# Optimization of Virtual Honeynet with Implementation of Host Machine as Honeywall

***Thesis Report***

*Submitted in fulfillment of the requirements*

*For the award of degree of*

**Master of Engineering**

**in**

**Information Security**


*Submitted By*

**ROHITKUMAR GAUTAM**

**(Roll No. 801333022)**



COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

PATIALA – 147004


**July 2015**

# CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "*Optimization of Virtual Honeynet with Implementation of Host Machine as Honeywall*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Information Security* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Mr. Sanjeev Kumar* and *Dr. Jhilik Bhattacharya* and refers other researcher's work which are duly listed in the reference section. The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

(Rohitkumar Gautam)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Mr. Sanjeev Kumar)
Engineer, CSTD,
CDAC, Mohali

(Dr. Jhilik Bhattacharya)
Assistant Professor, CSED,
Thapar University, Patiala

**Countersigned by**

**(Dr. Deepak Garg)**
Head
Computer Science and Engineering Department
Thapar University
Patiala

**(Dr. S. S. Bhatia)**
Dean (Academic Affairs)
Thapar University
Patiala

## *ACKNOWLEGMENT*

I want to express my gratitude to **Dr. Jhilik Bhattacharya** who is my thesis co-guide, for giving me wonderful opportunity to work with him and supporting me throughout the research work. I would like to thank both **Dr. Jhilik Bhattacharya** my co-guide at Thapar University and **Dr. Deepak Garg**, Head of Department, CSED, Thapar University for giving me opportunity to work at Cyber Security Technology Division, CDAC, Mohali which is among the most reputed research labs in India.

I consider myself very fortunate to work under the expert guidance of my thesis co-guide **Mr. Sanjeev Kumar**, Engineer, Cyber Security Technology Division, CDAC Mohali. He has given me valuable guidance throughout this thesis work and he has been a person for me to look up to for research work. His ways of thinking and approaching a problem is very refreshing. One thing which has struck me the most is his ways of mentoring - without pressurizing he tries to bring the best out of his subordinates and creates inquiry and interest among them to look for solution to a problem. It has been a lifetime experience and I just want to thank him for allowing me to work under him.

My lab members at Cyber Security Technology Division, CDAC, Mohali have been very-very encouraging and always helpful. I want to thank Mr. Karan Kapoor, Mr. Kuwar Singh Bisht, Mr. Yogesh Rana and Mr. Ram Swaroop Misra for all little helps which they were always prepared to offer.

Last but not the least I want to thank the person whom I love and admire the most my parents and friends for always being there for me.

# *ABSTRACT*

A Honeynet System is widely deployed by the information security researchers for studying different types of attack pattern and methodology. It helps to determine best security practices & solution. It assists security researchers to conduct research in the field of cyber security. Easiest way of deploying honeynet system is by using open sourced honeywall Roo developed by honeynet project. Honeywall Roo can be implemented in Virtual Honeynet within virtual environment. Virtual machine based honeywall implementation do not take advantages of underlying architecture. It is also resource intensive and bulky. Hence new architecture has been proposed in this thesis work. In proposed architecture, host machine on which honeynet system has been deployed will act as honeywall and it has improved data control and capture mechanism based upon the type of honeypots. New enhanced architecture support hybrid honeypots framework for malware collection.

Implementation is specific to Linux based host machine having single network interface card deployed in distributed environment. Security of host machine is of utmost importance and hence special techniques are discussed to ensure its security and methods to mitigate security risk associated with host and virtual machines.

***Keywords- Honeypots, Virtual Honeynet, Honeywall, Honeynet gateway***

# TABLE OF CONTENT

# LIST OF FIGURE

# CHAPTER 1: INTRODUCTION

## 1.1 Overview

With the revolution of Information Technology, security of IT assets has become a matter of great concern. Security of application, services and network is of utmost importance. Traditional approach to handle cyber security concern was mainly based upon on defensive approach such as Intrusion Detection System (IDS), Data Encryption and Firewalls. In this approach, defending computer network and resource was based on detecting any failure in the defense and then reinforcing the compromised firewall and intrusion detection system by reacting to nature of failure [1-4].

But this approach has certain limitation as it is based upon limiting the interaction with intruder with the network resources. In tradition approach for cyber security, intruder has initiative as approach was purely based on defensive strategies.

Researcher turn to new approach and methods called "Honeynet" due to limitation of traditional approach. New approach using honeynet is based upon interacting with the intruder instead of restriction interaction with intruder. In case of honeynet, all interactions between honeypots and intruder are under surveillance and being logged. Honeynet is configured in such a way that intruder is not able to know that it's all activities with honeypot are being monitored and logged. Purpose of monitoring and logging all activities between honeypot and intruder is to gather as much information as possible about tactics, tools and attack pattern of intruder. Information gathered is used by security professional and researchers for generating IDS signature and hardening of system against future attacks [3-5].

## 1.2 What is Honeynet?

Honeynet is specially design architecture wherein networks of computers called honeypots having intentional vulnerability lies and all its activities are being monitored, controlled and captured. Honeynet consist of network of honeypots

designed with the purpose to attract hackers to perform attack on it and collect malware so that hacker's activities & attack tactics can be analyzed. Once intruder or attacker is inside the honeynet network all their activities such as file downloading, telnet request, FTP request, emails and HTTP request are captured without letting intruder know about it. In case attacker drops a malware inside honeypot then honeynet has a capability to securely collect dropped malware so that security researcher can perform malware analysis.

## 1.3 Components of Honeynet

Honeynet consist of two main components honeypot and honeywall.

### 1.3.1 Honeypots

Honeypots are the network resource whose true value lies in being probed, attacked, compromised and misused by a hacker to perform some malicious activities. Lance Spitzer defines Honeypots as "an information system whose value lies in unauthorized or illicit use of that resource" [6].

Usually Honeypots do not perform any activity by own on the system and hence it have no network traffic associate with them. Therefore, if any interactions with a honeypot are found then it is most likely an attempted probe, attack or compromise.

Honeypots are computer system used to lure an attacker by providing real and emulated environment of operating system, services and applications. Honeypots can provide real as well as emulated environment of operating system such as Windows XP, 7, Windows Server 2008, Redhat Linux, Ubuntu, etc and various kinds of services like mail server, web server, etc. Fig. 1 shows basic structure of honeynet system.

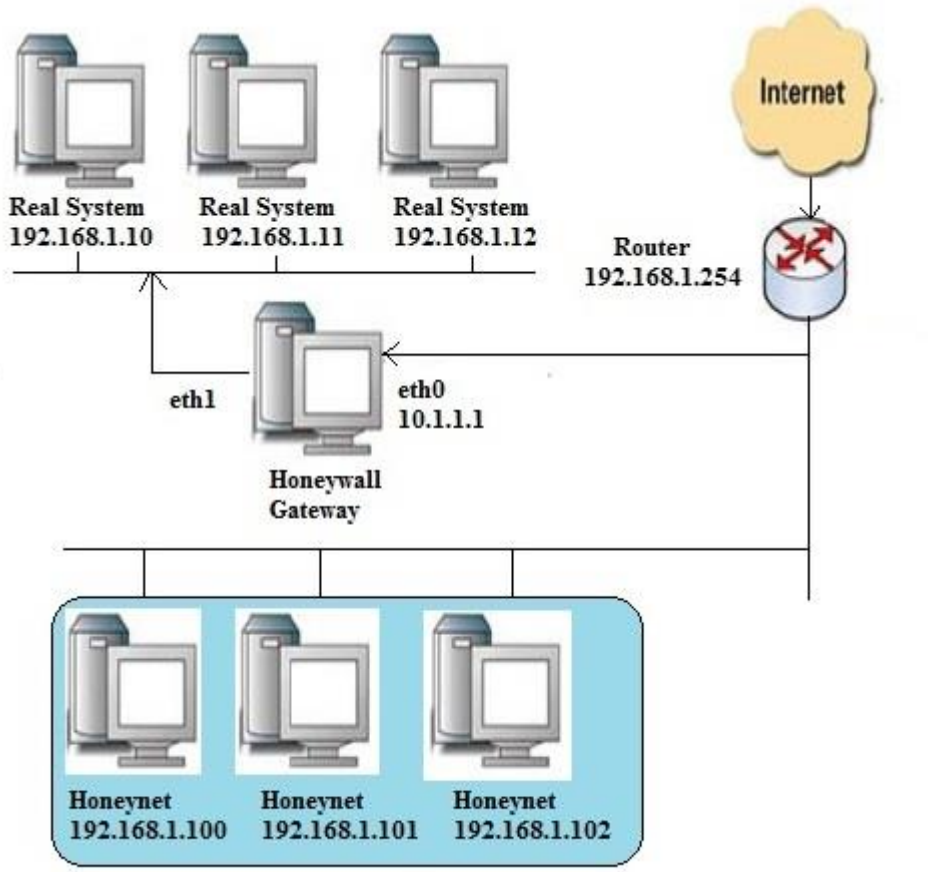


Figure 1.1 Basic Structure of Honeynet System

Honeypots are classified into two ways based upon type of interaction and how they are being deployed. Interaction means type of activity that an attacker may have with honeypot whereas way of deployment defines where they are deployed.

Firstly based upon the level of interaction between honeypots and intruder, honeypots are mainly classified as:

**a. Active Honeypot**

Active Honeypots has got its name because interaction with honeypots is initiated from honeypot itself. Active honeypots intentionally interact and open malicious URLs or files to collect malware samples. List of malicious or suspicious URLs are provide to active honeypot as input for processing. Active honeypot processed this list of URLs to collect malware samples [7].

### b. Passive Honeypot

Passive Honeypots are those honeypots which do not perform any activity or interaction on its own unlike active honeypots. In case of passive honeypots any kind of interaction is assumed to be done by intruder or attacker. Passive honeypots gather malware samples by luring the attackers. Passive honeypots provide real as well as emulated environment to lure an attacker [7].

Passive honeypots are further classified into two types depending upon type of environment they are providing to lure an attacker is:

#### i. Low Interaction Honeypot

Low Interaction Honeypots provide emulated environment resembling real looking system to lure an attacker. It emulated various types of vulnerable services, application and operating system rather than presenting real vulnerable systems. Level of interactions with intruder is restricted as emulated environment are provided to an attacker by low interaction honeypots.

Low interaction honeypots can emulate various kind of services such as mysql, telnet, ftp, webmail server, mail server, etc. as well as various kinds of hardware such as router, printer, etc.

When an attacker or intruder tries to interact with honeypot all its activities are being logged. This collected logged data are then used for further analysis by security professionals.

Example of well know low interaction honeypots are Nepenthes, Honeyd, Honeytrap, Specter and KFSensor [4].

### ii. High Interaction Honeypot

High interaction honeypots provide real environment to an intruder or attacker by running real service and operating system unlike emulated one in case of low interaction honeypot.

Deployment of high interaction honeypot required special attention as once honeypot get compromised it may be used as pawns to attack on other systems both on-site and off-site.

Advantage of high interaction honeypot is that it shows real pattern or environment that an intruder is looking for to hack. As high interaction honeypot provide real environment instead of emulated services level of interaction is high in case of high interaction honeypots. High interaction honeypots helps in gathering real time attack pattern as all its activities are captured and logged using various utilities within honeynet system. Cyber security researchers used this capture data to find malware sample and other relevant data that are not possible to gather by any other means [6-10].

Based upon the way honeypots are deployed and its usage in a network, honeypots are classified as:

### a. Production Honeypots

Production honeypots are deployed within organization's production network. The purpose of deployment of production honeypots is to detect any kind of intrusion within production network. Deployment of production honeypots in production network of an organization helps to minimize the overall risk as number of available options for an attacker gets increases. The main role of production honeypot is to detect any malicious activities within production network and based upon these activity it generate an alert for security professionals.

Generally emulated services are being offered on production server to lure an attacker by making attacker believe that the target computer an attacker is trying to compromise is real production server. Production honeypots are generally low interaction honeypots. They are developed and setup in such a way that they can

integrate with an organization's infrastructure. In production honeypots, security professionals can analyze limited behavior and activities of an attacker as most of the services are being emulated.

**b. Research Honeypots**

Research honeypots are usually deployed with sole purpose of research. It has comparatively complex structure. The main purpose of deployment of research honeypot is to gather as much information as possible about tools and techniques adopted by an attacker. In research honeypots usually real operating systems and services with potential vulnerability are being deployed instead of simulated or emulated services.

Research honeypots has great degree of freedom. There is also great risk associated with research honeypots as once honeypot system get compromised it can be used to further attack other systems within or outside network. Implementation of data control is more difficult in case of research honeypots due to its complex structure. Research honeypots are usually called honeynet and are deployed across multiple networks in different geographic area.

### 1.3.2 Honeywall

Honeywall is a transparent gateway behind which honeypots are being deployed. Honeywall act as transparent gateway and hence remain undetected by attackers. It is used to log all network activities within honeynet and capture all network data.

Honeywall is configured to achieve:

#### 1.3.2.1 Data Control

As honeypots are vulnerable machines there is potential risk associated with it. If honeypots get attacked and compromised by intruder, it may be used as pawn to attack other systems including non-honeynet. It may also abuse honeynet system

in some unexpected way. Too mitigate the risk associate with honeypots data control module is implemented to ensure that once honeypot get compromised, it cannot harm other systems [13-14].

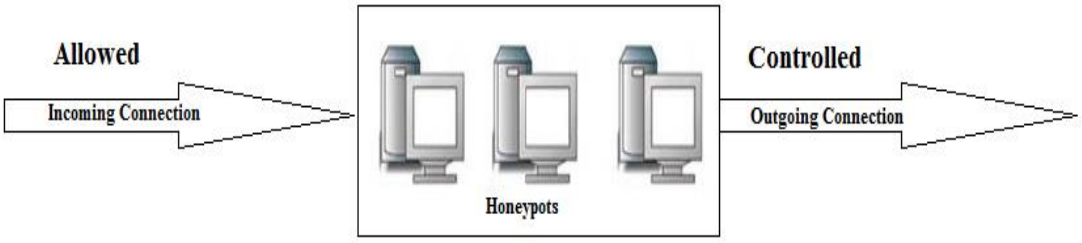


Figure 1.2 Data Control

Data control restricts the number of connections from honeypot to outer world while allowing all incoming connection towards honeypots. Data control module controlled the communication from honeypots. Data control allows limited connection from honeywall to centralized server in case of distributed honeynet system for forwarding logs and data to central repository.

While limiting connection from honeypots to outside, data control ensure that some degree of freedom are allowed to intruder so that some information can be gathered to learn more about the attack pattern and methodology.

#### 1.3.2.2 Data Capturing

Data capturing involves monitoring, capturing and logging of all activities that takes place within honeynet. [10]. Data capturing gathers all data and logs such as firewall alerts, IDS and IPS logs, captured network traffic data (pcap), etc.

Data capturing helps in knowing the tools, tactics & motive of attackers. To have more detailed information about attacker multiple layer of data capturing are implemented. Multiple layers of data capturing also help to minimize the risk of

failure in case honeypots get compromised. [14]. In case data capturing failed at any layer then combination of other layer help to understand the attacker's activity.

Generally captured data are not saved inside the honeypots as there is risk that honeypot might get detected and data may be deleted by an attacker.

#### 1.3.2.3 Data Collection

In distributed honeynet system, data collection plays an important role. Data captured using data capturing module are compressed and forwarded to host machine. After captured data are saved on host machine it is further compressed and securely forwarded to central data collection server where it stored as repository and used as and when required.

#### 1.3.2.4 Alert Generation

Honeywall collect all the logs and data related to honeypots and based upon the analysis of these logs it generate an alert if any suspicious activity takes place within honeynet. Alert make security professional aware about possible security breach in honeynet system.

Many well-known open source tools are used in Honeywall for data capture, control & collection and alert generation. Honeywall usually include following mentioned tools:

- **Snort-IDS**: Snort is used in honeywall in intrusion detection system mode to detect all kind of malicious activities within honeynet.
- **Snort-IPS**: Snort is used in inline mode as intrusion prevention system to filter all malicious outgoing connections and report any suspicious activity.
- **IPTABLES**: IPTABLES is used in honeywall for firewall implementation and data control.

- **TCPDUMP**: TCPDUMP is used in honeywall to capture and view network traffic.

- **p0f**: It is used for passive operating system fingerprinting.

# CHAPTER 2: LITERATURE REVIEW

## 2.1 Overview

Honeynet system is the new approach towards cyber security in which some degree of freedom is provided to an attacker so that more information can be collected about attack pattern. With the passage of time architecture or generation of honeynet has been updated based upon the technology implemented and way of implementation of data control, capture & collection within honeynet.

## 2.2 Generation of Honeynet

Over the years, honeynet system has evolved three generation or architecture as outlined below:

### 2.2.1 Generation I

The first phase of honeynet began in 1999 with aim to develop, deploy, and test honeynet technologies and their ability to gather information on attackers. These first phase of honeynet is called Gen I honeypot. They were simple and easy to deploy and depended on basic mechanism to control attackers. They had no sophisticated methodology to collect encrypted network activities though they captured most of automated attacks successfully. They also have early warning and prediction [15].

In Generation I architecture, two interface were required on honeywall gateway, one connecting external network and other facing internal network. Drawback of this architecture is that gateway act as layer 3 device and therefore could be easily detected by the attacker [15-16].

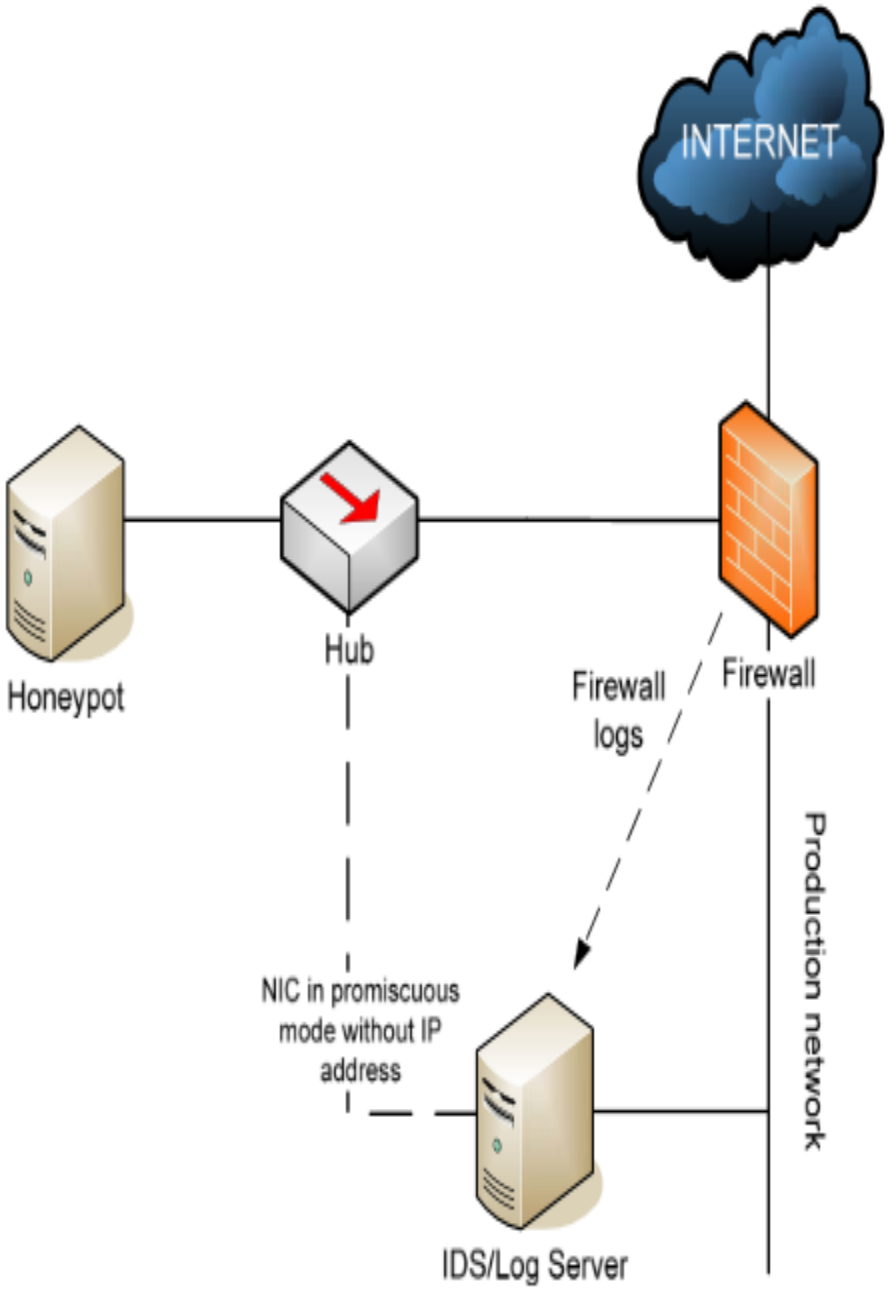


Figure 2.1 Gen I honeynet architecture

.

### 2.2.2 Generation II

Work on Gen II honeynet architecture began in 2002 with aim to improve and simplify honeynet capabilities. Generation II honeynet had more advanced method to monitor and control attacker's activities within honeynet system. New device called

IDS gateway or the Honeywall had been introduced to handles data control and data capture mechanism of honeynet. Honeywall has been implemented as transparent bridge. Usage of data control minimizes the risk of attacker detecting the honeynet and harming other systems within network [15].

Gen II honeynet provides high level of interaction to the attacker. Due to this high level of interaction there was great security risk and hence advanced methods of data control and capturing were implemented.

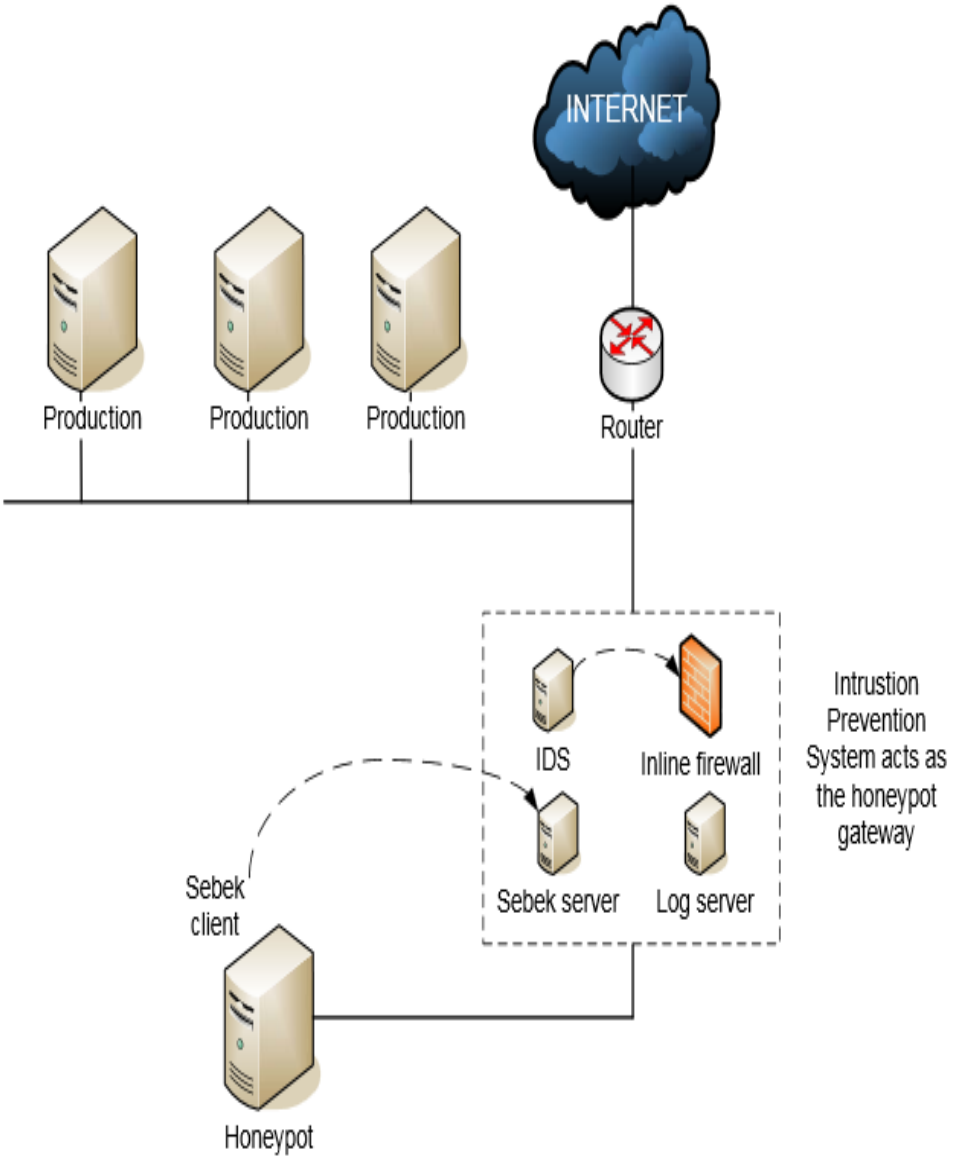


Figure 2.2 Gen II honeynet architecture

### 2.2.3 Generation III

Gen III honeynet has been developed with aim to create easy to deploy bootable CD-ROM so that organizations simply boot the CD-ROM which will act as honeynet gateway or honeywall, deploys all required tools and configure honeynet to log all captured activities to a central server.

Architecture of Gen III honeynet is similar to Gen II with improvement in deployment and management capability. Sebek server was also added in Gen III honeynet at honeywall [15-16].

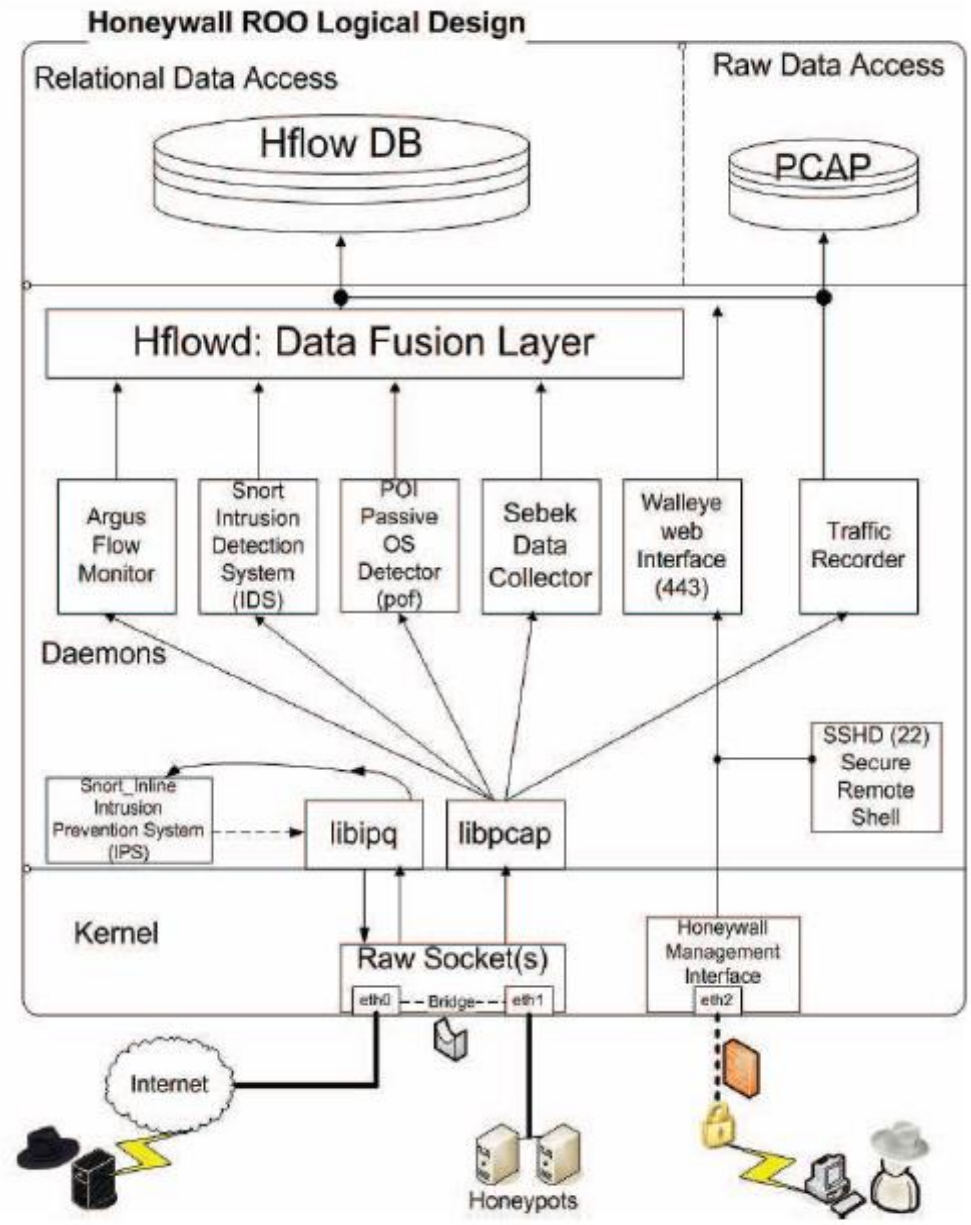


Figure 2.3 Gen III Honeywall Roo Logical Design

Honeywall Roo

Honeywall Roo is ready to deploy bootable CDROM developed as part of Gen III architecture. It was implemented in CentOS operating system. Honeywall Roo automates the installation and maintenance of honeynet [17].

### 2.2.4 Virtual Honeynet

Virtualization is a technology that allows running multiple virtual machine on a single physical machine such that each virtual machine act as independent operating system installation & run concurrently with others. Virtualization is achieved by sharing physical resources such as CPU, memory, storage, etc through specialized software. Virtualization helps to reduce project costs [15,18].

A virtual honeynet is a complete honeynet running on single physical machine in such a way that components of honeynet such as honeypots and honeywall are running in a virtual environment using virtualization tools like VMware, VirtualBox, etc. Virtual machines are machine that run on host machine in virtual environment with the appearance that they are the sole occupant of underlying hardware.

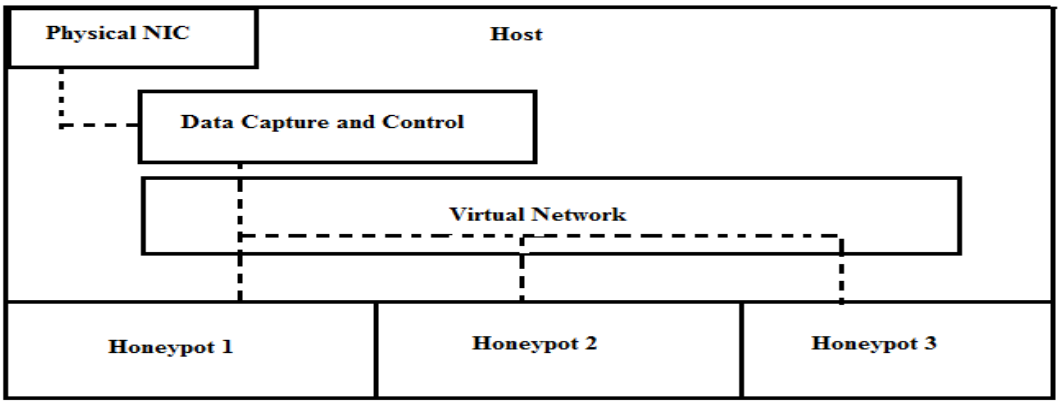


Figure 2.4 Classical Virtual Honeynet Architecture

In virtual honeynet, all systems including honeywall and honeypots are hosted on a single machine within virtual environment. As shown in Figure 2.4, the overall architecture of virtual honeynet is similar to Gen II honeynet except it is hosted on single physical machine.

Architecture of traditional virtual honeynet having virtual machine based honeywall does not take advantage of underlying architecture [19].

Tools and utilities used in virtual honeywall Roo for implementation of virtual honeynet are [16]:

- Hflow-snort: Intrusion Detection System
- Hflow-snort_inline: Intrusion Prevention System
- Hflow-pcap: Uses tcpdump utility for capturing network traffic.
- Hwdaemon: It is used to manage utilities like firewall, IDPS, tcpdump.
- Rc.firewall: To automate firewall implementation.
- Hwctl and Dialog Menu: Utility that allows administrator to configure system variables used by various programs.

# CHAPTER 3: PROBLEM STATEMENT

Virtual Honeynet system though utilized the single physical machine for its implementation, but cannot take advantages of underlying architecture. During our literature survey, it is found that running virtual machine based honeywall such as Roo in virtual honeynet is a feasible solution and it needs improvement.

Virtual machine based honeywall being a full-fledged operating system running in virtual environment unnecessarily utilizes available physical resources like CPU, memory, storage, etc. As virtual machine based honeywall is implemented in virtual environment, it needs to transfer all its log and data to host machine or physical machine. This regular transfer of logs and data from virtual machine to host machine within same physical machine create unrequired load on physical resources such as CPU, memory, etc.

With passage of time, if logs and data collected in virtual machine based honeywall is not transfer to host machine from virtual machine then storage space in virtual machine based honeywall get fully filled. Once this happen virtual machine based honeywall do not perform its activity properly.

During implementation it is also found that virtual machine based honeywall do not support different types of honeypots such as active honeypots, router honeypots, passive honeypots, web based honeypots, etc. There is also no provision for disk imaging of virtual honeypots in virtual machine based honeywall.

Research problem of this thesis work is to design host machine based honeywall for virtual honeynet which will take advantage of underlying architecture and support different types of honeypots. It will have better network traffic management with respect to different types of honeypots. It will supports disk imaging & forensic analysis of honeypot to find malware samples. Such host machine based honeywall will optimize virtual honeynet.

# CHAPTER 4: IMPLEMENTATION

## 4.1 Introduction

In the following discussion, Virtual Box is used for hosting honeypots similar to traditional virtual honeynet whereas honeywall is implemented on host machine. Traditionally for implementation of virtual honeynet, honeywall is implemented on separate virtual machine to monitor and logs traffic passing to and fro. In our implementation, virtual honeynet architecture has been enhanced by removing virtual machine based honeywall and implementing data control and capture on host machine. For optimization of virtual honeynet, better network traffic management and data capture & control is implemented based upon type of honeypots.

## 4.2 Network Architecture

Figure 4.1 represent the network architecture of implemented optimized virtual honeynet with host machine acting as honeywall whereas all honeypots are implemented in virtual machine.

As shown in the Figure 4.1, host machine act as honeywall controlling all incoming and outgoing network traffic. In this implementation, host machine has the main role being functioning as honeywall.

For better management of network traffic based upon type of honeypots different network taps are created. Different data control and capturing rules are implemented on these network taps based upon honeypot type. For example, in case of active honeypots outgoing dns request and incoming dns response are allowed but it is not the same in case of router honeypot. For same type of honeypot same network taps are used.

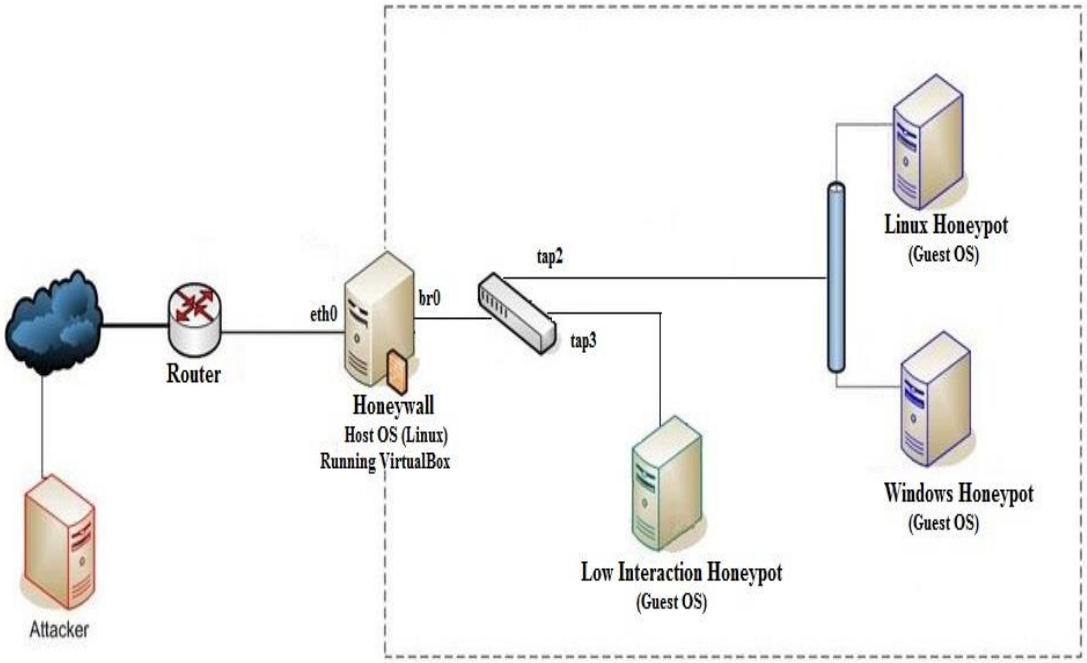


Figure 4.1 Network Architecture of Optimized Virtual Honeynet

## 4.3 Architectural Enhancement and Optimization

In our implementation architectural enhancement has been done by implementing host machine as honeywall. In enhanced architecture host machine performs the various task such as data controlling, data capturing, data collection and alert generation. Our proposed architecture takes the advantage of underlying architecture and controls the honeypot machine. In our implementation, host machine based honeywall can shut down and restart any honeypots as desired or at interval of specific time to mimic the real system environment of computer lab.

In our implementation of host machine based honeywall for virtual honeynet, all logs and data are directly saved on host machine and hence save time and unnecessary load on transferring it form virtual machine to host machine as in cased of traditional virtual honeynet. In our optimized virtual honeynet, it displays the real time logs and statistics on host machine.

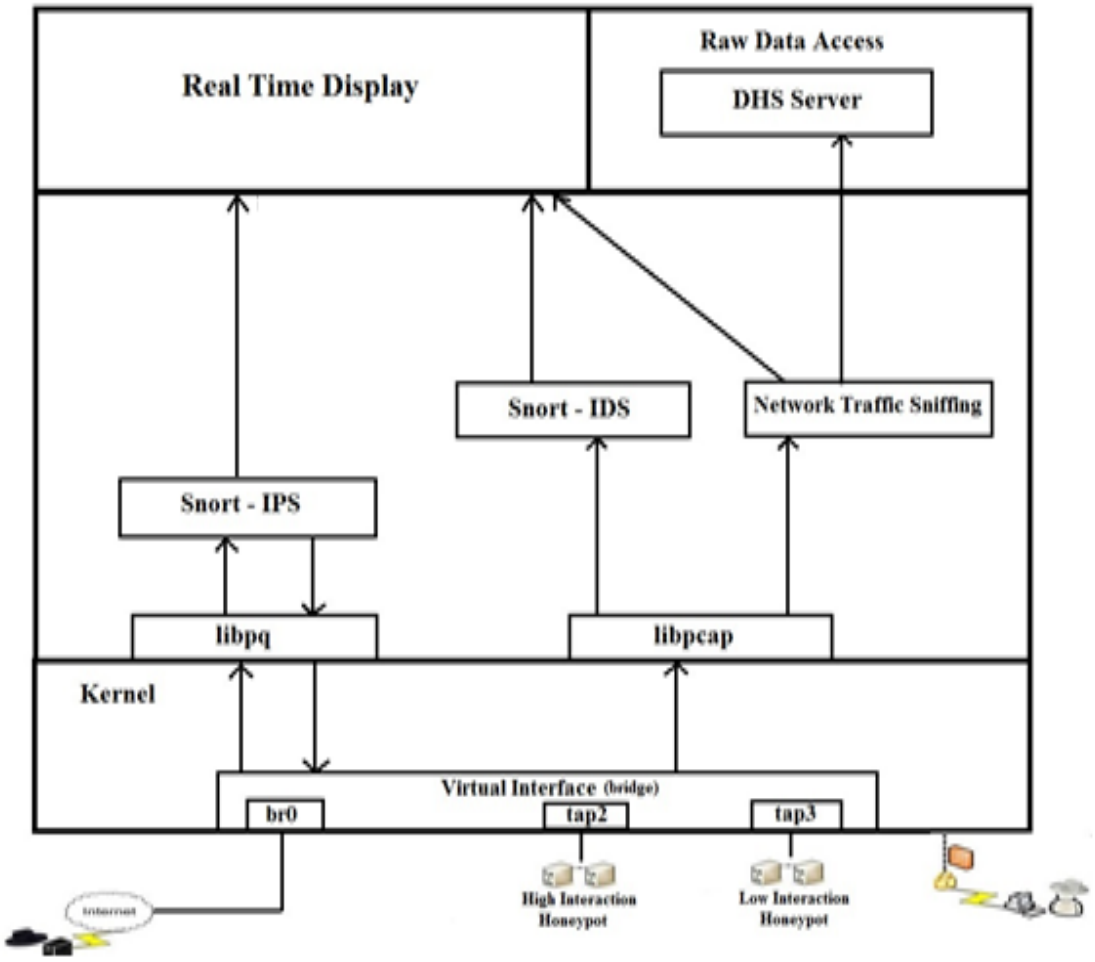


Figure 4.2 Architecture of host machine based honeywall

Figure 4.2 show architecture of host machine based honeywall. It used many tools and utilities like iptables, snort, tcpdump, etc. Our implementation work also supports disk imaging and honeypot image forensic analysis by taking backup of virtual hard disk. This helps in collection of malware samples saved on virtual hard disk.

## 4.4 Tools

For our implementation of virtual honeynet having host machine as honeywall, open source tools and utilities are used. These tools helps in functioning of data control and data capturing module of host based virtual machine.

- **Snort-IDS**: Snort is configured to work in Intrusion Detection System mode to parse all incoming network traffic and detect any malicious activity on the physical interface of host machine i.e. honeywall
- **Snort-IPS**: Snort is configured to work in Intrusion Detection & Prevention mode to parse all outgoing connections. Snort-IPS is implemented in inline mode using queue to filter all the outgoing connection and report any suspicious activity originating from honeypot.
- **IPTABLES**: IPTABLES are used for configuring firewall rules and implementing connection limiting on incoming connections while allowing controlled outbound connections originating from honeypots. IPTABLES is configured in such a way that only limited connection can be possible between host machine based honeywall and central data collection server.
- **TCPDUMP**: TCPDUMP is used to capture all the network traffic of honeypots.
- **Diff**: Diff is used to find malware samples by finding difference between two honeypot's virtual disk image i.e. live honeypot's virtual disk image and original honeypot virtual disk image saved as repository on honeywall.

## 4.5 Advantages

Our proposed architectural enhancement of virtual honeynet by implementing host machine based honeywall has following advantages:

- Optimum utilization of available physical resources. Resource allotted to run full-fledged operating system in virtual machine has been minimized by

removing virtual machine based honeywall. Hence load on CPU, memory, storage, etc has been significantly reduced

- All logs and data are directly saved on host machine. Hence there is no need to transfer logs and captured data form virtual machine to host machine.
- Host machine based honeywall display real time information based upon saved logs and data
- Forward the collected logs and data securely and in compressed form to centralized server in Distributed Honeynet Environment so that it can be easily available whenever required.
- Supports disk imaging and honeypot image forensic analysis by taking backup of honeypot's disk image and then comparing it will disk image saved in repository on honeywall.
- Use different network taps for better management of network traffic based upon types of honeypot
- Supports different types of honeypots such as active honeypots, passive honeypot, router honeypot, etc
- Automatically shut down and restarts honeypots after certain interval of time to mimic the real system environment.

# CHAPTER 5: EXPERIMENTAL RESULTS

## 5.1 Experimental Setup

We have designed and implemented virtual honeynet on a server running Redhat Enterprise Linux over Intel i5-4rth generation processor and 4 GB of RAM. Virtual honeynet has been deployed in such a way that host machine will act as honeywall.

Data control, capturing & collection are implemented on host machine based honeywall. Alert generation module has also been implemented on host machine based honeywall. Networking has been implemented in bridged mode to avoid detection of honeynet. Network taps has been created for better management of network traffic based upon the types of honeypots. Physical interface of host machine based honeywall has no network address. Designed virtual honeynet supports both active and passive honeypots.

Low interaction honeypots are configured using honeyd and diaonea which provide emulated services hardware and application. In case of high interaction honeypots actual operating systems with service has been installed in virtual machine using virtual box.

## 5.2 Stability and Performance

Total number of four honeypots running and emulating various operating systems such as windows XP & 7, Linux was implemented on optimized virtual honeynet successfully.

To test the stability and performance of virtual honeynet, virtual honeynet has been implemented on two physical machines, one having virtual machine based honeywall Roo and other having host machine based honeywall. Figure 5.1 shows the CPU load generated on host machine in both scenario i.e. virtual machine based honeywall and host machine based honeywall.

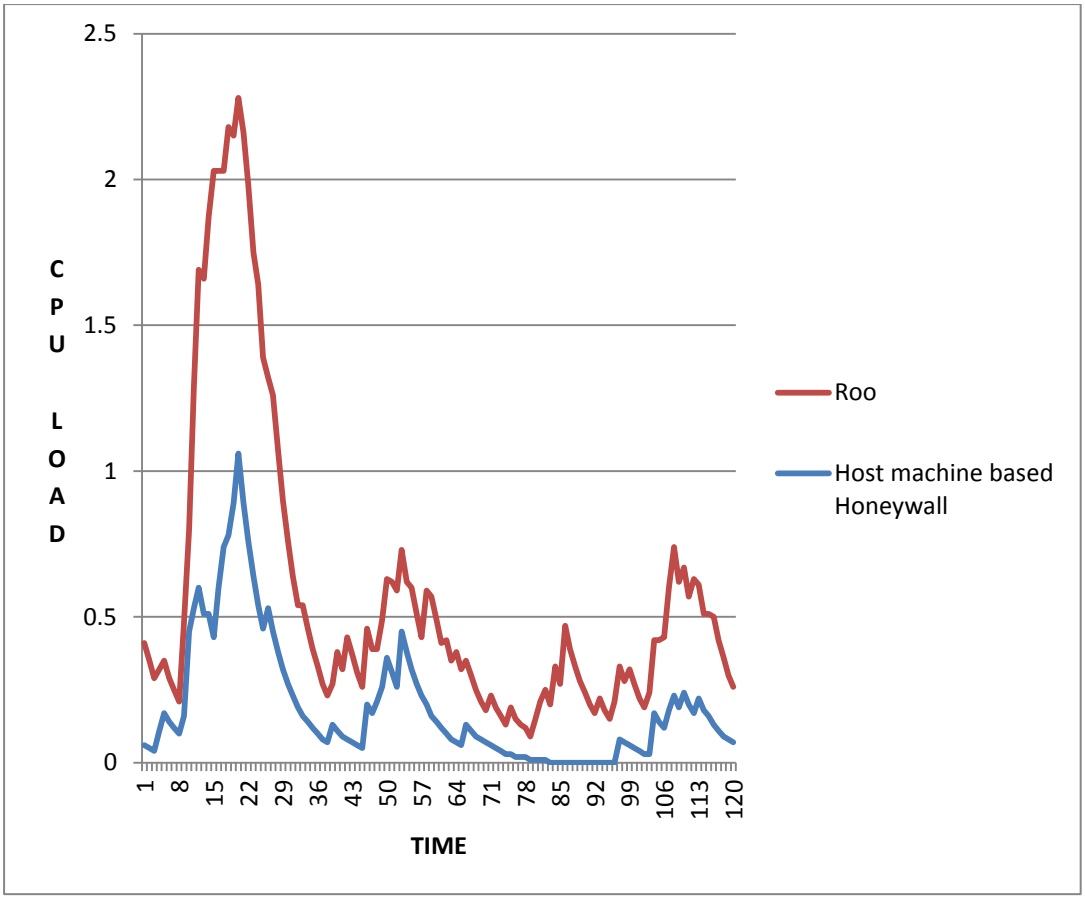


Figure 5.1 Load generated on CPU by host machine based honeywall and Roo Honeywall

As shown in the Figure 5.1, load generated on CPU by honeywall Roo is more as compare to load generated by host machine based honeywall. During running both systems for particular time interval it is found excess load is generated when honeypots get power on and during disk imaging which cause Roo Honeywall to become unstable while host machine based honeywall comparatively works fine.

During our implementation work, we also took reading of load generated on RAM in case Roo honeywall and host machine based honeywall. Figure 5.2 show comparative study of load generated on RAM based upon memory consumption.

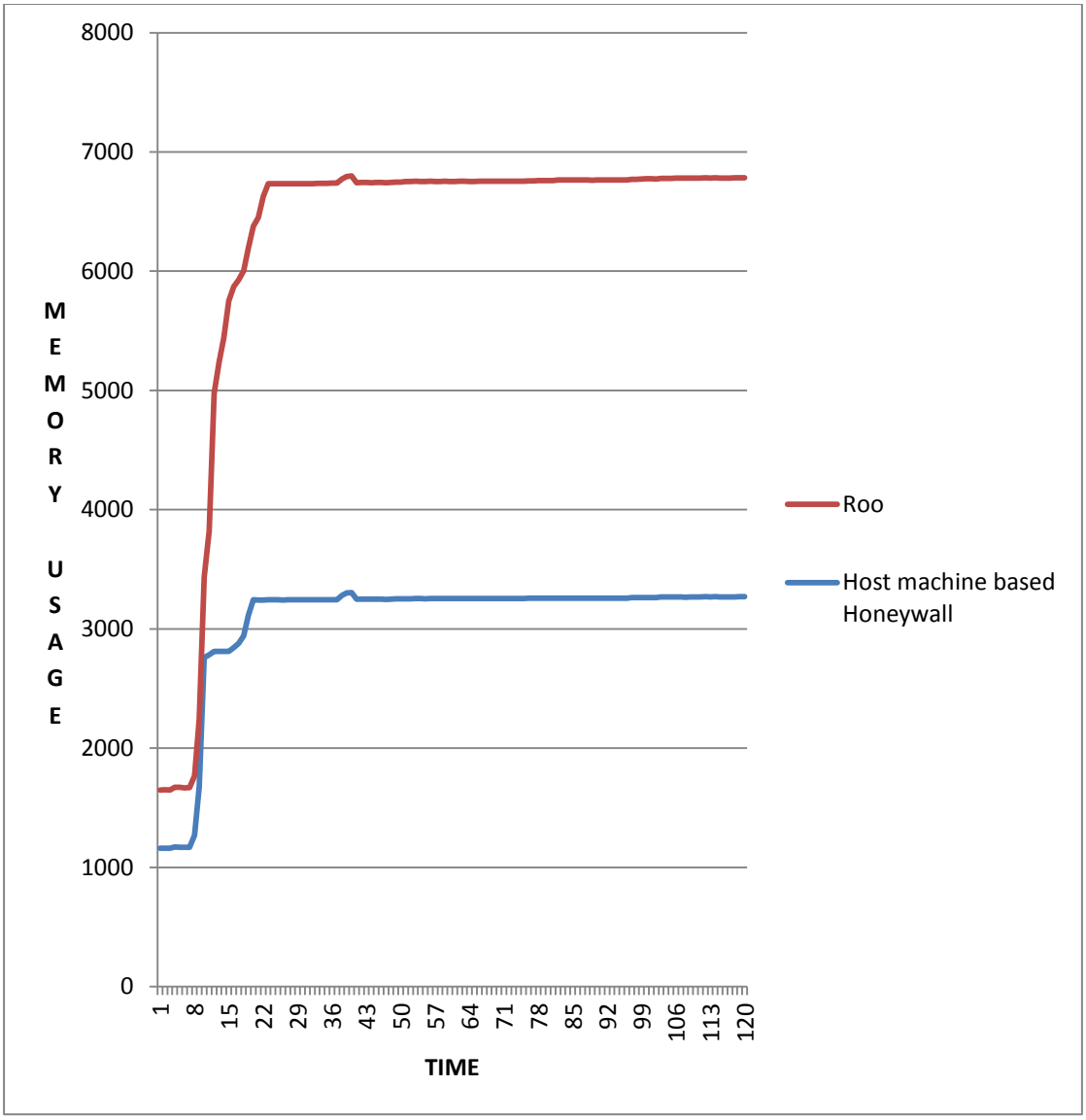


Figure 5.2 Comparison of Memory consumption by host machine based honeywall and Roo honeywall

Figure 5.2 clearly shows that memory consumption during honeywall Roo is comparatively more than host machine based honeywall while running same number of honeypots simultaneously.

Comparison of CPU load and memory consumption clearly shows that Roo honeywall based virtual honeynet takes more resources as compare to virtual honeynet with host machine based honeywall. Results show that if new architecture is used for virtual honeynet instead of traditional virtual honeynet, then resource used

for running full-fledged operating system for Roo honeywall can be utilized for running one more honeypot.

## 5.3 OS Fingerprinting

Nmap tool is used to test whether the honeypots is working properly or not in new architecture. Port scan has been performed for open ports using nmap and checked for whether honeypot respond properly to incoming probe request with appropriate response.

Result of nmap on one of honeypot running windows & system can be seen here:

*Starting Nmap 6.47 ( http://nmap.org ) at 2015-06-06 09:29 India Standard Time*

*Nmap scan report for 192.168.8.124*

*Host is up (0.00035s latency).*

*Not shown: 990 closed ports*

*PORT STATE SERVICE VERSION*

*135/tcp open msrpc Microsoft Windows RPC*

*139/tcp open netbios-ssn*

*445/tcp open netbios-ssn*

*5357/tcp open http Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)*

*|_http-methods: No Allow or Public header in OPTIONS response (status code 503)*

*|_http-title: Service Unavailable*

*49152/tcp open msrpc Microsoft Windows RPC*

*49153/tcp open msrpc Microsoft Windows RPC*

*49154/tcp open msrpc Microsoft Windows RPC*

*49155/tcp open msrpc Microsoft Windows RPC*

*49156/tcp open msrpc Microsoft Windows RPC*

*49158/tcp open  msrpc  Microsoft Windows RPC*

*MAC Address: 08:00:27:78:71:47 (Cadmus Computer Systems)*

*Device type: general purpose*

*Running: Microsoft Windows 7|2008*

*OS CPE: cpe:/o:microsoft:windows_7::- cpe:/o:microsoft:windows_7::sp1 cpe:/o:microsoft:windows_server_2008::sp1 cpe:/o:microsoft:windows_8*

*OS details: Microsoft Windows 7 SP0 - SP1, Windows Server 2008 SP1, or Windows 8*

*Network Distance: 1 hop*

*Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows*

*Host script results:*

*|_nbstat: NetBIOS name: MASTER-PC, NetBIOS user: <unknown>, NetBIOS MAC: 08:00:27:78:71:47 (Cadmus Computer Systems)*

*| smb-os-discovery:*

*|  OS: Windows 7 Ultimate 7600 (Windows 7 Ultimate 6.1)*

*|  OS CPE: cpe:/o:microsoft:windows_7::-*

*|  Computer name: master-PC*

*|  NetBIOS computer name: MASTER-PC*

*|  Workgroup: WORKGROUP*

*|_  System time: 2015-07-06T09:30:57+05:30*

*| smb-security-mode:*

*|  Account that was used for smb scripts: guest*

*|  User-level authentication*

*|  SMB Security: Challenge/response passwords supported*

*|_  Message signing disabled (dangerous, but default)*

*|_smbv2-enabled: Server supports SMBv2 protocol*

*TRACEROUTE*

*HOP RTT  ADDRESS*

*1 0.35 ms 192.168.8.124*

*OS and Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .*

*Nmap done: 1 IP address (1 host up) scanned in 102.69 seconds*

An example of emulated windows XP port configuration using honeyd to respond with windows XP's fingerprint:

*create windows*

*set windows personality "Microsoft Windows XP Professional SP1"*

*set windows tcp port 21 "sh /usr/share/honeyd/scripts/unix/linux/ftp.sh"*

*add windows tcp port 135 open*

*add windows tcp port 139 open*

*add windows tcp port 445 open*

*set windows default tcp action accept*

*set windows default udp action accept*

*set windows uptime 1332645*

*set windows Ethernet "00:00:24:bc:8d:40"*

*bind 192.168.1.15 Windows*

## 5.4 IPTABLES LOG

Data controlling has been implemented with the help of iptables. Iptables is used to configure firewall to allow all incoming connection while control outgoing connection based upon honeypot's type. To control outgoing connection from passive honeypots, firewall is implemented such that it allows specific number of TCP, UDP, ICMP, etc connection. This number of outgoing connection is determined by administrator of honeynet.

Iptables logs showing incoming connection:

Apr 26 18:42:58 localhost kernel: INBOUND CONNECTION IN=br0 OUT=br0 PHYSIN=eth0 PHYSOUT=tap2 SRC=118.244.136.200 DST=192.168.8.124 LEN=40 TOS=0x00 PREC=0x00 TTL=103 ID=256 PROTO=TCP SPT=3578 DPT=3389 WINDOW=16384 RES=0x00 SYN URGP=0

Apr 26 18:42:58 localhost kernel: INBOUND CONNECTION IN=br0 OUT=br0 PHYSIN=eth0 PHYSOUT=tap2 SRC=118.244.136.200 DST=192.168.8.124 LEN=40 TOS=0x00 PREC=0x00 TTL=103 ID=256 PROTO=TCP SPT=12757 DPT=3389 WINDOW=16384 RES=0x00 SYN URGP=0

Apr 26 18:46:36 localhost kernel: INBOUND CONNECTION IN=br0 OUT=br0 PHYSIN=eth0 PHYSOUT=tap2 SRC=199.217.113.243 DST=192.168.8.124 LEN=445 TOS=0x00 PREC=0x00 TTL=54 ID=0 DF PROTO=UDP SPT=5232 DPT=5060 LEN=425

Apr 26 18:46:45 localhost kernel: INBOUND CONNECTION IN=br0 OUT=br0 PHYSIN=eth0 PHYSOUT=tap2 SRC=61.153.104.38 DST=192.168.8.124 LEN=40 TOS=0x00 PREC=0x00 TTL=102 ID=256 PROTO=TCP SPT=6000 DPT=1433 WINDOW=16384 RES=0x00 SYN URGP=0

Iptables log showing connection limiting based upon type of connection:

ICMP Connection

Apr 30 16:04:24 localhost kernel: OUTBOUND ICMP: IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=eth0 SRC=192.168.8.124 DST=207.241.227.207 LEN=60 TOS=0x00 PREC=0x00 TTL=128 ID=3406 PROTO=ICMP TYPE=8 CODE=0 ID=512 SEQ=2048

Apr 30 16:04:24 localhost kernel: OUTBOUND ICMP: IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=eth0 SRC=192.168.8.124 DST=38.91.54.14 LEN=60 TOS=0x00 PREC=0x00 TTL=128 ID=3407 PROTO=ICMP TYPE=8 CODE=0 ID=512 SEQ=2304

Apr 30 16:04:24 localhost kernel: OUTBOUND ICMP: IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=eth0 SRC=192.168.8.124 DST=201.49.160.181 LEN=60 TOS=0x00 PREC=0x00 TTL=128 ID=3408 PROTO=ICMP TYPE=8 CODE=0 ID=512 SEQ=2560

Apr 30 16:04:24 localhost kernel: DROP:ICMP > 10 Limit IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=eth0 SRC=192.168.8.124 DST=91.178.132.155 LEN=60 TOS=0x00 PREC=0x00 TTL=128 ID=3409 PROTO=ICMP TYPE=8 CODE=0 ID=512 SEQ=2816

TCP Connection

Apr 29 07:27:58 localhost kernel: OUTBOUND TCP: IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=eth0 SRC=192.168.8.124 DST=23.3.98.11 LEN=48 TOS=0x00 PREC=0x00 TTL=128 ID=186 DF PROTO=TCP SPT=1073 DPT=80 WINDOW=65535 RES=0x00 SYN URGP=0

Apr 29 07:28:01 localhost kernel: OUTBOUND TCP: IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=eth0 SRC=192.168.8.124 DST=156.151.59.19 LEN=48 TOS=0x00 PREC=0x00 TTL=128 ID=192 DF PROTO=TCP SPT=1074 DPT=80 WINDOW=65535 RES=0x00 SYN URGP=0

Apr 29 07:28:01 localhost kernel: OUTBOUND TCP: IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=eth0 SRC=192.168.8.124 DST=23.3.98.11 LEN=48 TOS=0x00 PREC=0x00 TTL=128 ID=198 DF PROTO=TCP SPT=1075 DPT=80 WINDOW=65535 RES=0x00 SYN URGP=0

Apr 29 07:28:03 localhost kernel: DROP:TCP > 10 Limit IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=eth0 SRC=192.168.8.124 DST=63.233.110.83 LEN=48 TOS=0x00 PREC=0x00 TTL=128 ID=212 DF PROTO=TCP SPT=1076 DPT=80 WINDOW=65535 RES=0x00 SYN URGP=0

UDP Connection

Apr 28 04:09:25 localhost kernel: OUTBOUND UDP: IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=tap3 SRC=192.168.8.124 DST=192.168.8.8 LEN=202 TOS=0x00 PREC=0x00 TTL=128 ID=5925 PROTO=UDP SPT=138 DPT=138 LEN=182

Apr 28 04:09:25 localhost kernel: OUTBOUND UDP: IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=eth0 SRC=192.168.8.124 DST=192.168.8.8 LEN=202 TOS=0x00 PREC=0x00 TTL=128 ID=5925 PROTO=UDP SPT=138 DPT=138 LEN=182

Apr 28 04:21:40 localhost kernel: OUTBOUND UDP: IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=eth0 SRC=192.168.8.124 DST=192.168.8.8 LEN=202 TOS=0x00 PREC=0x00 TTL=128 ID=5968 PROTO=UDP SPT=138 DPT=138 LEN=182

Apr 28 04:21:40 localhost kernel: DROP:UDP > 10 Limit IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=tap3 SRC=192.168.8.124 DST=192.168.8.8 LEN=78 TOS=0x00 PREC=0x00 TTL=128 ID=5969 PROTO=UDP SPT=137 DPT=137 LEN=58

Other Connection

Apr 28 04:01:01 localhost kernel: OUTBOUND OTHER: IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=tap3 SRC=192.168.8.124 DST=239.255.255.250 LEN=32 TOS=0x00 PREC=0x00 TTL=1 ID=5909 PROTO=2

Apr 28 04:06:03 localhost kernel: OUTBOUND OTHER: IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=tap3 SRC=192.168.8.124 DST=239.255.255.250 LEN=32 TOS=0x00 PREC=0x00 TTL=1 ID=5909 PROTO=2

Apr 28 04:07:04 localhost kernel: OUTBOUND OTHER: IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=tap3 SRC=192.168.8.124 DST=239.255.255.250 LEN=32 TOS=0x00 PREC=0x00 TTL=1 ID=5909 PROTO=2

Apr 28 04:09:30 localhost kernel: DROP:Other Rate > 10 Limit IN=br0 OUT=br0 PHYSIN=tap2 PHYSOUT=tap3 SRC=192.168.8.124 DST=224.0.0.9 LEN=32 TOS=0x00 PREC=0x00 TTL=1 ID=5931 PROTO=2

## 5.5 Snort Logs

Snort has been configured to work in IDS mode for incoming connections and IPS mode for outgoing connections.

Logs generated by Snort engine:

05/06-06:25:30.719834 [**] [123:8:1] (spp_frag3) Fragmentation overlap [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {UDP} 192.168.8.124 -> 104.255.65.232

05/06-06:36:27.044661 [**] [1:2010935:2] ET POLICY Suspicious inbound to MSSQL port 1433 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 116.255.160.245:6000 -> 192.168.8.124:1433

05/06-06:43:15.173097 [**] [1:2010936:2] ET POLICY Suspicious inbound to Oracle SQL port 1521 [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 222.186.21.201:60238 -> 192.168.8.124:1521

05/06-07:55:24.034856 [**] [1:2008578:6] ET SCAN Sipvicious Scan [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP} 178.162.201.166:57102 -> 192.168.8.124:5060

05/06-07:55:24.045724 [**] [1:2011716:4] ET SCAN Sipvicious User-Agent Detected (friendly-scanner) [**] [Classification: Attempted Information Leak] [Priority: 2] {UDP} 178.162.201.166:57102 -> 192.168.8.124:5060

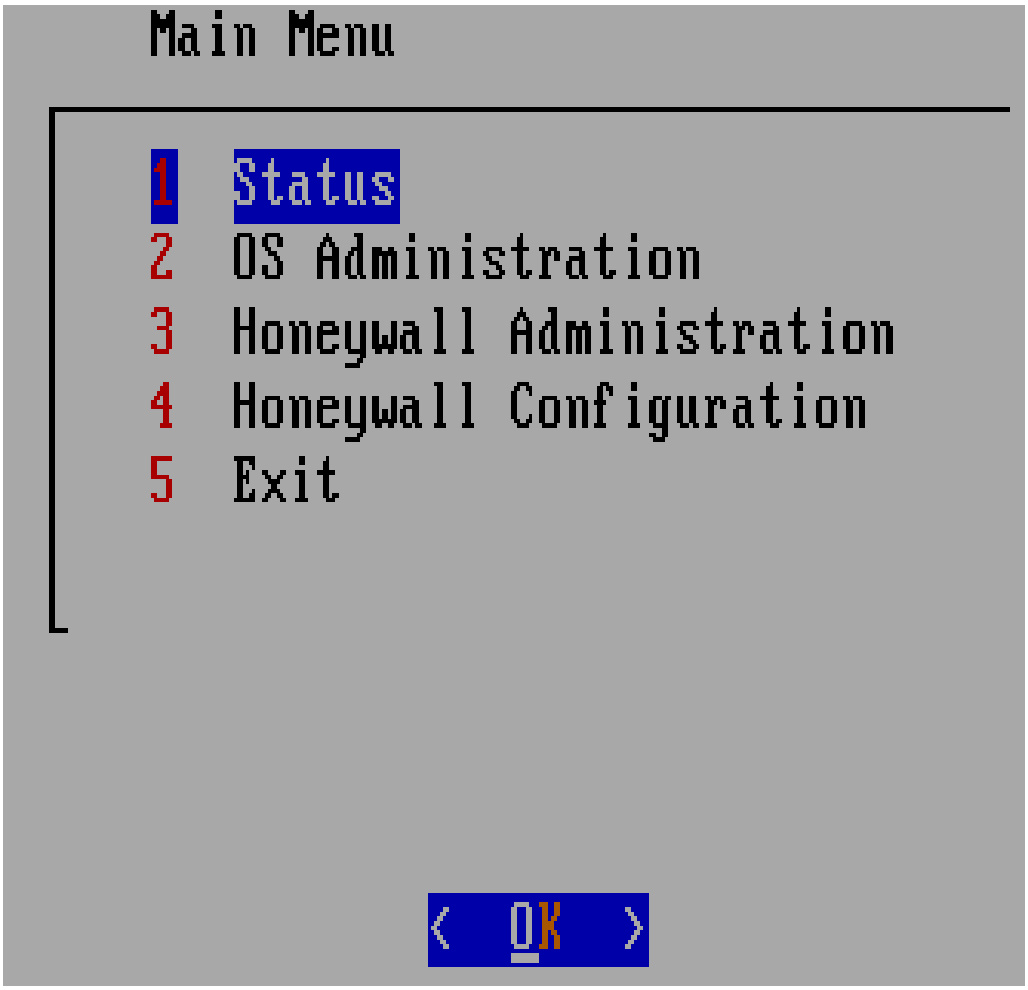


Figure 6.1 Honeywall Main Menu

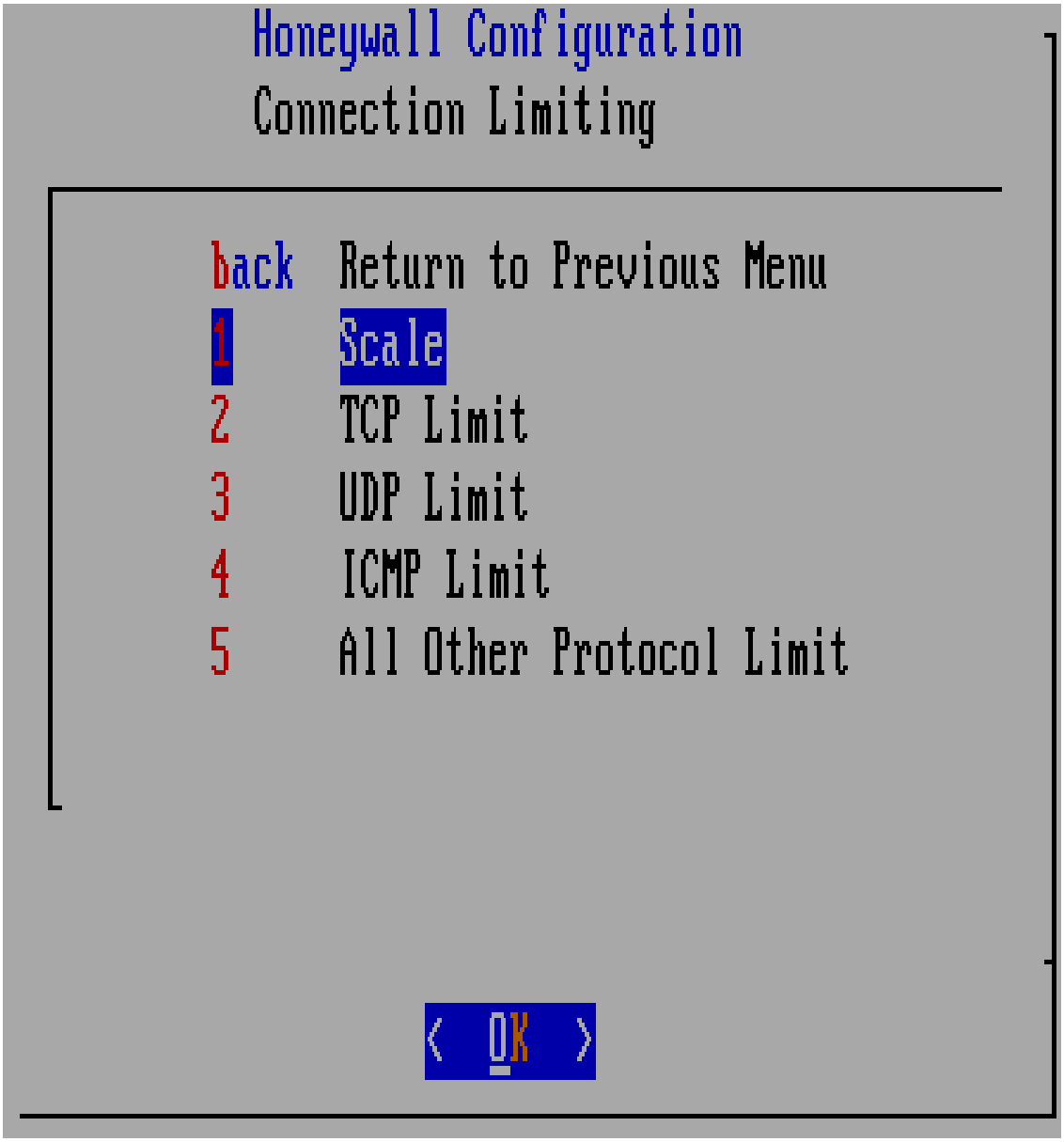


Figure 6.2 Honeywall Connection Limiting

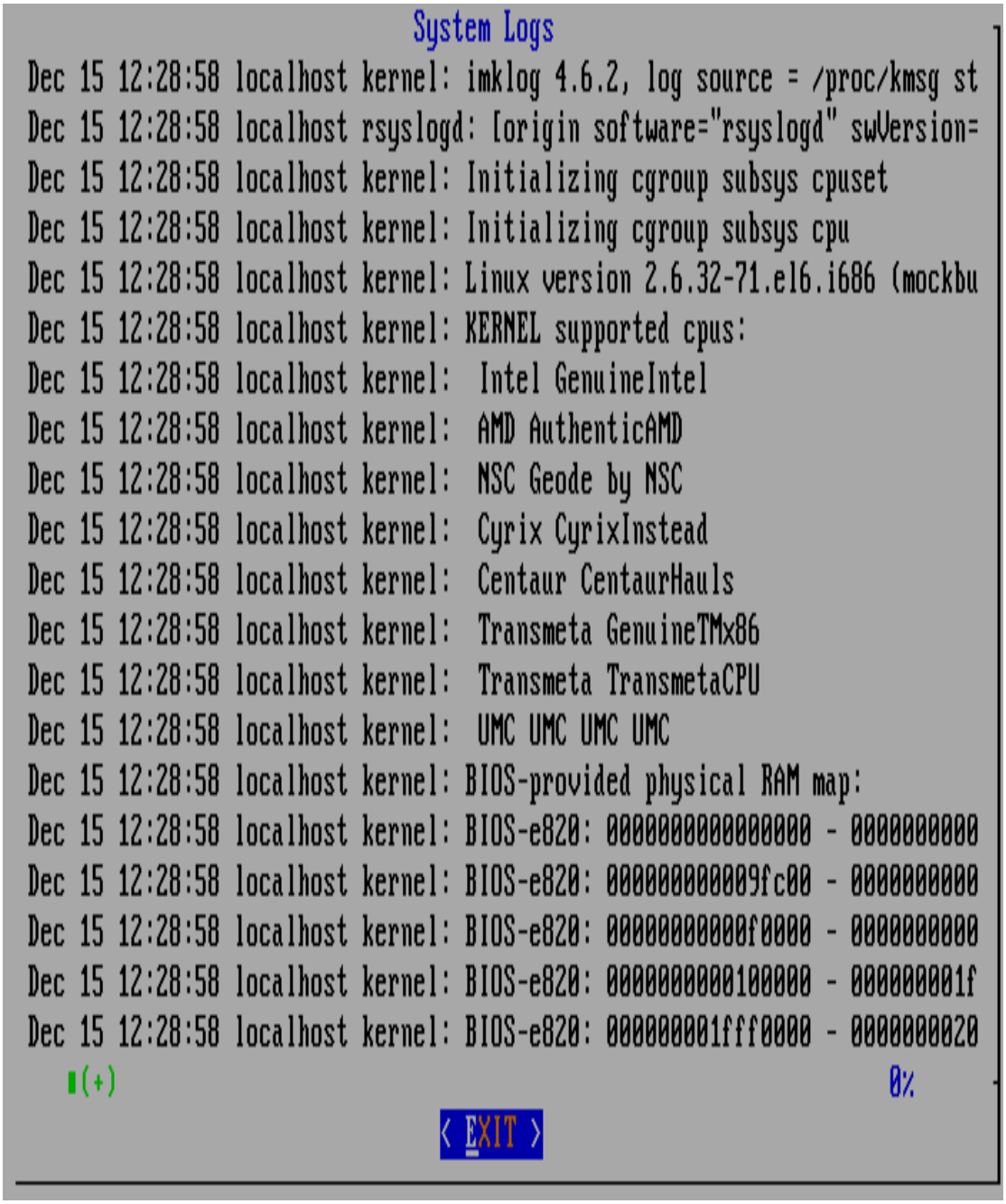


Figure 6.3 Honeywall System Logs

```
      Other:            0 (  0.000%)
Bad Chk Sum:            0 (  0.000%)
    Bad TTL:            0 (  0.000%)
     S5 G 1:            0 (  0.000%)
     S5 G 2:            0 (  0.000%)
      Total:           80
===============================================================================
Action Stats:
     Alerts:           20 ( 25.000%)
     Logged:           20 ( 25.000%)
     Passed:            0 (  0.000%)
Limits:
      Match:            0
      Queue:            0
        Log:            0
      Event:            0
      Alert:            0
Verdicts:
      Allow:           40 ( 50.000%)
      Block:           20 ( 25.000%)
    Replace:            0 (  0.000%)
  Whitelist:            0 (  0.000%)
  Blacklist:           20 ( 25.000%)
     Ignore:            0 (  0.000%)
===============================================================================
Normalizer statistics:
              ip4::trim: 0
               ip4::tos: 0
                ip4::df: 0
                ip4::rf: 0
               ip4::ttl: 0
              ip4::opts: 0
            icmp4::echo: 0
               ip6::ttl: 0
              ip6::opts: 0
            icmp6::echo: 0
           tcp::syn_opt: 0
```

Figure 6.4 Summary: Network Activity

# CHAPTER 7: CONCLUSION AND FUTURE SCOPE

## 7.1 Conclusion

Work presented in this thesis is towards architectural enhancement of virtual honeynet by implementing host machine as honeywall. Virtual honeynet is new method to implement honeynet system on a single physical machine. Though it has been evolved much earlier but traditional virtual honeynet architecture has certain limitations like it do not take advantage of underlying architecture on which it is deployed. Old architecture was resource intensive and bulky which was not easy to maintain and operate, and no mechanism was there to update snort signatures in automated way. Hence new optimized honeywall with architectural enhancement is needed where it will take advantage of underlying architecture. New enhanced architecture of virtual honeynet having host machine as honeywall has proper network management and supports different types of honeypots on a single physical machine. New optimized virtual honeynet is more fast and stable in terms of computation as data control and capturing is implemented on host machine. It reduce excess load generated on CPU and memory for running honeywall in a separate virtual machine. The host machine on which honeywall is implemented including data capturing, data control, data collection and alert generation is easy to deploy and maintain. The performance measurement in the form of computations is presented in the research implementation.

## 7.2 Future Scope

New enhanced architecture presented in this thesis work is mainly based upon functioning of virtual honeynet system as client node. It will just forward generated logs and data collected at host machine based honeywall.

Future work need to be done to implement early prediction system based upon machine learning by maintaining integrated database at honeywall and centralized server. This implementation will be done in distributed environment. Mentioned integrated database will contain iptables, snort and system logs.

# REFERENCES


[1] Watson, D. and J. Riden. The honeyney project: Data collection tools, infrastructure, archives and analysis, 2008: IEEE.

[2] Li, Z., Goyal, A., & Chen, Y. (2008). Honeynet-based botnet scan traffic analysis. In Botnet Detection (pp. 25-44). Springer US.

[3] Yegneswaran, V., P. Barford, and V. Paxson. Using honeynets for internet situation awareness. 2005: Citeseer.

[4] Memari, N., Samsudin, K., & Hashim, S. Towards Virtual Honeynet Based on LXC Virtualization, 2014: IEEE

[5] Awad, J. and A. Derdemezis, Implementation of a high interaction honeynet testbed for educational and research purpose, 2009.

[6] Spitzner, L. (2012). Honeypots: Tracking Hackers. US: Addison Wesley. pp 1-430.

[7] Ying-Dar Lin; Chia-Yin Lee; Yu-Sung Wu; Pei-Hsiu Ho; Fu-yu Wang; Yi-Lang Tsai, "Active versus Passive Malware Collection," *Computer* , vol.47, no.4, pp.59,65, Apr. 2014 doi: 10.1109/MC.2013.226

[8] Provos, N. and Holz, T (July 26, 2007). Virtual Honeypots: From Botnet Tracking to Intrusion Detection. US: Addison-Wesley Professional

[9] Liu, X., L. Peng, and C. Li, The Dynamic Honeypot Design and Implementation Based on Honeyd, Advances in Computer Science, Environment, Ecoinformatics and Education 2011, Springer. p 93-98.

[10] Mokube, I., & Adams, M. (2007, March). Honeypots: concepts, approaches, and challenges. In proceedings of the 45th annual southeast regional conference (pp 321-326). ACM

[11] Singh, A. N., & Joshi, R. X. (2011, July). A honeypot system for efficient capture and analysis of network attack traffic. In Signal Processing, Communication, Computing and Networking Technologies (ICSCCN), 2011 International Conference on (pp. 514-519). IEEE.

[12] Zhuge, J., Holz, T., Han, X., Song, C., & Zou, W (2007). Collecting autonomous spreading malware using high interaction honeypots. In Information and Communication Security (pp 438-451). Springer Berlin Heidelberg

[13] Hunter, S. O., Virtual Honeypots: Management, attack analysis and democracy. 2010

[14] Gani, A., Improving exposure of intrusion deception system through implementation of hybrid honeypots. The International Arab Journal of Information Technology, 2012. 9(5): p 436-444.

[15] Spitzner, L., "The Honeynet Project: trapping the hackers," Security & Privacy, IEEE , vol.1, no.2, pp.15,23, Mar-Apr 2003 doi: 10.1109/MSECP.2003.1193207

[16] Abbasi, F.H.; Harris, R.J., "Experiences with a Generation III virtual Honeynet," *Telecommunication Networks and Applications Conference (ATNAC), 2009 Australasian* , vol., no., pp.1,6, 10-12 Nov. 2009 doi: 10.1109/ATNAC.2009.5464785

[17] The Honeynet Project. (2005). Know Your Enemy: Honeywall CDROM Roo. Available: http://old.honeynet.org/papers/cdrom/roo/index.html. Last Access Dec 2014

[18] Oracle (2015). Oracle VM VirtualBox 4.3.28. Available: https://www.virtualbox.org/wiki/Downloads. Last accessed 01 July 2015

[19] Lok Kwong Yan, "Virtual honeynets revisited," *Information Assurance Workshop, 2005. IAW '05. Proceedings from the Sixth Annual IEEE SMC* , vol., no., pp.232,239, 15-17 June 2005 doi: 10.1109/IAW.2005.1495957

# LIST OF PUBLICATIONS

[1] Gautam, R., Kumar. S, & Bhattacharya, J. "Optimized Virtual Honeynet with Implementation of Host machine as Honeywall" 12th IEEE India International Conference (communicated)

[2] Gautam, R., Kumar. S, & Bhattacharya, J. "CPU mining Malware: Complete Analysis of Real World Virus " International Journal of Research in Advent Technology (communicated )

# LINK TO THE UPLOADED VIDEO

https://www.youtube.com/watch?v=OFDz5eSXSgQ