Experiences with a Generation III Virtual Honeynet

Fahim H. Abbasi and R. J. Harris
School of Engineering and Advanced Technology (SEAT)
Massey University
New Zealand
{F.Abbasi, R.Harris}@massey.ac.nz

Abstract:

This paper proposes a methodology for establishing a virtual Honeynet on a VMware Server running Honeywall CDROM Roo. The implementation is specific to a Linux based host having a single physical network interface card. Security of virtual Honeynets is always a concern, special techniques are discussed in the paper to ensure their security and to mitigate associated risks posed to the host and virtual machines. An effort has been made to ensure that all the software (both the OS and associated tools) used for the project are either free or Open Source. Special techniques were implemented in order to enhance the data capture mechanisms on the Linux-based Honeypot to efficiently generate reports. Risk evaluation and suggestions for improvements to the methodology are proposed.

I. Introduction:

The revolution in Information Technology has provided a flood of assets in the form of applications and services. Enterprises have based their entire business models on top of these assets. Networks have evolved from low speed half duplex links to full duplex, multi-homed, self-convergent, gigabyte streams, controlled by advanced protocols. The security of the available applications and services accessible over these networks currently represents a major challenge to the IT industry. Each day, exploits, worms, viruses and buffer overflows severely threaten the IT infrastructure and associated business assets along with mission critical systems. By learning the tactics and techniques used by malicious *black hats* crackers, we can secure our IT assets and infrastructure. Honeypots provide a means to study black-hat techniques and tactics through which they have been able to gain illegitimate access to system resources along with methods for analyzing the tools that they use to obtain this access. This is achieved by setting up a vulnerable environment that poses as a valid resource to any attacker, but is heavily logged.

Most network security tools are passive in nature; for example, firewalls and IDS. They operate on available rules and signatures in their database. Anomaly detection is limited only to these set of available rules. Any activity not in alignment with those rules goes undetected. Honeypots by design allow you to take the initiative by turning the tables on malicious black hats. The Honeypot system has no production value and has no authorized activity. Thus any interaction with the Honeypot is most likely the result of malicious intent. Honeypots do not solve the security problem but provide data and knowledge that aids the system administrator in enhancing the overall security of their network. This knowledge can be used as input to any early warning systems. Over the years, researchers have successfully isolated and identified worms and exploits using Honeypots placed in specialized architectures called Honeynets. These are then used for signature and rule development. Honeynets are capable of logging far more information than any other available security tools. They give insight into attacks and attackers, their skill level, their organization as groups or individuals, their motives and tactics; and thus, almost every aspect is logged and can be made auditable.

Virtualization technologies like VMware^{TM} [18] provide the ability and flexibility to create a specialized network of hosts on a single physical machine. This has considerably reduced hardware costs.

This project has been setup using free and Open Source tools and technologies that run on a Linux platform. Linux based operating systems have been used as the host OS and for guest OS virtual machines. This includes a Linux based Honeypot and a Honeywall gateway. Based on the results of this study we can enhance the overall security of our network resources.

The remainder of the paper is organized as follows: Section II defines and describes the technology that has been employed and briefly discusses the evolution of Honeynets. Section III discusses the tools used for setting up the Honeynet resource. Section IV outlines the problem statement and discusses our proposed design and implementation details. Section V evaluates the effectiveness of the virtual Honeynet by examining data that has been collected and correlates it with identifiable attacks and suspicious flows. Section VI summarizes the project and the overall effectiveness and efficiency of a virtual Honeynet in any organization. Finally Section VII discusses ideas to mark the road ahead to enhance this technology.

II. Background

A. Honeypots

A Honeypot is generally defined as a network security resource whose value lies in it being scanned, attacked, compromised, controlled and misused by an attacker to achieve his malicious goals. Lance Spitzner defines Honeypots as “A Honeypot is an information system resource whose value lies in unauthorized or illicit use of that resource” [1].

Honeypots can be classified into two main categories. Firstly, they can be based upon their level of interaction with an attacker. This can be further categorized as:

- **Low-interaction:** Emulate a variety of host services. These mimic real services but are implemented as a sandbox environment and run as an application. e.g. honeyd [6] and nepenthes.

- **High-interaction:** Attacker is given the freedom to interact with a real operating system and their every attempt is logged and accounted for.

The second Honeypot category is identified by the way they are deployed in a network. This includes:

- **Production Honeypots:** They are placed within an organization's production network for the purpose of detection. They extend the capabilities of intrusion detection systems. Such Honeypots are developed and configured to integrate with the organization's infrastructure. They are usually implemented as low-interaction Honeypots sitting within the server farm, but implementations may vary depending on available funding and requirements of the organization.

- **Research Honeypots**: These are deployed by network security researchers – the *white hat hackers*. They allow complete freedom for the attacker and, in the process; it is possible to learn their tactics. Using Research Honeypots zero day exploits, Worms, Trojans and viruses propagating in the network can be isolated and studied. Researchers can then document their findings and share them with system programmers, network and system administrators, various system and anti-virus vendors. They provide the raw material for the rule engines of IDS, IPS and firewall systems.

The identity of the Honeypot is crucial and we can conclude that the learning curve (from the attacker) is directly proportional to the level of stealth for the Honeypot [1,2,3,4]. Research Honeypots are usually deployed at Universities and by the R&D departments of various organizations as High-Interaction Honeypots.

C. Honeynet

A Honeynet is a special kind of high-interaction Honeypot. Honeynets extend the concept of a single Honeypot to a highly controlled network of Honeypots. A Honeynet is a specialized network architecture configured in a way to achieve:

- **Data Control**: It deals with the containment of activity within the Honeynet.
- **Data Capture:** It involves the capturing, monitoring and logging of all threats and attacker activities within the Honeynet
- **Data Collection:** Captured data is securely forwarded to a centralized data collection point.

This architecture creates a highly controlled network, in which one can control and monitor all kinds of system and network activity. Honeypots are then placed within this network. A basic Honeynet comprises of Honeypots placed behind a transparent gateway – the Honeywall. Acting as a transparent gateway the Honeywall is undetectable by attackers and serves its purpose by logging all network activity going in or out of the Honeypots.

D. Honeynet Architectures:

Depending on the technologies adopted, and the way data capture, control and collection activities have been carried out within the Honeynet network over the years, the Honeynet has evolved across 3 architectures or Generations as outlined below:

Generation I

Gen I Honeynet was developed in 1999 by the Honeynet Project. The architecture was simple with a firewall aided by an IDS as the gateway and Honeypots placed behind it. This architecture required 2 interfaces on the Honeywall gateway, one facing the external network and one facing the Honeypot's internal network. This architecture was flawed as the gateway acting as a Layer 3 device could be detected by attackers.

Generation II & III

Change in architecture was brought about by the introduction of a single device that handles the data control and data capture mechanisms of the Honeynet called the IDS Gateway or the Honeywall. This is implemented as a transparent bridge.

Gen II Honeynets were first introduced in 2001 and Gen III Honeynets were released at the end of 2004. Gen II Honeynets were made in order to address the deficiencies in Gen I Honeynets. Gen II and Gen III Honeynets have the same architecture, with the only difference being improvements in deployment and management in Gen III Honeynets along with the addition of a Sebek server built in the gateway – this is known as the Honeywall.

This architecture incorporates 3 interfaces on the Honeywall. Two interfaces acted as a bridge between the external network and the internal Honeypot network; whilst the third interface was used for management and configuration tasks.

E. Virtual Honeynet

Virtualization is a technology that allows running multiple virtual machines on a single physical machine. Each virtual machine can be an independent Operating System installation., running concurrently with others This is achieved by sharing the machine's physical resources such as CPU, memory, storage and peripherals through specialized software across multiple environments. This reduces project hardware costs.

A virtual Honeynet is a complete Honeynet running on a single computer in virtual environment.

VMware Server [18] was used as the virtualization solution for our project. VMware Server was selected because:
It is free, reliable, has large community support, and extensive documentation, Previous experience of managing virtual machines with VMware, it is flexible and has robust networking components. For the scope of our project, we used VMware Server version 1.0.6 for Fedora Linux.

III. Tools

A. On the Gateway

For our project implementation the aim was to use free or Open Source tools. The Honeynet Project is a non-profit, Open Source security research organization. This organization has actively published papers, developed and contributed Open Source security tools. For the scope of our project we used Honeywall CDROM Roo [22] version 1.4. Honeywall CDROM is a bootable CDROM operating system built on CentOS for installing, deploying and maintaining a Honeynet. The purpose of the Honeywall CDROM is to automate the installation and maintenance of a Honeynet and provide data analysis support for all activity within the Honeynet.

Honeywall Roo includes many well-known security tools such as:

- **Snort**[17]: Intrustion Detection System (IDS)
- **Snort_inline:** Intrusion Prevention System (IPS)**.**
- **Argus [15], Pof:** Passive OS fingerprinting tool
- **Tcpdump:** Viewing of packet headers.
- **Hflow2**: A data coalescing tool for Honeynet data analysis.
- **Walleye [22]:**Web based interface for Honeywall configuration, administration and data analysis.
- **Sebek [11]:** Sebek is a data capture tool designed to stealthily capture attacker's activities

Balas and Viecco [16] have given a generalized data collection and fusion diagram for a Generation III Honeywall. Extending their work further we propose an extended diagram for Honeywall Roo [22] Logical Design in Figure 1:

B. On the Honeypot

We implemented a standard Ubuntu server as our Honeypot. This server had basic services running that included SSHD, FTPD and HTTPD etc. Open SSH was patched and custom compiled to add both a user name and a password logging capability. The passwords were logged to a secure hidden directory within the server. A bash script was used to parse the logs and associate them with attacker IP's. Based on a timestamp difference, this script could distinguish between SSH scanners/crackers and interactive SSH sessions.

Sebek was also installed on the Honeypot and concealed as a printer driver name. It collected valuable system information such as attacker keystrokes, processes and associated system calls. After analysis of script and Sebek data with Honeynet flows, we could correlate attacks and attack techniques effectively.

IV. Proposed Architecture

The Honeynet Project provides some general documentation on deploying Generation III virtual Honeynets, this documentation was developed by the Pakistan Honeynet Project Chapter. The document was a *How-To* for deploying virtual Honeynets using VMware. This served as a standard template for anyone wanting to deploy a virtual Honeynet using VMware and Honeywall Roo.

A. Problem Identification & Solution

During our literature review it was decided to use [12] as the standard template for our project's implementation. Using VMware, a bridged interface like vmnet0 has direct access to the physical network interface. Bridging two such interfaces will cause a bridge between the same LAN segment resulting in loops in the network, whereas the requirement was to bridge between two different LAN segments i.e. the external network segment pointing to the router and the internal network segment on pointing towards the Honeypots.

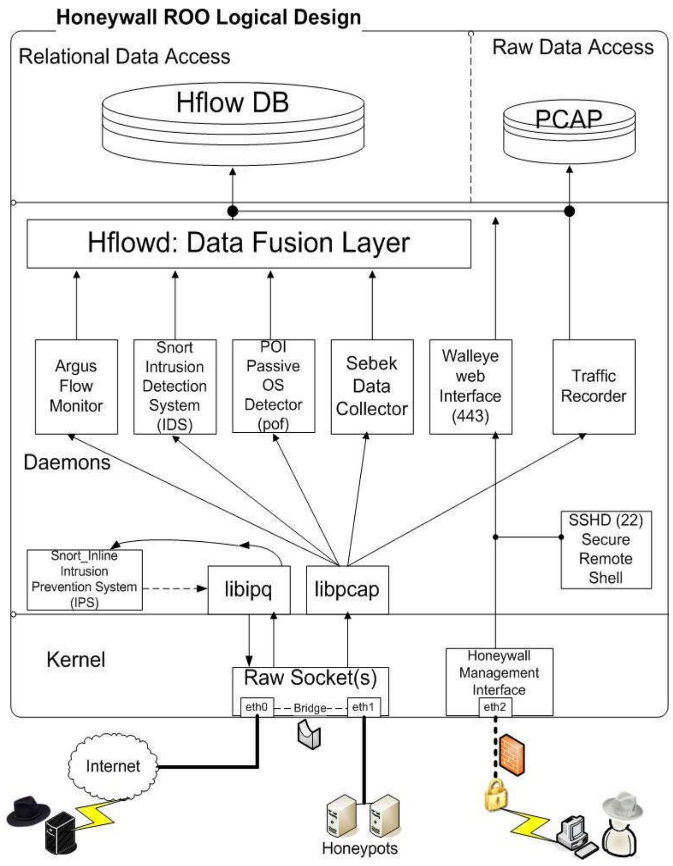
Honeywall ROO Logical Design
Relational Data Access
Raw Data Access
Hflow DB
PCAP
Hflowd: Data Fusion Layer
Argus Flow Monitor
Snort Intrusion Detection System (IDS)
POI Passive OS Detector (pof)
Sebek Data Collector
Walleye web Interface (443)
Traffic Recorder
Daemons
SSHD (22) Secure Remote Shell
Snort_Inline Intrusion Prevention System (IPS)
libipq
libpcap
Kernel
Raw Socket(s)
eth0
Bridge
eth1
Honeywall Management Interface
eth2
Internet
Honeypots

Figure 1: Honeywall Roo Logical Design

It was observed that the Honeynet design suggested by [12] configured both eth0 & eth1 interfaces as a VMware bridge interface and eth2 as a VMware host- only interface. This was causing loops in the topology and the Honeypot LAN segment was being avoided and unroutable. This problem was extended to the Pakistan Honeynet Project with a complete proof of concept in June, 2008. They accepted and updated the design on their website.

B. Design Details and Discussion

Similar design problems were being faced and discussed by security researchers all over the globe who wanted to implement a similar virtual Honeynet. Many of these questions appeared in the official Honeywall community mailing list. We shared and discussed our findings with the community on the Honeywall project mailing list. After necessary testing a design was prepared and followed for the project implementation. Based on the success of this effort it was decided to publish the improved design. This design proposes 3 virtual interfaces for the virtual Honeywall such that:

1. Interface "vmnet0" (on eth0) is a VMware bridge interface pointing towards the router. (As shown in Figure 2)
2. Interface "vmnet1" (on eth1) is a VMware host-only interface leading to the internal LAN segment where Honeypot(s) are kept. (Also shown in the figure).

3. Interface "vmnet2" (on eth2) is a VMware bridge interface that is firewalled and accessible for remote management purpose (remote access via SSH and Walleye.)

Interfaces 1 and 2 are picked up by Roo as eth0 and eth1 and are used for bridging. Interface 3 is used for remote management. As shown in Figure 2 the doted boxes indicate the publically assigned IP addresses. In this case, the host machine's eth0 interface and the Honeypot virtual machines' (1, 2 or many) are assigned public IP addresses. The Honeywall management interface (i.e. interface 3) is assigned a private IP from the Host machine's virtual

Remote management interface can be configured in many ways:

1. The interface can be routed to an internal virtual subnet (private IP subnet) on the same physical machine.
2. The interface can be routed to an internal/external routable network using a separate physical Ethernet controller on the physical machine.
3. The interface can be routed on the same subnet as of the Honeynet sharing the same physical machine's Ethernet controller.

For our implementation we assigned the management interface a public IP from within the Honeynet subnet, but restricted access to it only from a specific IP (via Roo [22] configuration). This project was implemented successfully with one physical gigabit Ethernet interface. Another physical interface could have been used by binding it with the remote management interface

.Results and Discussion

The virtual Honeynet was online for a period of approximately 60 days from 15th September 2008 to 15st November, 2008. During this period we received over 30,000 identifiable attack connections. The attack results were documented as attacked ports and services, Attacker IP's and Country of Origin. The first attack was documented after 4 days of setting up the Honeynet. After several port scans an attacker attempted a SSH brute force attack from "82.99.xx.xxx". Geo-location of the IP was retrieved [23] After several hundred attempts the attacker was successful in brute forcing a user account. A botnet client was installed from a free webhosting server and IRC [25] communication was initiated, the chat sessions were translated from Romanian to English using Google Translate service [24]. The tools and chat/commands were retrieved from this session successfully for further forensic analysis. During the project, five similar sophisticated attacks were observed, from which valuable information and tools have been successfully retrieved. Forensic analysis have revealed a depth of information on the attackers, their organization into groups, their ties with each other and some system credentials were logged during the chat exchange. These Forensic results are beyond the scope of this paper. However, we came to the conclusion that attackers originating from Europe are commanding an overly large army of zombie hosts in China and the US to gain access to targets across the globe. Servers are always a high value target for them as they offer a variety of services over stable high speed links.

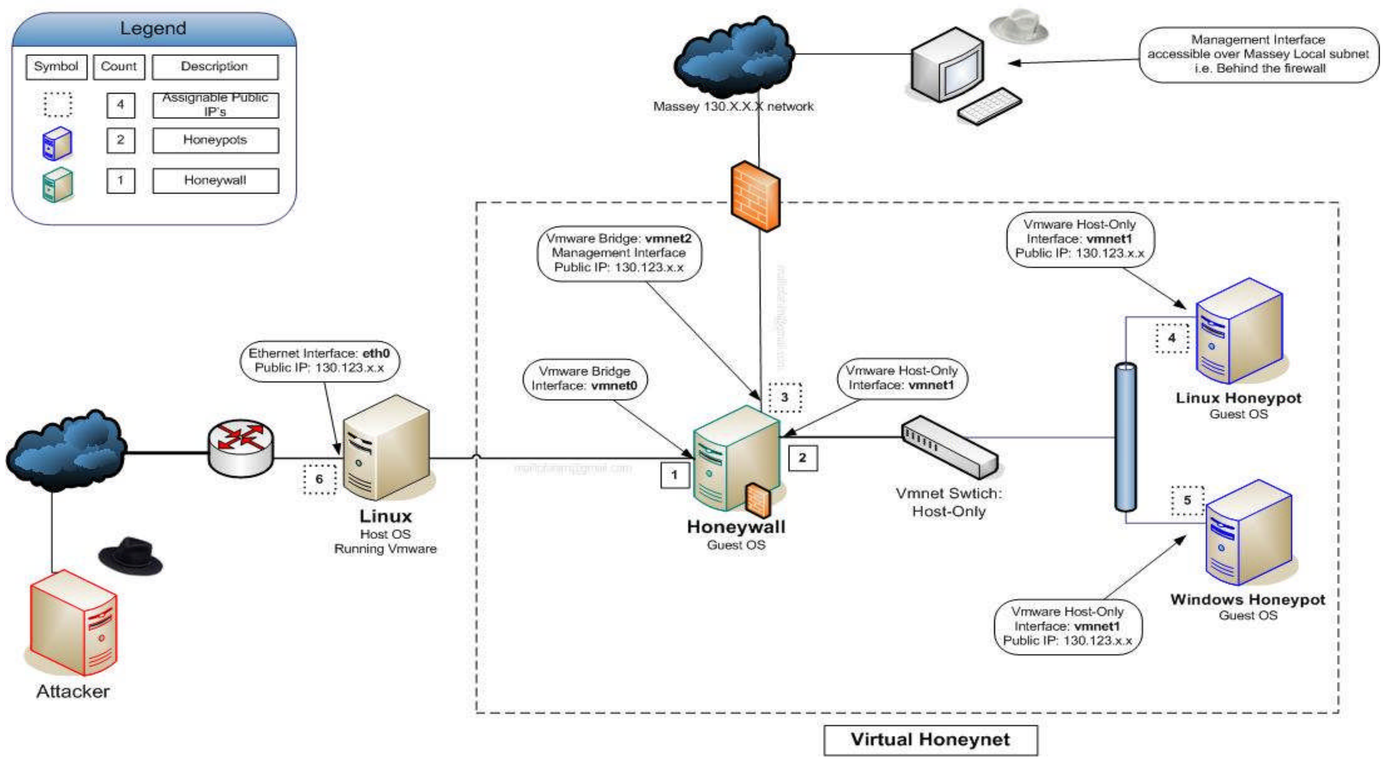
Legend
Symbol
Count
Description
4
Assignable Public IP's
2
Honeypots
1
Honeywall
Massey 130.X.X.X network
Management Interface accessible over Massey Local subnet i.e. Behind the firewall
Vmware Bridge: vmnet2 Management Interface Public IP: 130.123.x.x
Vmware Host-Only Interface: vmnet1 Public IP: 130.123.x.x
Ethernet Interface: eth0 Public IP: 130.123.x.x
Vmware Bridge Interface: vmnet0
Vmware Host-Only Interface: vmnet1
3
4
6
1
2
5
Linux
Host OS
Running Vmware
Honeywall
Guest OS
Vmnet Swtich:
Host-Only
Linux Honeypot
Guest OS
Windows Honeypot
Guest OS
Vmware Host-Only Interface: vmnet1 Public IP: 130.123.x.x
Attacker
Virtual Honeynet

Figure 2: Virtual Honeynet Proposed Design

A. Attacked Ports and Services

A small observation of attacked ports and services was made, which revealed that, out of total of 29,643 probed ports and services, 29,048 were targeted at SSH. This indicates the attackers' focus on brute force means of gaining access to the server. This is followed by high activity on IRC ports indicating botnet activity.

B. Attacker IP's

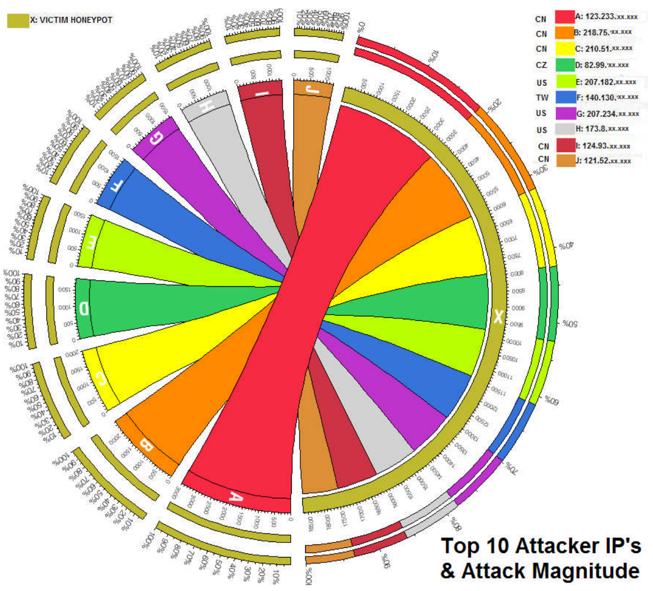

Figure 3: Top 50 Attackers by IP Address

During its 60 day tenure, the Honeypot received **34263** attacks from **615** unique IP's. A great many of these attacks originated from Europe and China, followed by the US.

C. Attackers Country of Origin

615 unique attacker IP addresses were identified originating from **79** countries across the globe [23]. Out of these 79 countries the highest number of attacks came from China and Europe followed by the US. The same proportion also stands for the highest attack frequencies.

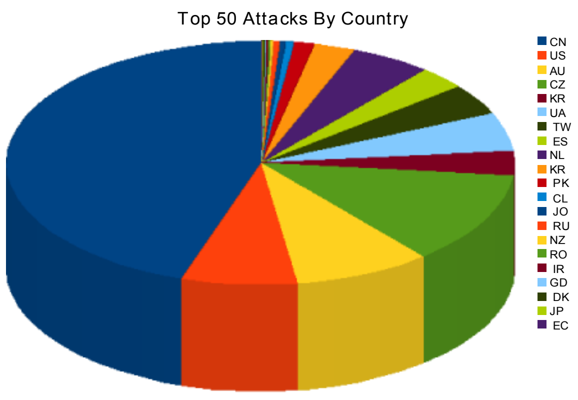

Figure 4: Top 50 Attackers by Country

V. Summary

A. Project Summary

The hardware and software used for this project has been summarized in Table 1. It can be inferred that standard hardware can easily be used to setup a virtual Honeynet. Large amounts of memory are always a preference for virtualized environments and can be used as a performance benchmark. The availability of free and Open Source tools and technologies such as Linux, VMware, Snort, Argus and those provided by the Honeynet Project [20] have considerably eased the process of setting up such projects. However, such tools demand a high degree of skill and customization, along with a thorough understanding of the system.

Since there is no out-of-the-box Honeynet solution available these free and Open Source tools serve as very flexible and robust toolset for security engineers. The security design, policy and architecture may vary from organization to organization, based on their implementation.

Project Summary		
Feature	**Product**	**Specs**
Host Operating System	Linux, Fedora 9	**HW Vendor:** Dell Optiplex 755 **Processor**: Intel(R) Core(TM)2 Duo CPU E6850 @ 3.00GHz **L2 Cache**: 4MB **RAM**: 4GB **Storage**: 250GB **NIC**: 1GB Ethernet controller (public IP)
Guest Operating System 1 (HONEYWALL)	Linux, Honeywall Roo 1.3	Single Processor Virtual Machine **RAM**: 512 MB **Storage**: 100 GB **NIC 1**: 100Mbps Bridged interface vmnet0 **NIC 2**: 100Mbps host-only interface vmnet1 **NIC 3**: 100Mbps Bridged interface vmnet2 (public IP)
Guest Operating System 2 (HONEYPOT)	Linux, Ubuntu 8.04 LTS (Hardy Heron)	Single Processor Virtual Machine **RAM**: 256 MB **Storage**: 20 GB **NIC**: 100Mbps host-only vmnet (public IP)
Virtualization software	VMware Server	VMware Server 1.0.6 for Linux x86
Architecture	Gen III	Gen III implemented as a virtual Honeynet
Honeywall	Roo	Roo 1.4
IDS	Snort	Snort 2.6.x
IPS	Snort _inline	Snort_inline 2.6.1.5
Data Capture Tool	Sebek	Sebek 3.2.0
Honeynet Project	Online Tenure	September 12, 2008 TO November 12, 2008

Table 1: Project Summary

We successfully setup and maintained this setup. A great deal was learned from this experience and certain areas for improvement have been identified. We have achieved a significant level of familiarity with all the tools utilized for the project and have identified some areas for further tool development and enhancement.

VI. Future Work

Our future work will incorporate running virtual Honeynets using other available Open Source virtualization software such as VMware ESX, Xen [26] and VirtualBox [27] and performing a comparative study of these tools in terms of running Honeynets efficiently. We believe that there is a great need for enhancing and automating the data capture mechanism. The current mechanism lacks correlation of network and host events. Automated Attacker profiling can be implemented in Sebek. We believe virtual Honeynets can serve as an excellent environment for development of an automated IDS and firewall signature engineering tool that can be used in collaboration with Snort. Finally, the Hflow database may serve as an excellent platform for building feature rich future security applications. We feel there is a dire need to develop better analysis tools to efficiently and effectively correlate attacks with attackers and suggest prevention techniques.

Acknowledgement

- Our thanks to Dr. Peter Komisarczuk who kindly provided the means and equipment to host the project at Victoria University of Wellington.

References:

1. Spitzner, L. (2002). Honeypots: Tracking Hackers. US: Addison Wesley. pp 1-430.
2. Stoll, C. The Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage. Pocket Books, New York, 1990
3. Cheswick, B. (1991). "An Evening with Berferd, in Which a Cracker Is Lured, Endured, and Studied." Forum of Incident Response and Security Teams (FIRST).
4. T H Project, Know your Enemy. Addison-Wesley, 2nd ed., 2004.
5. CERT Advisory CA-2001-31 Buffer Overflow in CDE Sub process Control Servicehttp://www.cert.org/advisories/CA-2001-31.html
6. Provos, N and Holz, T (July 26, 2007). Virtual Honeypots: From Botnet Tracking to Intrusion Detection. US: Addison-Wesley Professional.
7. Talabis, R. (2005). The Gen II & Gen III Honeynet Architecture. Available: http://www.philippinehoneynet.org/index2.php?option=com_docman&task=doc_view&gid=11&Itemid=29. Last accessed June, 2008.
8. William Stallings, "Cryptography and Network Security Principles and Practices", Third Edition, Prentice Hall, 2003.
9. Security architecture for open systems interconnection for CCITT applications, ITU-T, Study Group VII - Data Communications Networks, 1991
10. Snort user manual 2.8.3 , www.Snort.org
11. Know Your Enemy: Sebek, A kernel based data capture tool, The Honeynet Project, http://www.honeynet.org, Last Modified: 17 November 2003
12. Shuja, F. (October, 2006). Virtual Honeynet: Deploying Honeywall using VMware. Available: http://www.honeynet.pk/honeywall/index.htm. Last accessed June, 2008.
13. Robert McGrew, Rayford B. Vaughn, JR. Experiences With Honeypot Systems: Development, Deployment, and Analysis. Proceedings of the 39th Hawaii International Conference on System Sciences – 2006.
14. Levine.J, LaBella.R, Owen.H, Contis.D, Culver.B. (2003). The Use of Honeynets to Detect Exploited Systems. Proceedings of the 2003 IEEE. 3 (2),
15. "Argus project", 2004
16. Edward, B and Camilo, Viecco. (2005). Towards a Third Generation Data Capture Architecture for Honeynets. *Proceedings of the 2005 IEEE, Workshop on Information Assurance and Security, United States Military Academy, West Point, NY.* 1 (1), p21-28.
17. Snort, 2006, SNORT - The de facto standard on Intrusion Detection and Prevention, www.Snort.org
18. VMware. (2008). *VMware Server 1.0.6 Free.* Available: http://www.vmware.com/download/server/. Last accessed 20 Aug 2008.
19. VMware. (2006). *VMware Server Virtual Machine Guide.*Available: http://pubs.vmware.com/server1/wwhelp/wwhimpl/js/html/wwhelp.htm. Last accessed 2 August 2008.
20. "The Honyenet Project, 1999".
21. Duncan Napier. IPTables/NetFilter – Linux's next generation stateful packet filter. Sys Admin: The Journal for UNIX Systems Administrators, 10(12):8, 10, 12,14, 16, December 2001.
22. The Honeynet Project. (2005). *Know Your Enemy: Honeywall CDROM Roo.* Available: http://old.honeynet.org/papers/cdrom/Roo/index.html. Last accessed 5 May 2008.
23. V. N. Padmanabban and L. Subramanian. Determining the geographic location of Internet hosts. In SIGMETRICS/Performance, pages 324–325, 2001.
24. Google.com. (2009). Google Translate. Available: http://translate.google.com/. Last accessed 15 December 2008.
25. J. Oikarinen and D. Reed, "Internet Relay Chat Protocol RFC 1495," 1993.
26. Paul Barham , Boris Dragovic , Keir Fraser , Steven Hand , Tim Harris , Alex Ho , Rolf Neugebauer , Ian Pratt , Andrew Warfield, Xen and the art of virtualization, Proceedings of the nineteenth ACM symposium on Operating systems principles, October 19-22, 2003, Bolton Landing, NY, USA
27. VirtualBox. (2004). Sun VirtualBox® User Manual. Available: http://www.virtualbox.org/manual/UserManual.html. Last accessed 20 July 2008.