null Byte attack detected	8663	1.57%
ICMP Echo Request Nmap or HPING2	5238	0.95%
Possible trojan server activity	5016	0.91%
Watchlist 000220 IL-ISDNNET-990517	4286	0.78%
ICMP Echo Request Windows	1880	0.34%
INFO Possible IRC Access	1872	0.34%
ICMP Router Selection	1796	0.33%
ICMP Fragment Reassembly Time Exceeded	1654	0.30%
WEB-IIS view source via translate header	1647	0.30%
INFO Inbound GNUTella Connect request	1267	0.23%
FTP DoS ftpd globbing	1219	0.22%
Incomplete Packet Fragments Discarded	545	0.10%
WEB-IIS _vti_inf access	420	0.08%
WEB-FRONTPAGE _vti_rpc access	419	0.08%
Watchlist 000222 NET-NCFC	382	0.07%
Null scan!	377	0.07%
NMAP TCP ping!	247	0.04%
INFO FTP anonymous FTP	198	0.04%
INFO Outbound GNUTella Connect request	145	0.03%
SCAN Proxy attempt	126	0.02%
SCAN Synscan Portscan ID 19104	121	0.02%
WEB-MISC Attempt to execute cmd	113	0.02%
Port 55850 tcp - Possible myserver activity - ref. 010313-1	99	0.02%
ICMP traceroute	96	0.02%
INFO Inbound GNUTella Connect accept	73	0.01%
EXPLOIT x86 NOOP	72	0.01%
WEB-CGI scriptalias access	67	0.01%
FTP CWD / - possible warez site	61	0.01%
MISC traceroute	56	0.01%
High port 65535 tcp - possible Red Worm - traffic	49	0.01%
WEB-MISC compaq nsight directory traversal	45	0.01%
ICMP Destination Unreachable (Communication Administratively Prohibited)	44	0.01%
INFO Napster Client Data	31	0.01%
Attempted Sun RPC high port access	29	0.01%
EXPLOIT NTPDX buffer overflow	28	0.01%
SYN-FIN scan!	28	0.01%
INFO - Possible Squid Scan	27	0.00%
WEB-MISC http directory traversal	25	0.00%
WEB-MISC 403 Forbidden	23	0.00%

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Queso fingerprint	22	0.00%
Back Orifice	21	0.00%
ICMP Destination Unreachable (Protocol Unreachable)	19	0.00%
MYPARTY - Possible My Party infection	17	0.00%
Russia Dynamo - SANS Flash 28-jul-00	15	0.00%
ICMP Echo Request CyberKit 2.2 Windows	13	0.00%
EXPLOIT x86 setgid 0	11	0.00%
TCP SRC and DST outside network	10	0.00%
RPC tcp traffic contains bin_sh	10	0.00%
ICMP Echo Request BSDtype	9	0.00%
RFB - Possible WinVNC - 010708-1	8	0.00%
EXPLOIT x86 setuid 0	8	0.00%
SCAN FIN	7	0.00%
WEB-IIS Unauthorized IP Access Attempt	6	0.00%
BACKDOOR NetMetro Incoming Traffic	5	0.00%
Port 55850 udp - Possible myserver activity - ref. 010313-1	4	0.00%
RPC udp traffic contains bin sh	3	0.00%
WEB-CGI phf access	3	0.00%
NIMDA - Attempt to execute cmd from campus host	3	0.00%
WEB-IIS encoding access	2	0.00%
WEB-IIS File permission canonicalization	2	0.00%
SUNRPC highport access!	2	0.00%
WEB-MISC Lotus Domino directory traversal	2	0.00%
X11 outgoing	1	0.00%
WEB-MISC whisker head	1	0.00%
BACKDOOR NetMetro File List	1	0.00%
DNS named iquery attempt	1	0.00%
Tiny Fragments - Possible Hostile Activity	1	0.00%
WEB-CGI redirect access	1	0.00%
x86 NOOP - unicode BUFFER OVERFLOW ATTACK	1	0.00%
TFTP - External UDP connection to internal tftp server	1	0.00%
EXPLOIT x86 stealth noop	1	0.00%

Table 3-2 gives a description of the top ten alerts:

Table 3-2 - Alert Descriptions

connect to 515 from inside	This alert was triggered by attempts to connect to the Unix print spooler service, LPD port 515, from an internal source address. This service contains vulnerabilities that may allow an attacker to gain root access to the host.
spp_http_decode: IIS Unicode attack detected	This alert was triggered by a possible Microsoft IIS Unicode attack. A flaw exists in the handling of .asp requests. Typically when a request is made for an .asp file, IIS will identify that it is a script and run it as such. However if the host is formatted with a FAT file system and a request is made with an .asp

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

	Unicode encoded file extension, IIS may not handle the request properly and return the source code of the file.
SMB Name Wildcard	This alarm is triggered by attempts to connect to the NETBIOS Name Service, port 137. In many cases this can be normal traffic but it can also be used by an attacker to obtain a list of Windows hosts on the network.
SNMP public access	This alert is triggered when an SNMP request, port 161, is made with a password of "public" which is the default community string.
MISC Large UDP Packet	This alert was triggered by an oversized UDP packet. This could be a sign of a UDP flood. If many large UDP packets are sent to a host it can cause a DoS. Another possibility is that the UDP session is actually a covert channel used by an attacker to communicate with a compromised host.
ICMP Echo Request L3retriever Ping	This alert was triggered by a ping request that could be doing possible network reconnaissance.
INFO MSN IM Chat data	This alert was triggered by Microsoft Network Instant Messaging chat.
High port 65535 udp - possible Red Worm - traffic	This alert was triggered by a source or destination port of 65535, a Red Worm port. The Red Worm attempts to gain access to Linux systems through vulnerabilities in LPRng, rpc.statd or BIND. If successful it installs a trojan that listens on TCP port 65535. This alert triggers on TCP traffic to or from port 65535 which could indicate a potential Red Worm infection. More information can be found at http://www.cert.org/advisories/CA-2001-19.html.
spp_http_decode: CGI Null Byte attack detected	This alert was triggered by a possible CGI null byte attack. If the http decoding routine finds a %00 in an http request, it will alert with this message. Sites that use cookies with URL encoded binary data or SSL encrypted traffic may trigger false positives. More information

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

	can be found at http://www.phrack.com/phrack/55/P55-07.
ICMP Echo Request Nmap or HPING2	This alert was triggered by a ping request that could be doing possible network reconnaissance.

Table 3-3 lists the top scanning source and destination ports:

Table 3-3 - Top Scanning Source and Destination Ports

Source Port	Count	Description
7000	405647	afs3-fileserver
123	379007	Network Time Protocol (NTP)
7001	273062	afs3-callback - callbacks to cache managers
0	177337	invalid port - possible packet crafting
514	142189	syslog
137	95892	NETBIOS Name Service
778	56918	Unknown
88	53093	Kerberos
1347	38708	bbn-mmc - multi media conferencing
6970	37403	RealTime Protocol

Destination Port	Count	Description
7001	405961	afs3-callback - callbacks to cache managers
80	352985	World Wide Web HTTP
7000	234979	afs3-fileserver
53	180459	Domain Name Server
514	154020	syslog
0	139891	invalid port - possible packet crafting
137	95780	NETBIOS Name Service
1214	60597	KaZaA
1346	38725	Alta Analytics License Manager
7003	38435	afs3-vlserver - volume location database

As seen from the listed source and destination ports, much of the traffic is file sharing. This includes AFS, KaZaA, RealTime, and bbn-mmc. This is typical of a university environment and the tools made available today for peer-to-peer file sharing.

The OOS files contained 41 total alerts. Each of these packets violated accepted standards for construction of IP packets. Each alert contained packets with various combination of TCP flag bit settings, such as 2*SFRPAU, **SF*P**, **SFRP**, etc. Packets with these types of settings are often used for reconnaissance and fingerprinting. Often the response to the types of packets allows the attacker to determine the operating system of the target machine. Packets of this

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

nature can also be used as a DoS if the target operating system or application does not know how to handle them. The following tables list the source and destination address totals for the OOS packets, as well as the destination ports. There were 17 different offending machines, all with external addresses. However there were only 7 target machines. Also notice that there are 8 destination port addresses that are being targeted. The most popular being 1214, the well-known KaZaA port. This could be evidence of attackers attempting a KaZaA exploit.

Table 3-4 - OOS Alerts Source and Destination Addresses

Source Address	OSS Alert Count
217.226.76.95	15
165.121.26.190	3
216.218.255.227	3
66.32.57.247	3
63.98.19.242	2
62.201.85.15	2
12.7.27.178	2
193.226.125.42	2
66.68.209.145	1
80.135.223.208	1
68.37.65.44	1
213.66.11.166	1
212.204.48.222	1
209.240.11.125	1
68.80.152.66	1
149.225.27.203	1
24.150.5.159	1

Destination Address	OOS Alert Count
MY.NET.88.162	17
MY.NET.150.133	16
MY.NET.153.198	3
MY.NET.152.15	2
MY.NET.152.185	1
MY.NET.152.179	1
MY.NET.152.178	1

Table 3-5 - OOS Alert Destination Ports

Destination Port	OOS Alert Count
1214	25
113	5
6346	3
0	3
3	2
191	1
154	1
70	1

3.3 TOP TALKERS

Table 3-6 represents the top ten source IP addresses in the Scan log, including the count and percent of total scans. Top Talkers per Scan log:

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Table 3-6 - Top Talkers per Scan Log

Scanning Source Address	Count	Percent of Total
MY.NET.60.43	465333	14.96%
MY.NET.6.49	196603	6.32%
MY.NET.6.52	195638	6.29%
MY.NET.6.45	152707	4.91%
MY.NET.6.48	146189	4.70%
MY.NET.6.50	132242	4.25%
MY.NET.6.60	68216	2.19%
MY.NET.60.11	61917	1.99%
MY.NET.6.53	57450	1.85%
MY.NET.5.7	38345	1.23%

Table 3-7 represents the top ten source IP addresses in the Alert log that were performing scanning. I calculated this information to verify the data from the two logs. As seen, both logs contain the same top talkers. The counts and percentages will differ due to the rate and frequency of scanning and the method of logging. Top Talkers (Scanning) per Alert log:

Table 3-7 - Top Talkers (Scanning) per Alert Log

Scanning Source Address	Count	Percent of Total
MY.NET.60.43	102106	10.44%
MY.NET.6.45	37050	3.79%
MY.NET.60.11	25372	2.59%
MY.NET.6.60	18854	1.93%
MY.NET.6.49	17354	1.77%
MY.NET.6.52	17093	1.75%
MY.NET.5.7	16103	1.65%
MY.NET.6.53	15982	1.63%
MY.NET.6.50	12835	1.31%
MY.NET.6.48	12583	1.29%

Table 3-8 represents the top ten source IP addresses in the Alert log, including the count and percent of total scans. Top Talkers per Alert log:

Table 3-8 - Top Talkers per Alert Log

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Alert Source Address	Count	Percent of Total
MY.NET.153.106	18983	3.45%
MY.NET.11.7	13800	2.51%
MY.NET.11.6	12561	2.28%
MY.NET.88.240	11426	2.08%
MY.NET.153.119	9868	1.79%
MY.NET.70.177	7900	1.44%
MY.NET.150.198	7346	1.33%
MY.NET.70.177	5686	1.03%
MY.NET.11.5	4859	0.88%
202.30.244.134	4257	0.77%

3.4 EXTERNAL SOURCE ADDRESS INFORMATION

Table 3-9 lists external IP addresses that need further investigation. I chose these addresses from the Alerts logs based on activity and known exploit port numbers.

Table 3-9 - External Source Addresses

Source Address:Port	Count
209.223.8.35:27374	2998
63.250.205.43:0	915
211.233.70.162:1042	482
210.76.63.49:1083	511
64.124.157.32:65535	208

The system with source IP address 209.223.8.35 is using port 27374, which indicates possible SubSeven 2.0 trojan traffic. The destination port 1214 is the well-known KaZaA server port, a popular peer to peer file sharing program. The SubSeven trojan is a Windows remote control program that allows a client to control over 100 functions of the server computer.

Here is a sample from the Alert file for the trojan activity:

02/28-11:35:55.686663 [**] Possible trojan server activity [**] 209.223.8.35:27374 -> MY.NET.150.41:1214
02/28-11:35:55.705152 [**] Possible trojan server activity [**] 209.223.8.35:27374 -> MY.NET.150.41:1214

I used the American Registry for Internet Numbers (ARIN) Whois program to search ARIN's database network information. The ARIN output for 209.223.8.35 is as follows:

Continental Airlines (NETBLK-SAVV-CONTINE1)
1600 Smith
Houston, TX 77019
US
Netname: SAVV-CONTINE1
Netblock: 209.223.8.32 - 209.223.8.63
Coordinator:
Gold, Andre (AG290-ARIN) agold@coair.com

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

(713) 324-2339
Record last updated on 12-Jan-2000.
Database last updated on 11-Mar-2002 19:58:33 EDT.

The system with source IP address 63.250.205.43 is using source and destination ports as 0, which is an illegal packet. This could be an indication of packet crafting for fingerprinting or some sort of DoS attempt. The fact that the IDS alerted the trace as a large UDP packet also indicates a possible DoS attempt.

Here is a sample from the Alert file for the source port 0 traffic:

```
02/26-12:39:03.373442  [**] MISC Large UDP Packet [**] 63.250.205.43:0 -> MY.NET.152.184:0
02/26-12:39:05.571128  [**] MISC Large UDP Packet [**] 63.250.205.43:0 -> MY.NET.152.184:0
```

The ARIN output for 63.250.205.43 is as follows:

Yahoo! Broadcast Services, Inc. (NETBLK-NETBLK2-YAHOOBS)
2914 Taylor st
Dallas, TX 75226
US
Netname: NETBLK2-YAHOOBS
Netblock: 63.250.192.0 - 63.250.223.255
Maintainer: YAHO
Coordinator:
Bonin, Troy (TB501-ARIN) netops@broadcast.com
214.782.4278 ext. 2278
Domain System inverse mapping provided by:
NS.BROADCAST.COM 206.190.32.2
NS2.BROADCAST.COM 206.190.32.3
ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
Record last updated on 29-Jun-2001.
Database last updated on 11-Mar-2002 19:58:33 EDT.

The system with source IP address 211.233.70.162 is using port 1042, which indicates possible NetSpy trojan traffic. The NetSpy trojan is a Windows remote control program.

Here is a sample from the Alert file for the trojan activity:

```
02/27-17:06:21.355702  [**] MISC Large UDP Packet [**] 211.233.70.162:1042 -> MY.NET.152.213:2039
02/27-17:06:21.477752  [**] MISC Large UDP Packet [**] 211.233.70.162:1042 -> MY.NET.152.213:2039
```

The ARIN output for 211.233.70.162 is as follows:

inetnum 211.232.0.0 - 211.255.255.255
netname KRNIC-KR
descr KRNIC

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

```
descr          Korea Network Information Center
country         KR
admin-c         HM127-AP, inverse
tech-c         HM127-AP, inverse
remarks         ****************************************
remarks         KRNIC is the National Internet Registry
remarks         in Korea under APNIC. If you would like to
remarks         find assignment information in detail
remarks         please refer to the KRNIC Whois DB
remarks         http://whois.nic.or.kr/english/index.html
remarks         ****************************************
mnt-by          APNIC-HM, inverse
mnt-lower        MNT-KRNIC-AP, inverse
changed         hostmaster@apnic.net 20000908
changed         hostmaster@apnic.net 20010627
source         APNIC
person         Host Master, inverse
address         Korea Network Information Center
address         Narajongkeum B/D 14F, 1328-3, Seocho-dong, Seocho-ku, Seoul,
                   137-070, Republic of Korea
country         KR
phone          +82-2-2186-4500
fax-no         +82-2-2186-4496
e-mail         hostmaster@nic.or.kr, inverse
nic-hdl        HM127-AP, inverse
mnt-by          MNT-KRNIC-AP, inverse
changed         hostmaster@nic.or.kr 20010514
source         APNIC
```

The system with source IP address 210.76.63.49 is using port 1083, which indicates possible WinHole trojan traffic. WinHole is a trojanized version of Wingate proxy server, which allows remote access.

Here is a sample from the Alert file for the trojan activity:

```
03/02-17:21:35.626913  [**] MISC Large UDP Packet [**] 210.76.63.49:1083 -> MY.NET.153.136:1983
03/02-17:21:35.728892  [**] MISC Large UDP Packet [**] 210.76.63.49:1083 -> MY.NET.153.136:1983
```

The ARIN output for 210.76.63.49 is as follows:

```
inetnum         210.76.63.0 - 210.76.63.255
netname         HLJDZJ
descr          Temblor board of Heilongjiang Province
country         CN
admin-c         ZZ153-AP, inverse
tech-c         ZZ153-AP, inverse
mnt-by          MAINT-CNNIC-AP, inverse
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

changed	zhguzh@ems.dragon.net.cn 20010330
source	APNIC
person	ZhangGuangZhong Zhang, inverse
address	HeiJLongJiang HaRBin ZhongShan Road#204 of china
country	CN
phone	+86-0451-2624234
fax-no	+86-0451-2619722
e-mail	zhguzh@ems.dragon.net.cn, inverse
nic-hdl	ZZ153-AP, inverse
mnt-by	MAINT-CNNIC-AP, inverse
changed	zhguzh@ems.dragon.net.cn 20010328
source	APNIC

The system with source IP address 64.124.157.32 is using port 65535, which indicates possible RedWorm traffic. Snort.org also lists this port as the RC1 trojan, which is a remote control program.

Here is a sample from the Alert file for the activity:

```
02/27-19:49:15.302936  [**] High port 65535 udp - possible Red Worm - traffic [**] 64.124.157.32:65535 ->
MY.NET.153.199:65535
02/27-19:49:27.980441  [**] High port 65535 udp - possible Red Worm - traffic [**] 64.124.157.32:65535 ->
MY.NET.153.199:65408
```

While investigating this source address, the following alerts were also discovered:

```
02/27-19:55:09.373742  [**] EXPLOIT NTPDX buffer overflow [**] 64.124.157.32:123 -> MY.NET.153.199:123
02/27-21:25:55.996916  [**] Attempted Sun RPC high port access [**] 64.124.157.32:37194 ->
MY.NET.153.199:32771
02/27-22:06:39.775483  [**] RPC udp traffic contains bin sh [**] 64.124.157.32:32311 -> MY.NET.153.199:63683
```

The ARIN output for 64.124.157.32 is as follows:

```
Abovenet Communications, Inc. (NETBLK-ABOVENET)
    50 W. San Fernando Street, Suite 1010
    San Jose, CA 95113
    US
    Netname: ABOVENET
    Netblock: 64.124.0.0 - 64.125.255.255
    Maintainer: ABVE
    Coordinator:
       Metromedia Fiber Networks/AboveNet  (NOC41-ORG-ARIN)
       noc@ABOVE.NET
      408-367-6666
  Fax- 408-367-6688
    Domain System inverse mapping provided by:
    NS.ABOVE.NET              207.126.96.162
    NS3.ABOVE.NET             207.126.105.146
```


© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

ADDRESSES WITHIN THIS BLOCK ARE NON-PORTABLE
Record last updated on 27-Apr-2001.
Database last updated on 11-Mar-2002 19:58:33 EDT.

Each of the above external source addresses should be further investigated for possible trojan and exploit attempts. The alerts provided above indicate possible compromised systems on the internal network. Compromised systems can be used for attack purposes for DoS attacks as well as other intrusion attempts.

3.5 CORRELATIONS

As part of my analysis process I consulted numerous web sites and other student practicals. I have sited the websites at the end of this document, among them, www.sans.org, www.incidents.org, www.snort.org, www.cert.org, and www.securityfocus.com. I consulted other student practicals to determine the format and expectations for the document and to gain information on the process of formatting, compiling, and analyzing the data. I have listed below the practicals that I used most as references and correlating my findings.

Jamil Farshchi's practical contains detailed information on a variety of attacks. He also correlates the data well with other practicals and external sources.

I referenced James Hoover's practical for information on Red Worm, connect to 515 from inside, and SNMP public access.

I referenced IE Naumann's practical for information on Red Worm, INFO MSN IM Chat data, and connects to TCP port 515.

I referenced Dennis Ruck's practical for information on MISC Large UDP Packet, High port and Red Worm traffic.

I referenced David Leach's practical for information on SMB Name Wildcard, connect to 515 from inside, and Red Worm.

I referenced Jeff Zahr's practical for information on spp_http_decode: IIS Unicode attack and Red Worm.

Stan Hoffman's practical has some good explanations and correlation data for alerts. I referenced his information on MISC Large UDP Packet, INFO MSN IM Chat data, ICMP Echo Request Nmap or HPING2, and Red Worm. His destination port traffic also correlated to the same type of data I was seeing, with ports such as 80, 53, 0, 137, and 1214.

I referenced Steve Lukacs practical for information on Red Worm, MISC Large UDP Packet, INFO MSN IM Chat data, ICMP Echo Request Nmap or HPING2, and SNMP public access.

I noticed a trend in the practicals that I read in terms of trojans, Red Worm, messaging, and file sharing activities. These types of activities were prevalent and numerous is most practicals. I

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

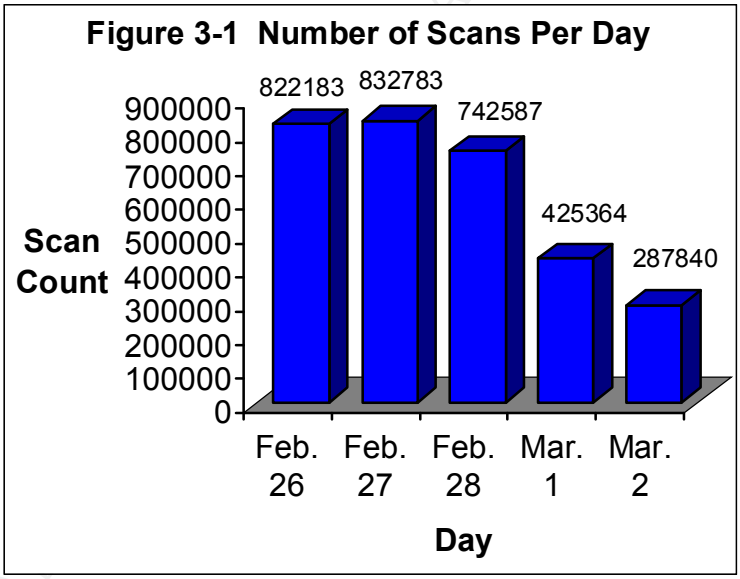
also noticed that over time the number of alerts and scans have increased. I am unsure if this is related to an increase in attacks, traffic, false positives, or changes in monitoring.

3.6 DATA GRAPH AND ANALYSIS

There are two reasons that I chose the five-day span of Feb. 26 through Mar. 2. First, I wanted to analyze trend data that included data from both the workweek and the weekend. The data meets this requirement by spanning Tuesday through Saturday. Second, I wanted to analyze trend data that covered both the end of the month and the first of the month. I wanted to see if these two characteristics had any affect on the volume or type of detects. I have produced several graphs below that show various aspects of the correlated data.

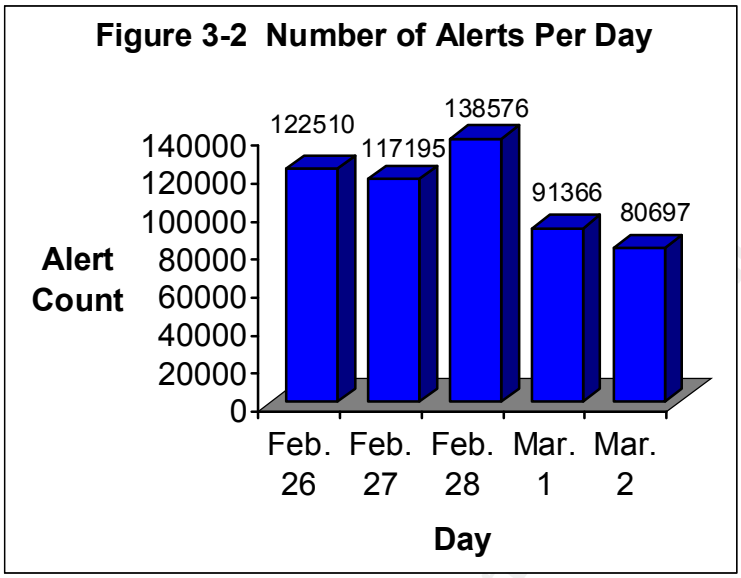
The graph in Figure 3-1 shows the number of scans per day for the five-day period. Notice that the number of scans decreases as the week ends, with Saturday having the least number of scans. I was surprised by this trend because I assumed that attackers would be more likely to run scans and attempt to penetrate the network on the weekend. I attributed this to the fact that attackers may have more time on the weekend and that they may be less noticed. The graph leads me to believe that some of the daily network traffic may be generating false positives.

Figure 3-1 Number of Scans Per Day



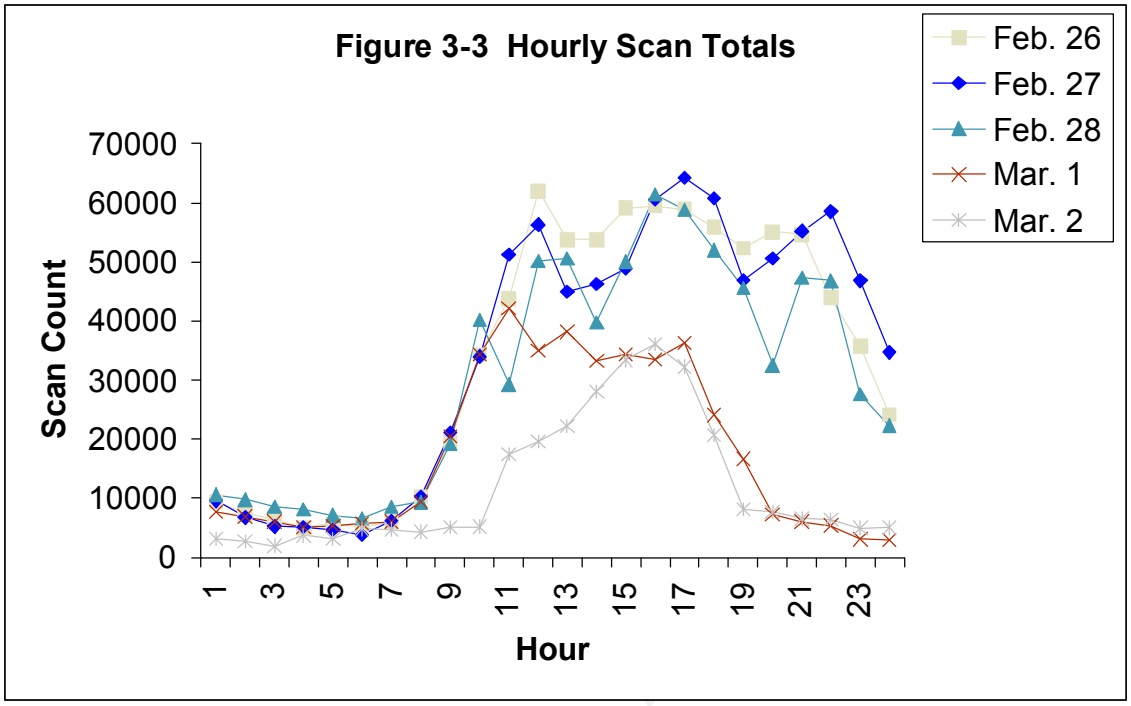
The graph in Figure 3-2 shows the number of alerts per day for the five-day period. The alert data shows the same trend as the scan data by decreasing at the end of the week, with the exception of the slight spike on Thursday, Feb. 28. I would suggest a more in-depth analysis to determine which types of alerts are significantly less at the end of the week. This may show what type of traffic is less at the end of the week and may point out any network traffic that may be generating false positives.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

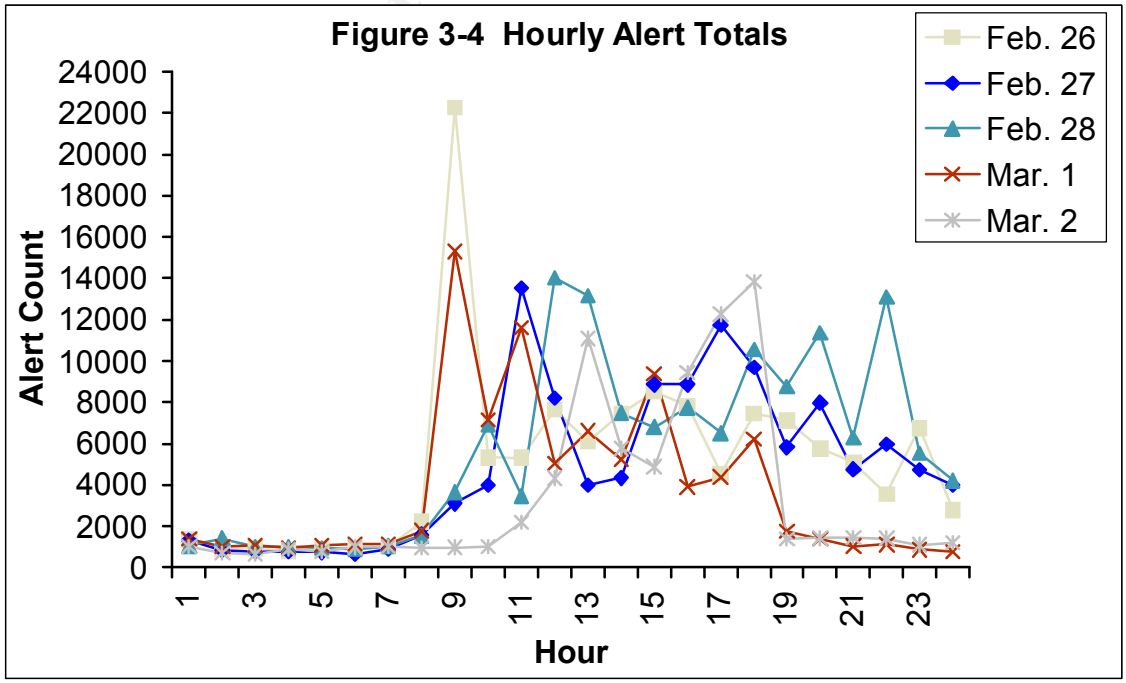


The graph in Figure 3-3 shows the number of scans per hour for each day in the five-day period. Notice that the scanning begins to increase around 8 a.m. and begins to decrease around 6 p.m. This trend reflects a normal workday. Once again I began to wonder if normal network traffic may be generating false positives.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.



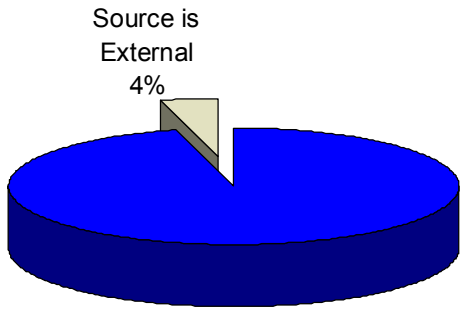
The graph in Figure 3-4 shows the number of alerts per hour for each day in the five-day period. Once again the scanning begins to increase around 8 a.m. and begins to decrease around 6 p.m. I suggest further investigating the types of traffic on the network during the workday to determine if there are false positives generating alerts. I also suggest investigating the spike at 8 a.m. on Feb. 26.



© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

The pie graph in Figure 3-5 shows the percentage of scanning data that originates from inside of the network and the percentage that originates from outside of the network. Notice that almost all of the scanning data originates internally.

Figure 3-5 Scanning Source Address Percentage



The pie graph in Figure 3-6 shows the percentage of scanning data with destination addresses inside of the network and the percentage with destination addresses outside of the network. Notice that most of the data has an internal destination.

Figure 3-6 Scanning Destination Address Percentage

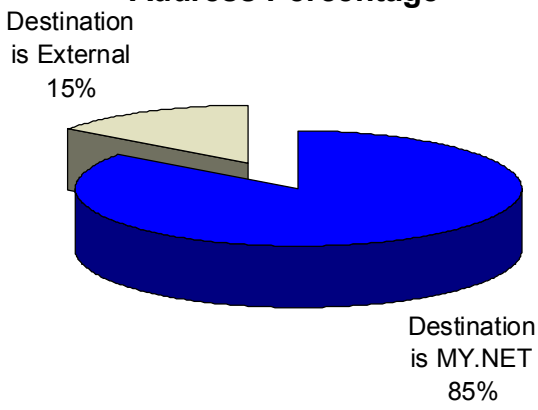
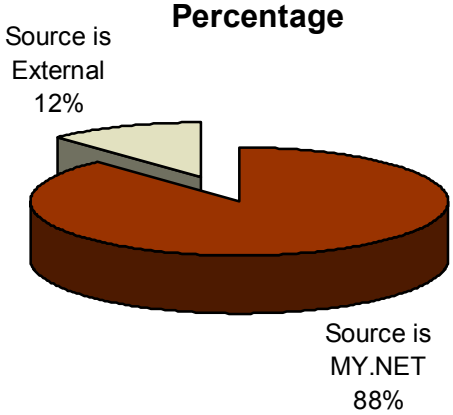
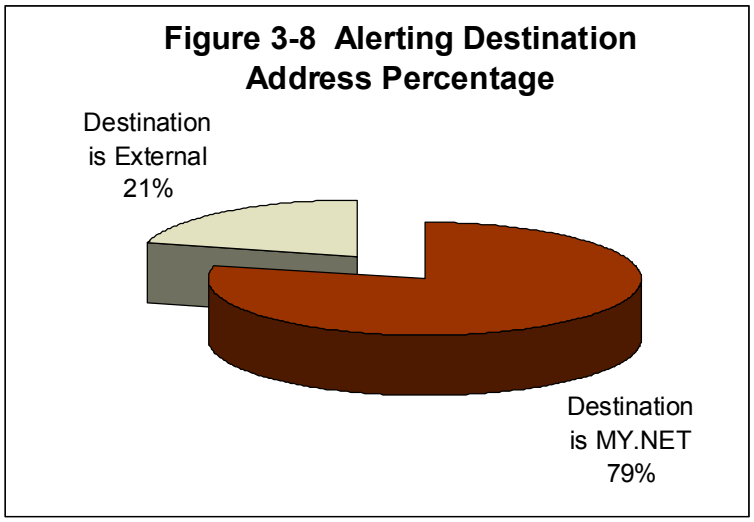


Figure 3-7 Alerting Source Address Percentage



The pie graph in Figure 3-7 shows the percentage of alert data with source addresses inside of the network and the percentage with source addresses outside of the network. Notice again that most of the data originates internally.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.



The pie graph in Figure 3-8 shows the percentage of alert data with destination addresses inside of the network and the percentage with destination addresses outside of the network. Notice again that most of the data has an internal destination. All four pie charts indicate a lot of internal activity. Once again this should be further investigated to determine if false positives exist during normal network traffic.

3.7 CRITICAL ACTIVITY

Each of the machines listed in the External Source Address Information section should be further investigated for possible trojan and exploit attempts. The alerts provided indicate possible compromised systems on the internal network. Compromised systems can be used for attack purposes for DoS attacks as well as other intrusion attempts.

There are also several internal machines that show evidence of SubSeven 2.0 activity. Each of the following machines is communicating with the internal machine MY.NET.5.83 via port 27374.

MY.NET.5.44

02/27-00:10:37.668501 [**] Possible trojan server activity [**] MY.NET.5.44:27374 -> MY.NET.5.83:9321
02/27-00:10:37.718673 [**] Possible trojan server activity [**] MY.NET.5.83:9321 -> MY.NET.5.44:27374
02/27-00:10:37.720782 [**] Possible trojan server activity [**] MY.NET.5.44:27374 -> MY.NET.5.83:9321
02/27-00:10:37.742602 [**] Possible trojan server activity [**] MY.NET.5.83:9321 -> MY.NET.5.44:27374
02/27-00:10:37.899180 [**] Possible trojan server activity [**] MY.NET.5.44:27374 -> MY.NET.5.83:9321

MY.NET.5.88

02/26-02:33:04.153728 [**] Possible trojan server activity [**] MY.NET.5.88:27374 -> MY.NET.5.83:9321
02/26-02:33:04.171026 [**] Possible trojan server activity [**] MY.NET.5.83:9321 -> MY.NET.5.88:27374
02/26-02:33:04.172368 [**] Possible trojan server activity [**] MY.NET.5.88:27374 -> MY.NET.5.83:9321
02/26-02:33:04.172512 [**] Possible trojan server activity [**] MY.NET.5.83:9321 -> MY.NET.5.88:27374
02/26-02:33:04.219032 [**] Possible trojan server activity [**] MY.NET.5.83:9321 -> MY.NET.5.88:27374
02/26-02:33:04.219098 [**] Possible trojan server activity [**] MY.NET.5.88:27374 -> MY.NET.5.83:9321

MY.NET.5.55

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

02/28-10:34:51.981922 [**] Possible trojan server activity [**] MY.NET.5.55:27374 -> MY.NET.5.83:9321
02/28-10:34:51.981993 [**] Possible trojan server activity [**] MY.NET.5.83:9321 -> MY.NET.5.55:27374
02/28-10:34:51.982055 [**] Possible trojan server activity [**] MY.NET.5.55:27374 -> MY.NET.5.83:9321
02/28-10:34:52.000331 [**] Possible trojan server activity [**] MY.NET.5.83:9321 -> MY.NET.5.55:27374
02/28-10:34:52.000843 [**] Possible trojan server activity [**] MY.NET.5.55:27374 -> MY.NET.5.83:9321

MY.NET.5.77

03/01-02:59:44.257009 [**] Possible trojan server activity [**] MY.NET.5.77:27374 -> MY.NET.5.83:7938
03/01-02:59:44.257077 [**] Possible trojan server activity [**] MY.NET.5.83:7938 -> MY.NET.5.77:27374
03/01-02:59:44.257145 [**] Possible trojan server activity [**] MY.NET.5.77:27374 -> MY.NET.5.83:7938
03/01-02:59:44.260000 [**] Possible trojan server activity [**] MY.NET.5.77:27374 -> MY.NET.5.83:7938
03/01-02:59:44.267740 [**] Possible trojan server activity [**] MY.NET.5.83:7938 -> MY.NET.5.77:27374

MY.NET.5.34

03/01-03:04:16.685579 [**] Possible trojan server activity [**] MY.NET.5.83:9321 -> MY.NET.5.34:27374
03/01-03:04:16.685646 [**] Possible trojan server activity [**] MY.NET.5.34:27374 -> MY.NET.5.83:9321

Due to the suspicious nature of the above activity, the machine MY.NET.5.83 should be investigated. Each of the possibly compromised machines should be scanned for the SubSeven trojan and disinfected if needed. For more information on the SubSeven trojan please refer to the SANS IDS FAQ at http://www.sans.org/newlook/resources/IDFAQ/subseven.htm.

3.8 DEFENSE RECOMMENDATIONS

Universities offer unique circumstances in terms of information security. It is often difficult to restrict the openness of educational institutions. Because of this, certain types of data must be allowed into and out of the network. Appropriate segregation of various parts of the network (i.e. residential, research, administration, etc.) can alleviate some security headaches. I have offered below some security suggestions for university information security.

1. The security policy is always the foundation for protecting information. A comprehensive review of the university's security policies should be completed including consideration of legal liabilities. This includes restricting the use of peer-to-peer file sharing services such as Napster, Gnutella, and KaZaA, which are notorious for being used to illegally exchange copyrighted material. Acceptable use polices should be developed and communicated to each student, faculty, and staff.

2. Incorporate a network perimeter defense that will prevent network activities that are prohibited by the security policy. A firewall should be installed to protect the MY.NET network from external attack by providing ingress filtering of ports and services. The firewall should be configured to specifically permit authorized traffic and deny all other traffic. Network address translation can also be used to protect the topology of MY.NET.

3. A high percentage of the alerts generated are related to instant messaging and file-sharing utilities. If these utilities are to be permitted, it would be more effective to

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

filter these alerts out of the ruleset to prevent the generation of excessive alerts for approved traffic.

4. Discontinue the use of the default SNMP community string (public).

5. Perform a network analysis on the type of traffic occurring during a regular workday. Use this information to update firewall and IDS rulesets. This may alleviate false positives and decrease the size of the logs to analyze.

6. Update the Snort Watchlist to include the IP addresses that are most actively scanning or otherwise targeting the university network.

7. Perform regular network port scans for proxy services listening on popular ports and other services that should not be running to proactively eliminate these vulnerabilities.

8. Investigate all indicators of trojan activity. Compromised systems can be used for attack purposes for DoS attacks as well as other intrusion attempts.

9. Keep all patches up to date on all servers, desktops, and network devices.

10. Keep all virus software up to date.

11. Install personal firewalls and host based intrusion detection systems on all critical systems.

12. Perform regular audits on the security architecture and systems.

13. Train University administrators on intrusion detection.

14. Develop a Security Awareness project to educate users on security issues.

3.9 ANALYSIS PROCESS

I began my analysis by downloading the appropriate *.gz files and renaming them to *.gx.txt to open them in Notepad. I scanned through each of the files to gain a better understanding of the format and size of the logs. I decided to import all of the logs to a Microsoft Access database. I had to use a combination of Microsoft Word and Unix grep and sed commands to edit the data and add delimiters. I used Microsoft Word to perform a "find and replace" to delimit the data. When Word became too cumbersome and time consuming due to the size of some of the files I transferred the files to my Solaris 8 machine and used grep and sed. I then transferred the files back and imported them into the database. I processed all of the Scan logs into a Scan table. I separated the Alert log into two sections, Alerts and Scan data. Filtering out the scan data from the Alert logs helped to query the Alert logs more efficiently and to correlate the scan data with the Scan logs more effectively. I then created the OOS table and manually entered the data since there were so few entries.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

Once the tables were built I began to run queries to filter and summarize the data. This allowed me to total the top talkers, types of alerts and scans, and hourly and daily totals. I was also able to run specific queries on individual addresses and ports. I was also able to search for specific attacks based on the alert type and port information. I then researched the attacks, ports, and defensive measures. I was able to cut and paste my queries into Microsoft Excel to generate charts and graphs. I also used Excel to generate formulas to calculate percentages for my results and to format the tables for the document. Having a tool such as a database greatly helps to analyze the large volumes of data that can result from intrusion detection logs.

© SANS Institute 2000 - 2002, Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.

APPENDIX A: REFERENCES

"Analyze This" intrusion log data. http://www.research.umbc.edu/~andy.

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

Blade Software. http://www.blade-software.com.

CERT. http://www.cert.org.

The Ideahampster Organization. Tools – Intrusion Detection System (IDS) Testing. January 29, 2002. http://www.ideahampster.org/tools/ids.shtml.

Incidents.org. http://www.indicents.org.

Phrack 55 Perl CGI Problems. http://www.phrack.com/phrack/55/P55-07.

SANS. http://www.sans.org.

SC Info Security Magazine. Product Review: Blade IDS Informer. March 2002.

Security Focus. http://www.securityfocus.com.

Shipley, Greg. Network Computing. Intrusion Detection, Take Two. November 15, 1999. http://www.networkcomputing.com/1023/1023f19.html.

Shipley, Greg. Re: Generating Traffic to Stress Test IDS. SecurityFocus Online. January 24, 2002. http://online.securityfocus.com/archive/96/252328.

Snort. http://www.snort.org.

Thurman, Mathias. Computerworld. Testing Intrusion-detection systems. May 22, 2001. http://www.linuxworld.com.au/article.php3?aid=166&tid=4.

© SANS Institute 2000 - 2002 Author retains full rights.

© SANS Institute 2000 - 2002 As part of GIAC practical repository. Author retains full rights.