Design and Implementation of Linux Based Virtual HoneyClient

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Technology
in
Computer Science and Applications

Submitted By
Himani Gupta
(Roll No. 601003008)

Under the supervision of:
Mr. Gurpal Singh Chhabra
Lecturer

SCHOOL OF MATHEMATICS AND COMPUTER APPLICATIONS
THAPAR UNIVERSITY
PATIALA – 147004
July 2012

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, **"Design and Implementation Of Linux Based Virtual Honeyclient"**, in partial fulfillment of the requirements for the award of degree of Master of Technology in Computer Science and Applications submitted in School of Mathematics and Computer Applications of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of **Mr. Gurpal Singh Chhabra** 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.

Himani Gupta

(Himani Gupta)

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

(Mr. Gurpal Singh Chhabra)
Lecturer,
School of Mathematics and Computer Applications

Countersigned by

(Dr. S.S Bhatia)
Head
School of Mathematics and Computer Applications
Thapar University
Patiala

(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

ACKNOWLEDGEMENT

First of all, I thank the almighty for his blessings and showing me the right direction. With His mercy, it has been made possible for me to reach so far.

It gives me great pleasure to express my gratitude towards the guidance and help I have received from Mr. Gurpal Singh Chhabra. I am thankful for his continual support, encouragement, and invaluable suggestion. He not only provided me help whenever needed, but also the resources required to complete this thesis report on time.

I am also thankful to Dr. S.S Bhatia, Head, School of Mathematics and Computer Applications for his kind help and cooperation. I express my gratitude to all the staff members of School of Mathematics and Computer Applications for providing me all the facilities required for the completion of my thesis work.

I would like to say thanks to all my friends especially for their support. I want to express my appreciation to every person who contributed with either inspirational or actual work to this thesis.

Last but not the least I am highly grateful to all my family members for their inspiration and ever encouraging moral support, which enables me to purse my studies.

Himani Gupta

ABSTRACT

A honeypot is used in the area of computer and Internet security. It is a resource which is intended to be attacked and compromised to gain more information about the attacker and his attack techniques. It can also be used to attract and divert an attacker from the real targets. Compared to an intrusion detection system, honeypots have the big advantage that they do not generate false alerts as all traffic is suspicious, because no productive components are running on the system. This fact enables the system to log every byte and to correlate this data with other sources to draw a picture of an attack and the attacker.

Linux has been chosen as the Operating system of choice for a number of apparent advantages that it has over other Operating systems. The most important ones being that the original work on Honeynets / Honeypots has been done using Linux, so it is best supported out of all the contemporaries. Another reason for using Linux is that it is Open source and provides source code to work with freely

In this thesis the intention is to understand the intricacies of the workings of the honeypot / honeynet technology and also the reasoning, thought process and goals behind it. Then a “Virtual Client Honeypot” would be implemented to collect internet malwares.

Table of Contents

List of contents — Page No.

List Of Tables

List Of Figures

Chapter 1
INTRODUCTION

This chapter introduces network security, intrusion detection system, its classification and techniques and also about honeypots & its types and organisation of thesis.

Internet is a global public network. With the growth of the Internet and its potential, there has been subsequent change in business model of organizations across the world. More and more people are getting connected to the Internet every day to take advantage of the new business model popularly known as e-Business. Internet work connectivity has therefore become very critical aspect of today's e_business.

There are two sides of business on the Internet. On one side, the Internet brings in tremendous potential to business in terms of reaching the end users. At the same time it also brings in lot of risk to the business. There are both harmless and harmful users on the Internet. While an organization makes its information system available to harmless Internet users, at the same time the information is available to the malicious users as well. Malicious users or hackers can get access to an organization's internal systems in various ways. These are:

- Software bugs called vulnerabilities
- Lapse in administration
- Leaving systems to default configuration

Therefore, there needs to be some kind of security to the organization's private resources from the Internet as well as from inside users, as survey says that eighty percent of the attacks happen from inside users for the very fact that they know the systems much more than an outsider knows and access to information is easier for an insider.

Different organizations across the world deploy firewalls to protect their private network from the Public network. But when it comes to securing a Private network from the Internet using firewalls, no network can be hundred percent secured. This is because; the business requires some kind of access to be granted on the Internal systems to Internet users. The firewall provides security by allowing only specific services through it. The firewall implements a policy for allowing or disallowing connections based on organizational security policy and business needs. The firewall also protects the organization from malicious attack from the Internet by dropping connections from unknown sources [1].

SECURITY ATTACKS

Network systems are vulnerable to various attacks. Security attacks can be classified into *server side attacks* and *client side attacks*.

A. Server Side Attacks

Server-side attacks are launched against servers. The adversary tries to discover the vulnerabilities in services provided by server to compromise it. Therefore, merely running a server exposes potential risk, as an attacker can initiate attacks at any moment if the service is vulnerable indeed. For example, an adversary could send a maliciously crafted HTTP request to a vulnerable web server and attempt to leverage errors or other unexpected behavior.

B. Client Side Attacks

Client-side attacks refer to the attacks against *end user* (aka *client user).* In this type of attacks, an attacker uses client application vulnerability to take control of client system by malicious server. A typical target is web browser. However, these attacks can occur on any client/server pairs such as email, instant messaging, FTP, multimedia streaming, etc. One example of non-browser application vulnerability exploits is Adobe Reader v8.1.2 which is prone to stack-based buffer overflow vulnerability. In general, client-side exploits require user-interaction such as enticing the user to click a link, open a document, or somehow to let her visit the malicious websites [5].

1.1 Intrusion Detection System

An Intrusion detection system (IDS) is a security system that monitor computer systems and network traffic and analyses that traffic for possible hostile attacks originating from outside the organization and also for system misuse or attacks originating from inside organization.

The underlying reasons why you might use intrusion detection systems are relatively straightforward: You want to protect your data and systems integrity. The fact that you cannot always protect that data integrity from outside intruders in today's internet environment using mechanisms such as ordinary password and file security, leads to a range of issues [1].

Intrusion detection takes that one step further. Placed between the firewall and the system being secured, a network based intrusion detection system can provide an extra

layer of protection to that system. For example, monitoring access from the internet to the sensitive data ports of the secured system can determine whether the firewall has perhaps been compromised, or whether an unknown mechanism has been used to bypass the security mechanisms of the firewall to access the network being protected.

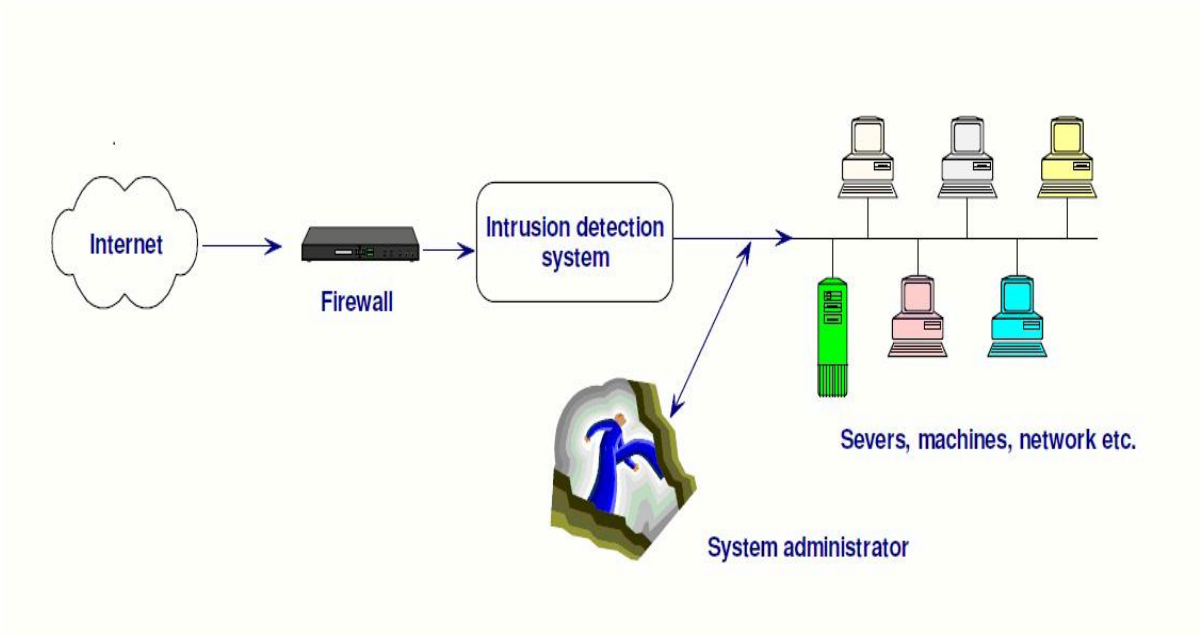

Figure 1.1 IDS Environment [1]

1.2 Classifications of Intrusion Detection System

- Host Based IDS
- Network Based IDS

1.2.1 Host Based IDS

Host based IDS consists of software or AGENT components, which exist on Server, Router, Switch or Network appliance. The agent versions must report to a console or can be run together on the same Host. This is NOT the preferred method though.

1.2.2 Network Based IDS

Network based IDS captures network traffic packets (TCP, UDP, IPX/SPX, etc.) and analyses the content against a set of rules or signatures to determine if a possible event took place. False positives are common when an IDS system is not configured or "tuned" to the environment traffic it is trying to analyse. Network

Node is merely an extended model of the networked IDS systems adding aggregated and dedicated IDS servers on each NODE of a network in order to capture all the networked traffic not visible to other IDS servers.

1.3 Techniques based classification

Intrusions can be detected by various techniques. Most important of those are:

1.3.1 Anomaly detection

Normal behaviour patterns are useful in predicting both user and system behaviour. Here, anomaly detectors construct profiles that represent normal usage and then use current Behaviour data to detect a possible mismatch between profiles and recognize possible attack attempts. In order to determine what is attack traffic, the system must be taught to recognize normal system activity. This can be accomplished in several ways, most often with artificial intelligence type techniques. Systems using neural networks have been used to great effect. Another method is to define what normal usage of the system comprises using a strict mathematical model,
and flag any deviation from this as an attack. This is known as strict anomaly detection.

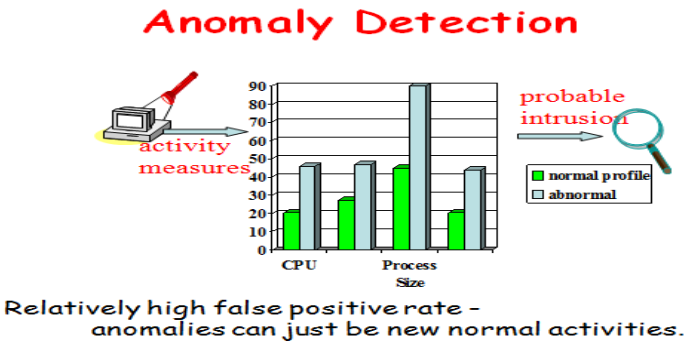

Figure 1.2 Diagram of Anomaly Detection [2]

1.3.2 Signature detection

Systems possessing information on abnormal, unsafe behavior are often used in real-time intrusion detection systems (because of their low computational complexity).

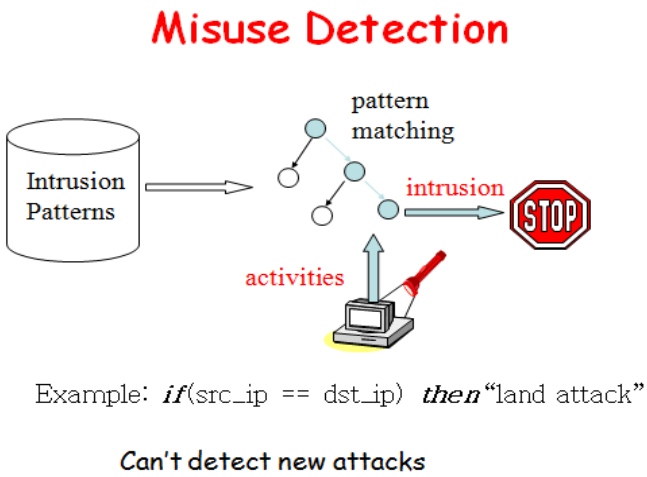

Figure 1.3 Diagram of Misuse Detection[2]

1.4 Types of Intruders:

The term "intruders" compromises more than just human attackers who manage to gain access to computer resources although the resource was not meant to be used by them.

Misfeasor

Imagine someone who emails blueprints and schematics the company he works for is holding a patent on to his home email account in order to sell it to a competitor company. Nowadays we can take for granted that someone has access to an email accounts or a printer at work. It is obvious that no data was accessed without authorization in this example. However, the user misused some of his privileges. On this account we define misfeasor as an individual who works within the scope of his privileges but misuses them.

Clandestine user

Another user might take advantage of a security hole in the operating system in or gain administrative privileges to a computer resource. We define clandestine user as an through individual who seizes supervisory control to disengage or avoid securitymechanisms of the system.

Masquerader

A third individual could steal another user's login id and the associated password. We define masquerader as an individual who overcomes a systems access control to exploit a legitimate user's account [3].

In the recent years, new trends in the adversary techniques have risen. This led to growth of new technologies and tools to supplement the existing security defends such as firewalls and intrusion detection systems (IDS). One of these technologies is *honeypot*. The purpose of intrusion detection systems is, as the name implies, to detect intruders. Honeypots are able to do the same, but they also go one step further. A honeypot is a deception device that lures attackers. It is a system that is located inside a computer network and has no productive value. It just waits to become attacked. Because this system has no productive value, all connections to it are most likely scans, probes or attacks. Attackers who operate on a honeypot are heavily monitored. This allows to retrace their activity and to learn from them. The gained knowledge can be used to harden the production systems.

1.5 Study of Honeypot

A honeypot is "an information system resource whose value lies in unauthorized or illicit use of that resource"[4].

A Honeypot is defined as "being a security resource whose value lies in being probed or compromised". They can be either host and/or network based, but are more often than not network based as all interaction is typically performed over a network connection. A honeypot's main utility comes from the fact that it simplifies the Intrusion Detection problem of separating "anomalous" from "normal" by having no legitimate purpose, thus any activity on a Honeypot can be immediately defined as anomalous. The characteristics of Honeypots make them well suited to the monitoring of malcious activity on networks, as well as being a valuable research tool in Computer Security.

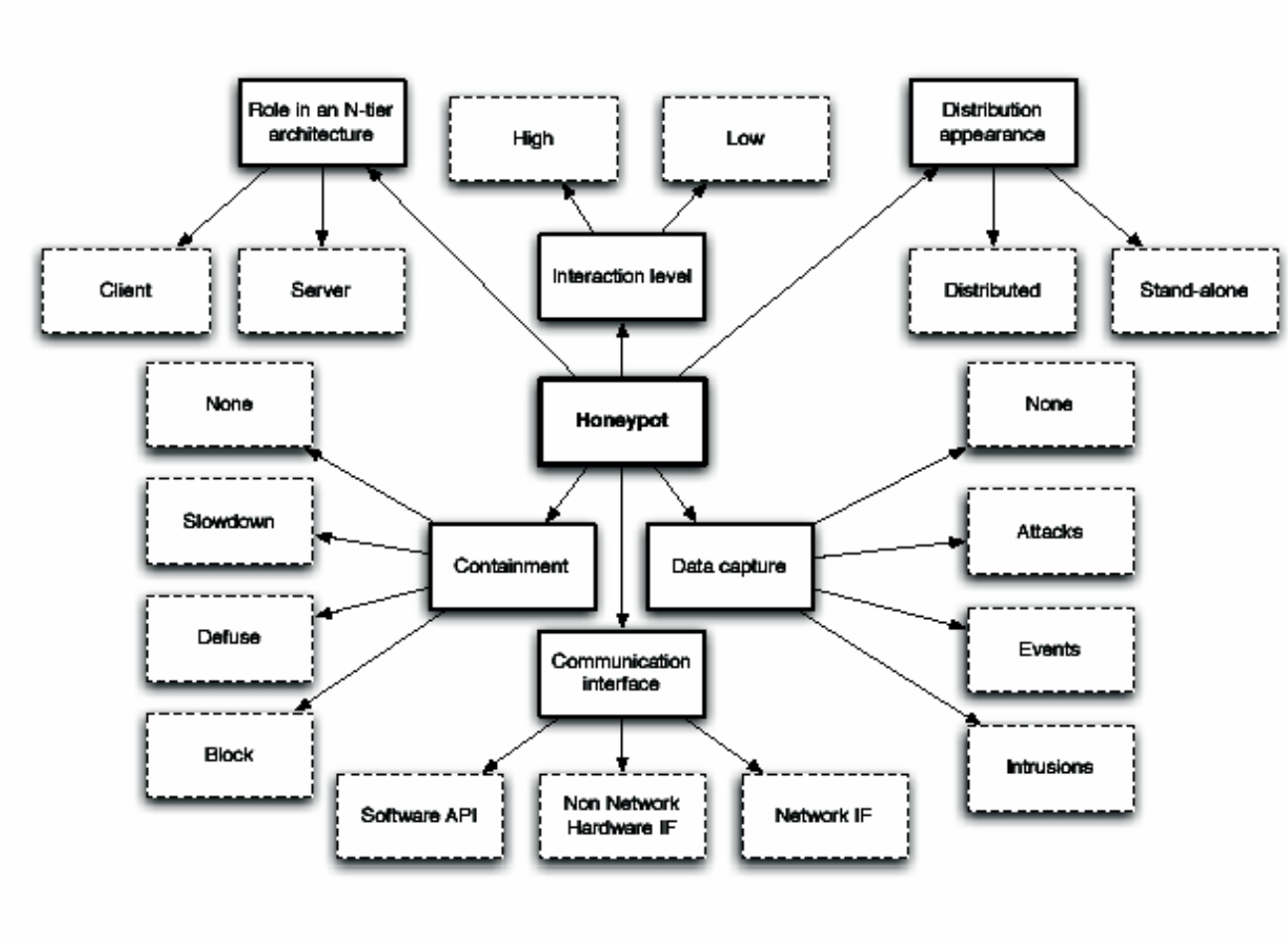

Figure 1.4 Taxonomy of Honeypot[4]

1.5.1 Types of Honeypots

There are several different types of honeypots and similar devices that vary in their field of application, strength or weakness. The main distinction between different types of honeypots is made on the level of interaction they provide to the attacker. Which type of honeypot to use depends on the aim to achieve.

Installation and Configuration: This category measures the time and effort in installing the honeypot. The more functionality a honeypot provides, the more complex is it to setup and operate.

Deployment and Maintenance: This category is similar to the one above. The more options and services a honeypot provides, the more time and resources are needed to deploy and maintain the system.

Information Gathering: The more interaction a honeypot allows, the more information can be gained.

Level of Risk: A higher level of interaction means more complexity and results in bigger risk. A honeypot that provides a maximum of interaction to anattacker, can be misused by the intruder to attack other systems.

1.5.2 Value of Honeypots

To examine the value of honeypots, they have to be separated into two categories- production honeypots and research honeypots.

Production Honeypot

A production honeypot is a system that reduces the risk in a networked environment. This is realized by maintaining a system that monitors unauthorized traffic but does not provide any attack point to the intruder[5]. A production honeypot is therefore mostly a low-interaction honeypot. The value of a production honeypot can be demonstrated by using Bruce Schneiers security concept of Prevention, Detection and Response.

- Prevention

The term prevention means to keep attackers out. Honeypot resources do that by keeping the attackers busy. As long as an attacker wastes his time on a honeypot, he cannot attack production systems. Attackers do not like to be under surveillance. They are often scared off, if they are aware of the presence of deception systems. Automated attacks, such as worms or auto-rooters, begin their attack by random scans of entire networks looking for vulnerable systems. If they find any, these automated tools will capture these systems. Honeypots are able to slow down the scans or even to stop them. They do this by using several TCP tricks. By using a windows size of zero, the attacker can be kept in a holding pattern. This is a trick that cannot be done on production systems, because this would slow down the network traffic. However in a non-productive environment like a honeypot, this tactic is able to prevent worms from spreading.

Although there are some arguments that justify honeypots as prevention tools, there might be better ways to increase prevention. A well defined and asserted security policy might provide a better prevention at a lower cost.

- Detection

The main goal of a production honeypot is to detect unauthorized activity. Whereas an intrusion detection system has to be configured properly to detect suspicious traffic, a production honeypot needs no long winded configuration just to detect unauthorized traffic. It does not produce any false alarms. This kind of system is very good at detecting new kinds of attacks, that have never been seen before. If an attacker evades the honeypot and assaults a productive system, the honeypot is not able to detect anything. Therefore it is important to fill the honeypot with as much honey as possible, to attract attackers.

- Response

To evaluate an incident it is necessary to take the compromised system offline for a precise investigation. Production systems have to run all day long and do not allow an in-depth examination. They also pollute the gathered information from the attacker with normal activity. This makes it hard to reproduce the incident. After an incident, the honeypot can be taken offline without caution. It can be stripped down for further investigations. This allows to get maximum information from the data recorded. One of the areas the Honeynet Project intends to research is Early Warning and Prediction. Their intent is to give more value to the data, that is gathered by honeynets, by predicting future attacks.

Research Honeypot

The lack of information about attackers, their proceedings and the tools they use is a big problem for the the IT industry. It is difficult to protect systems if the enemy is not known. If a system gets compromised, the blackhats often leave the tools they use on the machine. This allows security professionals to examine them just like archaeologists do with ancient weapons. A research honeypot goes one step further. It allows to follow an attack step by step from the beginning to the end. A research honeypot is often a regular computer system. It is a real system and does not just emulate some services[5]. The blackhat should have the possibility to interact with the honeypot just as with every other operating system on a production machine. This kind of honeypot has to pretend to be very interesting to the attacker. To get as much information as possible from the attacker, the honeypot logs all events from the system, the installed applications or the keystrokes to a remote system. This prevents the

blackhat from covering his tracks. Research honeypots are able to capture automate threats like worms or auto-rooters. They capture these tools for examination of their dangerous payload. Furthermore, it is possible to capture the communication between blackhats. They often start Internet relay chat (IRC) sessions from captured systems to talk to other members of the blackhat community.

Based on design criteria, honeypots can be classified as

1. pure honeypots
2. high-interaction honeypots
3. low-interaction honeypots

Pure honeypots are full-fledged production systems. The activities of the attacker are monitored using a casual tap that has been installed on the honeypot's link to the network. No other software needs to be installed. Even though a pure honeypot is useful, stealthiness of the defense mechanisms can be ensured by a more controlled mechanism.

High-interaction honeypots imitate the activities of the real systems that host a variety of services and, therefore, an attacker may be allowed a lot of services to waste his time. According to recent researches in high interaction honeypot technology, by employing virtual machines, multiple honeypots can be hosted on a single physical machine. Therefore, even if the honeypot is compromised, it can be restored more quickly. In general, high interaction honeypots provide more security by being difficult to detect, but they are highly expensive to maintain. If virtual machines are not available, one honeypot must be maintained for each physical computer, which can be exorbitantly expensive. Example: Honeynet.

Low-interaction honeypots simulate only the services frequently requested by attackers. Since they consume relatively few resources, multiple virtual machines can easily be hosted on one physical system, the virtual systems have a short response time, and less code is required, reducing the complexity of the security of the virtual systems. Example: Honeyd.

1.5.3 Advantages

A honeypot does not provide additional security on its own. It even attracts blackhats to itself or the network it resides in. Nevertheless does this technology provide some

advantages. Four of them will be presented according to the book Honeypots: Tracking Hackers.

- Data Value

Every computer infrastructure contains a lot of data sources that collect huge amount of data on what is going on. Every service and every machine has its own loggles to report any type of event. Most organizations are not able to derive information from that data, because they are overwhelmed by an incredible number of reported events. Honeypots, in contrary, collect very little data. They start logging if a scan or attack is in progress and they stop it if the attacker leaves the system. There is no need to separate normal and dangerous traffic, because the honeypot receives no normal or productive traffic. Every connection to this system is an evidence of an attack.

- Resources

As honeypots monitor little activity, they do not have the problem of resource exhaustion. Normal traffic that consists of huge amount of network packets bypasses the honeypot[6]. The honeypot system itself rarely has to handle more than a few attackers at a time. Moreover, it might be an additional advantage of a honeypot to slow down the attacks by minimal resources.

- Simplicity

A honeypot does not need complex configurations. Just to drop a system somewhere on the network might be enough to capture some suspicious activity. The simplicity of the honeypot concept is the reason for its reliability.

- Return on Investment

Investments into security technologies can become victims of their own success. Firewalls or encryption might reduce the risk of an attack. An organization that is well protected by such expensive technologies for years might perceive that there is no threat anymore. Honeypots demonstrate their own value by capturing unauthorized activity. They can be used to justify other investments into security technologies.

1.5.4 Disadvantages

There is no ultimate security solution in reality. Therefore even honeypots suffer from some disadvantages.

- Narrow Field of View

A disadvantage of honeypots is the fact that they have only a limited view.While an IDS analyzes the data of a whole network, the honeypot only captures connections to itself. If attackers evade the honeypot and attack directly production systems, they will not be detected.

- Fingerprinting

Fingerprinting means that attackers are able to identify a honeypot on the basis of some characteristics [6]. At the moment the attacker is aware of the presence of a honeypot, he might use it for an attack or just leave it and probably turn toward production systems.

- Risk

Honeypots might introduce an additional risk to the existing infrastructure. A successfully identified and attacked honeypot can be misused for further malicious activity. By maintaining and observing the honeypot, the risk can be minimized, but never totally eliminated.

1.6 Honeypot vs. IDS

Honeypots have the ability to overcome some of the disadvantages of intrusion detection systems as written by Lance Spitzner in Honeypots: Definitions and Value of Honeypots.

- Capacity

High traffic networks tend to overstrain the logging capacity of intrusion detection systems. IDS for gigabit networks increase expenses in time, money and hardware. In contrast to IDS, honeypots only need minimal resources, because normal traffic bypasses these systems.

- Information content

Intrusion detection systems in large networks suffer from the high amount of traffic. They have to inspect thousands of connections a day to find a few suspicious packets. There is a high chance to miss them or to generate a big number of false alarms. Honeypots in contrary just have to handle traffic directed to themselves [7]. All traffic to the honeypot is suspicious. This fact lowers the chance of false alarms.

- New kind of attacks

It is difficult for intrusion detection systems to detect new kind of attacks. They often need descriptions to identify attacks. All traffic to a honeypot seems malicious or dangerous and there is a high chance to detect attacks that never occurred before.

- Insider detection

A big challenge for intrusion detection systems is to detect attacks from insiders. Any connection from one machine to another inside the network is most likely caused by regular usage. However connections from machines inside the network to the honeypot are very suspicious and might be an evidence of a regular user who exceeds his competences.

- Encrypted traffic

Intrusion detection systems that eavesdrop on the network wire are not able to monitor encrypted traffic. As the honeypot is the endpoint of the encrypted connection, encrypted traffic to honeypot systems is easy to analyze.

1.7 Organization of Thesis

The rest of the thesis is organized as follows:

Chapter 2 – This chapter describes in detail the literature survey done to study the concept of Network security, Intrusion Detection System, Honeypot and Client Honeypot.

Chapter 3 – This chapter describes the problem statement of the thesis work. It gives the gap analysis and Goals of the Client Honeypot.

Chapter 4 – This chapter focus on the implementation details and experimental results – Design of Client Honeypot, Functional Diagram of Client Honeypot, Design of Implemented Setup and Experimental Results.

Chapter 5 –This chapter describes the conclusion, contribution to the work done and future research work possible.

Chapter 2

LITERATURE SURVEY

This Chapter discusses the state of the art & research issues of network security, intrusion detection system, Honeypot & Honeyclient.

With the explosion of the public Internet and e-commerce, private computers, and computer networks, if not adequately secured, are increasingly vulnerable to damaging attacks. Hackers, viruses, vindictive employees and even human error all represent clear and present dangers to networks. And all computer users, from the most casual Internet surfers to large enterprises, could be affected by network security breaches. However, security breaches can often be easily prevented. How? This guide provides you with a general overview of the most common network security threats and the steps you and your organization can take to protect yourselves from threats and ensure that the data traveling across your networks is safe[8].

Pikoulas J, Mannion M, Buchanan W in the titled "*Software Agents and Computer Network Security"* stated that security enhancement software system using software agents, in which a core software agent resides on one server in a Windows NT network system and user end software agents reside in each user workstation. The user agent at the client workstation takes all decisions and actions about invalid user behaviour. This means that even when the core agent process is no longer operating security breeches at the client workstation can still be detected and an appropriate course of action taken[9].

Robert E. Mahan presented a program in the title" *Introduction to Computer & Network Security"* stated that The purpose of computer and network security is to provide an umbrella of protection for the organization in the form of policies and procedures as well as the technological implementation of measures to secure and protect an organization's assets. A security policy is a set of rules that establish expected behavior and performance. A wide variety of implementing mechanisms are available to the organization to protect systems from a range of threats such as Confidentiality, Integrity, Authentication, Access control, Non-repudiation and Availability[10]**.**

In the last three years, the networking revolution has finally come of age. More than ever before, we see that the Internet is changing computing, as we know it. The possibilities and opportunities are limitless; unfortunately, so too are the risks and chances of malicious intrusions. It is very important that the security mechanisms of a system are designed so as to *prevent* unauthorized access to system resources and data. However, completely preventing breaches of security appear, at present, unrealistic. We can, however, try to detect these intrusion attempts so that action may be taken to repair the damage later. This field of research is called **Intrusion Detection** [11].

Usman Asghar Sandhu, Sajjad Haider, Salman Naseer, Obaid Ullah Ateeb stated in the title "*A Survey of Intrusion Detection & Prevention Techniques***"** gives review of the Intrusion detection system that IDS is a process of monitoring the events occurring in a computer system or network and analyzing them for sign of possible incident which are violations or imminent threats of violations of computer security policies or standard security policies. Intrusion Prevention System (IPS) is a process of performing intrusion detection and attempting to stop detected possible incidents [12]. IDPS provides the facility to detect and prevent from attacks by inheriting multiple approaches like secure mobile agent, virtual machine; high throughput string matching, multilayer and distributed approach provide greater and strongest security against multiple attacks. There are still many ways to improve the virtual machine based Intrusion Detection and Prevention System.

Wayne Jansen, Peter Mell, Tom Karygiannis, Don Marks suggested innovative ways to apply mobile agents in the paper titled "*Applying Mobile Agents to Intrusion Detection and Response"* that the idea of mobile and autonomous components intuitively seems useful in intrusion detection and many other applications. Although the barriers to creating practical mobile agent systems are high, the ability to move a running program from one hardware platform to another is a useful feature. Ultimately, as the security, performance, emerging technology, and standards barriers that inhibit this technology fall, mobile agents will enter mainstream use [13]. Not only do mobile agents appear to be useful in general, but they appear useful to IDSs. Mobile agents may enhance the performance of IDSs and even offer them new capabilities.

G. Cabri, L. Leonardi, F. Zambonelli focused on mobile agent technology in the paper named "*Mobile Agent Technology: Current Trends And Perspectives"* that in a high-connected world, agents are the active entities, which roam across network nodes on behalf of their

senders. There are several open issues, which are to be faced to grant the diffusion of applications based on mobile agents. The first important issue is coordination. Traditional approaches, do not suit properly to the new requirements imposed by the mobility of the agents [14]. A further issue is security. Mobile agents are going to remain inside the field of academic research and cannot be integrated in actual applications, unless an adequate degree of security is provided to defend both network sites from pirate agents and positive agents from malicious sites. Other issues are standardisation and efficiency. Even if these ones are important issues for the wide acceptance of this new technology, they are to be solved at implementation level, and do not affect the mobile agent model.

Shiv Shakti Srivastava, Nitin Gupta, Saurabh Chaturvedi, Saugata Ghosh summarized the current state of the mobile agent based intrusion detection system in the paper titled " *A Survey on Mobile Agent based Intrusion Detection System***"** that a Mobile agent (MA) is a composition of computer software and data which is able to migrate (move) from one computer to another autonomously and continue its execution on the destination computer. Immune mobile agent who is based upon Distributed ID System improves dynamic clonal section algorithm and collaborative signal mechanism to increase detection rate and reduce false positive rate [15]. In co-ordination association approach of IDS, dynamic load balancing technique can be used to resolve the problem of complexity in the network having large number of subnets . The IDS approach can be enhanced by providing more security to mobile agents.There is need to investigate the new concept of behavior to make this agent more intelligent to enhance the actual performance and track any new type of attack which is the main purpose to use the network IDS.

In the recent years, new trends in the adversary techniques have risen. This led to growth of new technologies and tools to supplement the existing security defends such as firewalls and intrusion detection systems (IDS). One of these technologies is *honeypot*. In the area of information security, the term of honeypot refers to a closely monitored computing resource that we want to be probed, attacked or compromised. Lance Spitzner defines a honeypot to be "a resource whose value is in being probed, attacked or compromised." [16]. While the battle is ongoing on the Internet between blackhats and whitehats, attackers have started to transfer the battlefield to the client user; as they believe the client applications are more likely to have security breaches and vulnerabilities.

Feng Zhang, Shijie Zhou, Zhiguang Qin, Jinde Liu presented typical honeypot solutions and prospect the technical trends of integration, virtualization and distribution for the future honeypot in the paper titled "*Honeypot: a Supplemented Active Defense System for Network Security"* that Honeypot is a supplemented active defense system for network security. It traps attacks, records intrusion information about tools and activities of the hacking process, and prevents attacks outbound the compromised system. Integrated with other security solutions, honeypot can solve many traditional dilemmas. Working with **IDS** and firewall, Honeypot provides new way to attacks prevention, detection and reaction. Honeypot can serve **as** a good deception tool for prevention of product system because of it's ability of trapping attacker to a decoy system [17]. Supplemented with IDS, honeypot reduces false positives and false negatives. Intelligence routing control provides flexible response to attacks. Different kinds of honeypot share the common technologies of data control and data capture. Researchers focus the two to make honeypot easier to deploy and more difficult to detect. Integrated honeypot encapsulates all the components in **a** single device. Virtual honeypot creates large number of honeypot systems in one machine. Distributed honeypot comprises different honeypot system in an actual network to offer high interaction between **attacks and** system.

Karthik Sadasivam, Banuprasad Samudrala, T. Andrew Yang presented a framework for designing assignments/projects for network security courses in the paper named "*Design of Network Security Projects Using Honeypots*' that Honeypots are closely monitored decoys that are employed in a network to study the trail of hackers and to alert network administrators of a possible intrusion. Nowadays, they are also being extensively used by the research community to study issues in network security, such as Internet worms, spam control, DoS attacks, etc. The design of our projects tackles the challenges in installing a honeypot in academic institution, by not intruding on the campus network while providing secure access to the Internet [18]. The virtual Honeynet was initially closed to the Internet due to concerns of ethical and legal issues, but was eventually open to the Internet world, in order to collect research data about the blackhat community. The collected data and their subsequent analysis helped us to protect our network from attackers.

Iyatiti Mokube and Michele Adams present an overview of Honeypots in the paper titled "*Honeypots: Concepts, Approaches, and Challenges*" that A honeypot is a security resource

whose value lies in being probed, attacked or compromised.In this paper researchers have discussed the different types of honeypots such as production honeypots, research honeypots, and honeytokens. The legal issues surrounding honeypots and their implementation were examined, An important point to remember is that experts advise using honeypots together with some other form of security such as an IDS. Honeypots are a relatively new technology that is becoming increasingly popular, and will become even more so as commercial solutions become available that are easy to use and administer. Because they can be used to collect information on attackers and other threats [19].

Client user has become the weakest link in the network security chain, and since the security chain is only robust as its weakest link, there is need to detect attacks against client side to protect the whole security system. One of the most common types of client-side attacks is using malicious websites to attack client user. Traditional forms of honeypots are unable to detect client-side attacks in many cases; because traditional honeypots are *passive*; they are waiting for attackers to attack them. Deception techniques are used to lure the adversary to initiate contact with honeypots.

Therefore, it has been necessary to use new form of *active* honeypots. The new form of active honeypots is called *client honeypot* or *honeyclient.*

Client honeypot aims at finding web servers which compromise the client applications. *HoneyClient* was the first open source client honeypot, which was developed in 2004 by K. Wang , and subsequently developed at MITRE. Later *HoneyMonkey* and *SiteAdvisor* have been released. However, HoneyMonkey and SiteAdvisor are proprietary systems, and not freely available for further research. A. Ikinci developed *Monkey-Spider* which is an open-source tool to detect malicious websites [20]. Two significant tools *Capture* and *HoneyC* have been developed by New Zealand Honeynet Project. These two tools have participated in the highlighting honeyclients as effective tools to detect malicious websites. Although honeyclients are the most prominent technology to fight malicious web sites today, but they are no sufficient resources about this topic .

2.1 Client Honeypot

Honeypots are security devices whose value lies in being probed and compromised. Traditional honeypots are servers (or devices that expose server services) that wait passively to be attacked. **Client Honeypots** are active security devices in search of malicious servers that attack clients. The client honeypot poses as a client and

interacts with the server to examine whether an attack has occurred. Often the focus of client honeypots is on web browsers, but any client that interacts with servers can be part of a client honeypot (for example ftp, ssh, email, etc.) [21].There are several terms that are used to describe client honeypots. Besides client honeypot, which is the generic classification, honeyclient is the other term that is generally used and accepted. However, there is a subtlety here, as "honeyclient" is actually a homograph that could also refer to the first open source client honeypot implementation although this should be clear from the context.

2.2 Architecture of Client Honeypot

A client honeypot is composed of three components. The first component, a queuer, is responsible for creating a list of servers for the client to visit. This list can be created, for example, through crawling. The second component is the client itself, which is able to make a requests to servers identified by the queuer. After the interaction with the server has taken place, the third component, an analysis engine, is responsible for determining whether an attack has taken place on the client honeypot [22].In addition to these components, client honeypots are usually equipped with some sort of containment strategy to prevent successful attacks from spreading beyond the client honeypot. This is usually achieved through the use of firewalls and virtual machine sandboxes.

Analogous to traditional server honeypots, client honeypots are mainly classified by their interaction level: high or low; which denotes the level of functional interaction the server can utilize on the client honeypot. In addition to this there are also newly hybrid approaches which denotes the usage of both high and low interaction detection techniques.

The Active honeypot architecture is divided into following three modules:

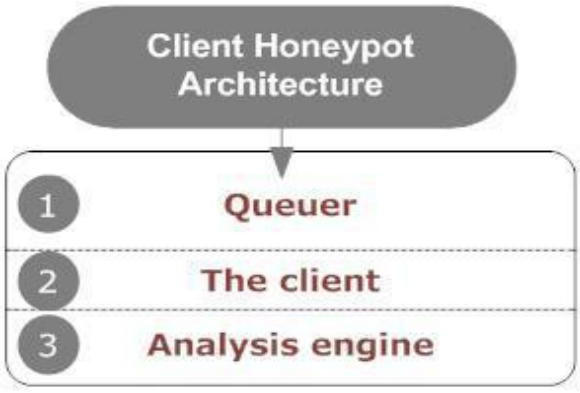

Figure 2.1 Architecture of Client Honeypot [22]

1) **Queuer**: the queuer is responsible for creating the list of the URLs that has to be browsed by the Active Honeypot. There are several techniques used to create URL lists, including search engines, Blacklists, Phishing and spam messages, and instant messaging.

2) **Client Module**: the client is the component that makes requests and interacts with the web servers. It emulates the browser level vulnerabilities.

3) **Analysis engine**: the analysis engine is responsible for determining and checking the state of the client honeypot to see if an attack has occurred or not.

2.3 Threat focus

Mainly there are three threat focuses which we should be aware of while selecting the client honeypots

Threat focus 1: Drive-by Download

- Download of malware without awareness of the user.
- Malware offered and executed through exploitation of (multiple) vulnerabilities in browser, plugin, etc [23].
- Specific vulnerabilities targeted, based on:
  - Browser (IE/Firefox)
  - JVM versions
  - Patch level operating system

Threat focus 2: Code obfuscation

> Code obfuscation
  - Hide the exploit-vector
  - Evasion of signature-based detection (AV products, Intrusion Detection Systems)
  - Examples seen for Javascript, VBScript

```
function xor_str(plain_str, xor_key){
   var xored_str = "";
   for (var i = 0 ; i < plain_str.length; ++i)
      xored_str += String.fromCharCode(xor_key ^ plain_str.charCodeAt(i));
   return xored_str;
}
var plain_str =
"\xf6\xdb\xdc\xdb\xdc\xa0\xb7\xa4....\xff\xed\xdb\xdc\xdb\xdc";
var xored_str = xor_str(plain_str, 214);
eval(xored_str);
```

Figure 2.2 Example of code obfuscation[23]

Threat focus 3: Compromised websites

Exploits imported from other servers via iframes, redirects, Javascript client side redirects

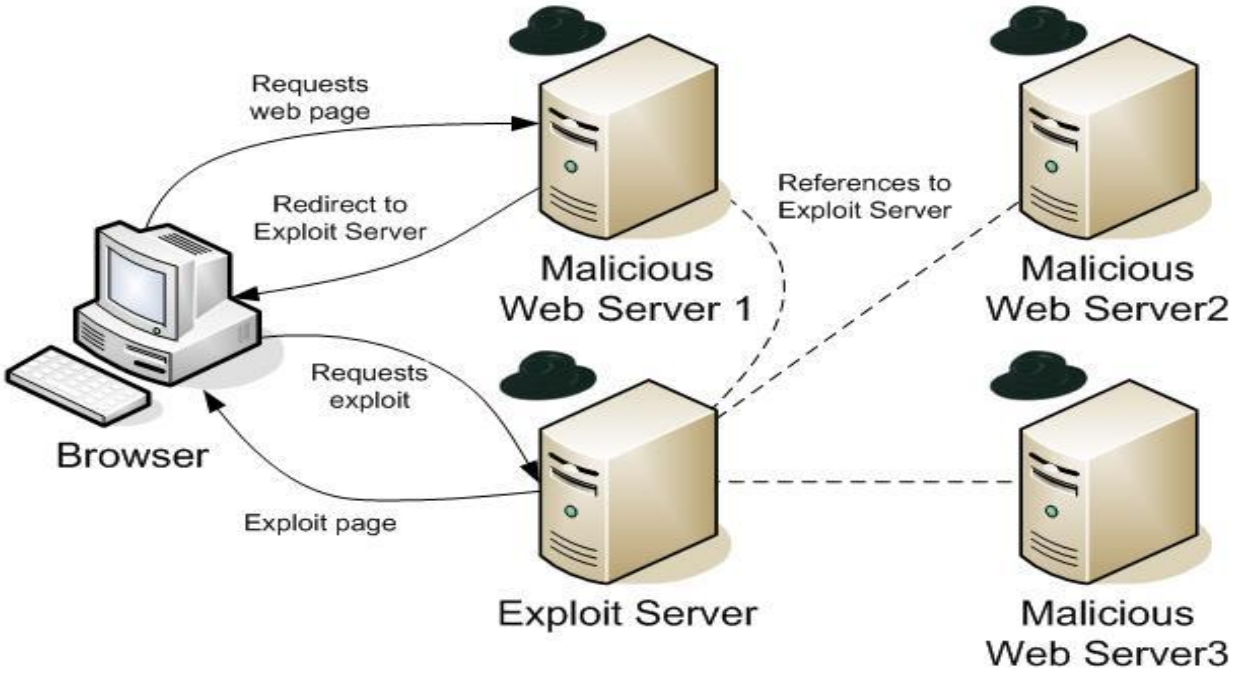

Figure 2.3 Example of Compromised websites [23]

2.4 CLIENT-SIDE ATTACKS

In order to understand client-side attacks, let us briefly describe server-side attacks that we can contrast to client-side attacks. Servers expose services that clients can interact with. These services are accessible to clients that would like to make use of these services. As a server exposes services, it exposes potential vulnerabilities that can be attacked. Merely running a server puts oneself at risk, because a hacker can initiate an attack on the server at any time. For example, an attacker could send a maliciously crafted HTTP request to a vulnerable web server and attempt to leverage errors or other unexpected application behavior.

Client-side attacks are quite different. These are attacks that target vulnerabilities in client applications that interact with a malicious server or process malicious data. Here, the client initiates the connection that could result in an attack. If a client does not interact with a server, it is not at risk, because it doesn't process any potentially harmful data sent from the server [24]. Merely running an FTP client without connecting to an FTP server, for example, would not allow for a client-side attack to take place. However, simply starting up an instant messaging application a potentially exposes the

client to such attacks, because clients are usually configured to automatically log into a remote server.

A typical example of a client-side attack is a malicious web page targeting a specific browser vulnerability that, if the attack is successful, would give the malicious server complete control of the client system. Client-side attacks are not limited to the web setting, but can occur on any client/server pairs, for example e-mail, FTP, instant messaging, multimedia streaming, etc. Client-side attacks currently represent an easy attack vector because most attention in protection technology has been focused on the protection of exposed servers from remote attackers. Clients are only protected in environments where access from internal clients to servers on the Internet is restricted via traditional defenses like firewalls or proxies. However, a firewall, unless combined with other technologies such as IPS, only restricts network traffic; once the traffic is permitted, a client interacting with a server is at risk. More advanced corporate server filtering solutions are available, but typically these only protect limited set of client technologies.

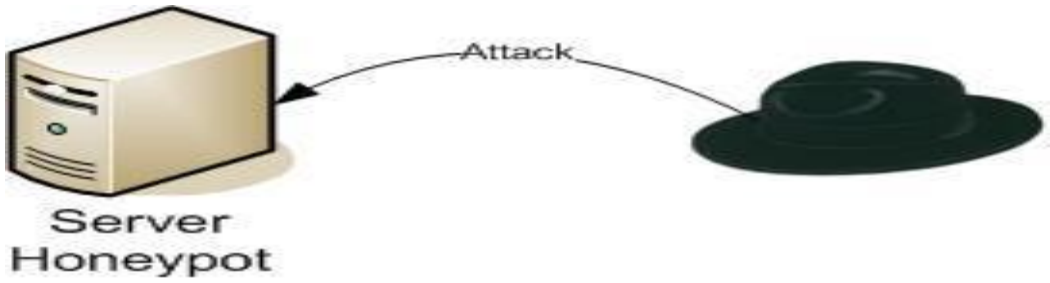

Figure 2.4 Traditional server honeypot being attacked by a “black-hat” [24]

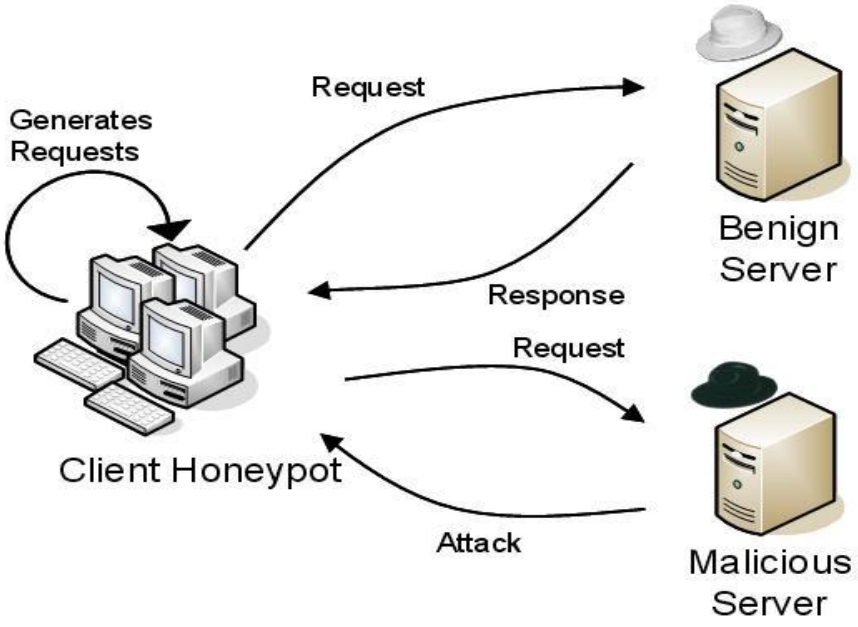

Figure 2.5 Client honeypot [24]

2.5 Server VS Client Honeypot

Traditional honeypot technology is server based and not able to detect client-side attacks. A low interaction honeypot like Honeyd, or a high interaction honeynet system with the Roo Honeywall, acts as a server, exposeing some vulnerable services and passively waiting to be attacked. However, to detect a client-side attack, a system needs to actively interact with the server or process malicious data. A new type of honeypot is therefore needed: the client honeypot. Client honeypots crawl the network, interact with servers, and classify servers with respect to their malicious nature.

The main differences between a client-side honeypot and traditional honeypot are:

- Client-side: it simulates/drives client-side software and does not expose server based services to be attacked.
- Active: it cannot lure attacks to itself, but rather it must actively interact with remote servers to be attacked.
- Identify: whereas all accesses to the traditional honeypot are malicious, the client-side honeypot must discern which server is malicious and which is benign.

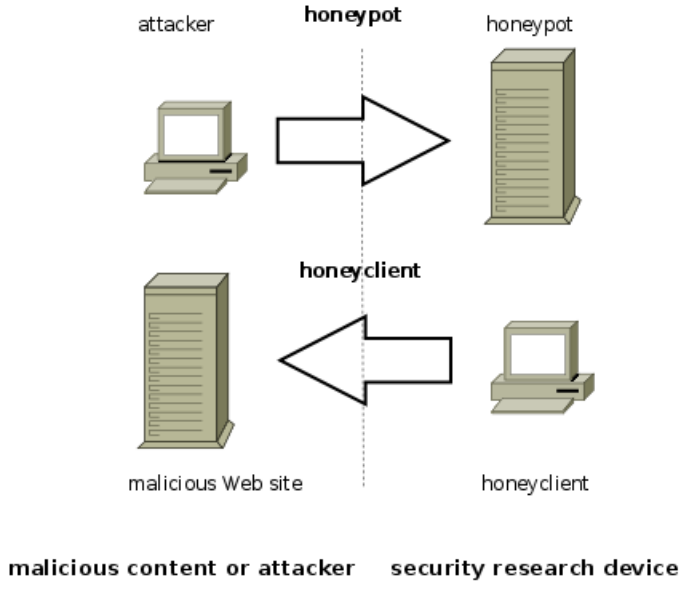

Figure 2.6 Honeyserver Vs Honeyclient [24]

Similarly to traditional server honeypots, there are two types of client honeypots: low and high interaction client honeypots. The low interaction client honeypot uses a simulated client (for example HoneyC or wget in the case of a browser-based client honeypot), interacts with servers, and classifies the servers based on some established definition of "malicious" activity. Usually this is performed via static analysis and signature matching. Low interaction client honeypots have the benefit of being quite

fast, but can produce false alerts or miss a malicious server, especially since they do not act like a "real" client and have programmatic limitations. They may also fail to fully emulate all vulnerabilities in a client application. The other type of client honeypot, the high interaction client honeypot, takes a different approach to make a classification of malicious activity. Using a dedicated operating system, it drives an actual vulnerable client to interact with potentially malicious servers. After each interaction, it checks the operating system for unauthorized state changes. If any of these state changes are detected, the server is classified as malicious. Since no signatures are used, high interaction client honeypots are able to detect unknown attacks.

In Honeyclients we are looking at malicious web servers that attack web browsers. Attacks on web browsers by malicious web servers seem to be the most prominent client-side attack type today, but they are still not well understood. The goal is to assess the threat to web browser client applications from malicious web servers with a high interaction client honeypot. In this a high interaction client honeypot, because it is used allows us to obtain information about attacks that are unknown or obfuscated in a way that low interaction client honeypots could not detect. We obtain and present information on malicious web servers, evaluate several defense mechanisms, and make recommendations on how to protect systems against these malicious web servers.

2.5.1 High interaction client honeypots

High interaction client honey pots are fully functional systems comparable to real systems with real clients. As such, no functional limitations (besides the containment strategy) exist on high interaction client honey pots. Attacks on high interaction client honey pots are detected via inspection of the state of the system after a server has been interacted with. The detection of changes to the client honey pot may indicate the occurrence of an attack against that has exploited a vulnerability of the client. An example of such a change is the presence of a new or altered file.

High interaction client honey pots are very effective at detecting unknown attacks on clients. However, the tradeoff for this accuracy is a performance hit from the amount of system state that has to be monitored to make an attack assessment. Also, this detection mechanism is prone to various forms of evasion by the exploit. For example, an attack could delay the exploit from immediately triggering (time bombs) or could trigger upon a particular set of conditions or actions (logic bombs). Since no immediate, detectable state change occurred, the client honey pot is likely to incorrectly classify the server as

safe even though it did successfully perform its attack on the client. Finally, if the client honey pots are running in virtual machines, then an exploit may try to detect the presence of the virtual environment and cease from triggering or behave differently.

Capture-HPC

Capture is a high interaction client honeypot developed by researchers at Victoria University of Wellington, NZ. Capture differs from existing client honey pots in various ways. First, it is designed to be fast. State changes are being detected using an event based model allowing to react to state changes as they occur. Second, Capture is designed to be scalable. A central Capture server is able to control numerous clients across a network. Third, Capture is supposed to be a framework that allows to utilize different clients. The intitial version of Capture supports Internet Explorer, but the current version supports all major browsers (Internet Explorer, Firefox, Opera, Safari) as well as other HTTP aware client applications, such as office applications and media players [25]

HoneyClient

HoneyClient [2] is a web browser based (IE/FireFox) high interaction client honeypot designed by Kathy Wang in 2004 and subsequently developed at MITRE. It was the first open source client honeypot and is a mix of Perl, C++, and Ruby. HoneyClient is state-based and detects attacks on Windows clients by monitoring files, process events, and registry entries. It has integrated the Capture-HPC real-time integrity checker to perform this detection. HoneyClient also contains a crawler, so it can be seeded with a list of initial URLs from which to start and can then continue to traverse web sites in search of client-side malware [26].

HoneyMonkey

HoneyMonkey [3] is a web browser based (IE) high interaction client honeypot implemented by Microsoft in 2005. It is not available for download. HoneyMonkey is state based and detects attacks on clients by monitoring files, registry, and processes. A unique characteristic of HoneyMonkey is its layered approach to interacting with servers in order to identify zero-day exploits. HoneyMonkey initially crawls the web with a vulnerable configuration. Once an attack has been identified, the server is reexamined with a fully patched configuration. If the attack is still detected, one can conclude that the attack utilizes an exploit for which no patch has been publicly released yet and therefore is quite dangerous [27].

SHELIA

Shelia is a high interaction client honeypot developed by Joan Robert Rocaspana at Vrije Universiteit Amsterdam. It integrates with an email reader and processes each email it receives (URLs & attachments). Depending on the type of URL or attachment received, it opens a different client application (e.g. browser, office application, etc.) It monitors whether executable instructions are executed in data area of memory (which would indicate a buffer overflow exploit has been triggered). With such an approach, SHELIA is not only able to detect exploits, but is able to actually ward off exploits from triggering.

UW Spycrawler

The Spy crawler developed at the University of Washington is yet another browser based (Mozilla) high interaction client honeypot developed by Moshchuk et al. in 2005. This client honeypot is not available for download. The Spycrawler is state based and detects attacks on clients by monitoring files, processes, registry, and browser crashes. Spycrawlers detection mechanism is event based. Further, it increases the passage of time of the virtual machine the Spycrawler is operating in to overcome (or rather reduce the impact) of time bombs.

Web Exploit Finder

WEF is an implementation of an automatic drive-by-download – detection in a virtualized environment, developed by Thomas Müller, Benjamin Mack and Mehmet Arziman, three students from the Hochschule der Medien (HdM), Stuttgart during the summer term in 2006. WEF can be used as an active Honey Net with complete virtualization architecture underneath for rollbacks of compromised virtualized machines [28].

2.5.2 Low interaction Client Honeypots

Low interaction client honey pots differ from high interaction client honey pots in that they do not utilize an entire real system, but rather use lightweight or simulated clients to interact with the server. (in the browser world, they are similar to web crawlers). Responses from servers are examined directly to assess whether an attack has taken place. This could be done, for example, by examining the response for the presence of malicious strings.

Low interaction client honey pots are easier to deploy and operate than high interaction client honey pots and also perform better. However, they are likely to have a lower

detection rate since attacks have to be known to the client honey pot in order for it to detect them; new attacks are likely to go unnoticed. They also suffer from the problem of evasion by exploits, which may be exacerbated due to their simplicity, thus making it easier for an exploit to detect the presence of the client honey pot.

HoneyC

HoneyC is a low interaction client honey pot developed at Victoria University of Wellington by Christian Seifert in 2006. HoneyC is a platform independent open source framework written in Ruby. It currently concentrates driving a web browser simulator to interact with servers. Malicious servers are detected by statically examining the web server's response for malicious strings through the usage of Snort signatures [29].

Monkey-Spider

Monkey-Spider is a low-interaction client honeypot initially developed at the University of Mannheim by Ali Ikinci. Monkey-Spider is a crawler based client honeypot initially utilizing anti-virus solutions to detect malware. It is claimed to be fast and expandable with other detection mechanisms. The work has started as a diploma thesis and is continued and released as Free Software under the GPL [30].

PhoneyC

PhoneyC is a low-interaction client developed by Jose Nazario. PhoneyC mimics legitimate web browsers and can understand dynamic content by de-obfuscating malicious content for detection. Furthermore, PhoneyC emulates specific vulnerabilities to pinpoint the attack vector. PhoneyC is a modular framework that enables the study of malicious HTTP pages and understands modern vulnerabilities and attacker techniques [31].

SpyBye

SpyBye is a low interaction client honeypot developed by Niels Provos. SpyBye allows a web master to determine whether a web site is malicious by a set of heuristics and scanning of content against the ClamAV engine [32].

2.5.3 Hybrid Client Honeypots

Hybrid client honeypots combine both low and high interaction client honeypots to gain from the advantages of both approaches.

HoneySpider

The HoneySpider network is a hybrid client honeypot developed as a joint venture between NASK/CERT Polska , GOVCERT.NL and SURFnet. The projects goal is to develop a complete client honeypot system, based on existing client honeypot solutions and a crawler specially for the bulk processing of URLs [33].

Strider Honey Monkey

Strider HoneyMonkey is a Microsoft Research project to detect and analyze Web sites hosting malicious code. The intent is to help stop attacks that use Web servers to exploit unpatched browser vulnerabilities and install malware on the computers of unsuspecting users. Such attacks have become one of the most vexing issues confronting Internet security experts. Strider HoneyMonkey is a project of the Cybersecurity and Systems Management group in Microsoft Research (which has grown to become today's Internet Services Research Center - ISRC) [34].

Honey wares

A new low interaction client honey pot tool which aims to combine the benefits of web-based technology that run on local or remote servers, it gives the user the ability to scan the target server with some of web browsers and to scan the target with five different scan engines.

Table I shows the main differences among low and high interactive honeyclients.

TABLE I. LOW AND HIGH INTERACTIVE HONEYCLIENTS COMPARISON

Honeyclient Type / Attribute	High Interactive	Low Interactive
Interaction Level with Attacker	High	Low
Risk	High	Low
0-Day Exploits Detection	Yes	No
Detection Rate	High	Low
Speed	Low	High
Collected Data	Rich	Limited
Cost and Maintenance	High	Low
Containment	Needed	Not Needed
Deployment and Maintenance	Difficult	Easy
False Negative Rate	Low	High
Usability	Need Skills	Easy

Table 2.1 Low and high interaction honeyclients comparison [35]

Ram Kumar Singh and Prof. T. Ramanujam stated in "*Intrusion Detection System Using Advanced Honeypots*" present the design and implementation of a load balancer that distinguishes between the traffic coming from clients and the traffic originated from the attackers. This system is an attempt to simultaneously solve the problems of load balancing and unauthorized intrusion. If, in the process of forwarding requests, the balancer detects traffic as an attack on the server ('an exploit'), it is then directed to an alternative server - a type of honeypot [36].

Y.Alosefer and O.F.Rana stated in "*Honeyware: a web-based low interaction client honeypot"* that attackers are now targeting client applications such as web browsers and media players, whereas they were previously primarily targeting servers. Therefore, there is a requirement to determine their modes of attack and to capture their malicious scripts and tools to analyse and improve security. Instead of passively waiting for attackers, an active honeypot will go and search for the attackers. Honeyware is a new low interaction client honeypot tool which aims to combine the benefits of web-based tools that run on local or remote servers with the ability to access the tool from a web browser, which is important for many different devices such as PCs or mobiles. Another important function of this tool is that it gives the user the ability to test the target server with almost all the available web browsers and to scan the target with five different scan engines. In the future it will also scan the target with an intrusion detection system such as Snort [37].

Xiaoyan Sun, Yang Wang, Jie Ren, Yuefei Zhi and Shengli Liu stated in "*Collecting Internet Malware Based on Client-side Honeypot"* that the client-side web attack techniques are applied to attacks based on browsers and email client. There are 3 kinds of attack techniques: Code Obfuscation, URL Redirection and Vulnerability Exploitation. They use the active ability of client-side honeypot to collect malware that traditional honeypot can not get in the Internet. Introduced the category of Internet malware, the clientside attack techniques and overall framework of the system in detail. Researchers gave the design and implementation of data acquirement and the honeypot deploying and designed crawler according to client-side attack techniques, and implemented the device-drive monitor and the server receiver tool. However, need to get more data source and test them in our honeypot [38].

Mahmoud T. Qassrawi and Hongli Zhang stated in "*Using Honeyclients to Detect Malicious Websites"* that Honeyclient is a new technology to find and identify malicious websites. Honeyclient technology can be used to detect malicious websites. Honeyclients comparison attributes are different from IDSs as they are not prone to being overwhelmed by the responses they receive. Researchers introduced evaluation approach to compare honeyclients. The comparison methodology can be generalized to evaluate various types of honeyclients. The comparison showed that the relation between low interactive and high interactive honeyclients technologies should be complementary rather than competitive. Moreover, open source Honeyclients such as HoneyC and Capture have an advantage over commercial tools as they are performing check in real time, thus false positives rate are expected to be lower. Although honeyclients can be effective tool to detect malicious attacks, there are various techniques can be used by malicious websites to counter and evade honeyclients [39].

Saurabh Chamotra, Rakesh Kumar Sehgal, Raj Kamal and J.S.Bhatia proposed a method in the paper title "*Data Diversity of a Distributed Honey Net Based Malware Collection System*" that The value of a Data collection mechanism like Honeypot/Honeyntes lies in being attacked and probed.Hence the efficiency of these resources depends upon the amount and value of data collected by them but then there is no appropriate measure present to quantify the value of data collated by these systems. Most of the honeynet projects proves the efficiency of their honeynet systems based upon the volume of data collected but then the volume of data in it self could be a misleading parameters as in the case where a honeypot collects a high volume of the data but the data lacks in the diversity as it collects the same

attacks in a given time frame again and again from different data sources, different nodes deployed with same configuration for similar duration in different ISPs greatly varies in the value of data collected by them in terms of the diversity. although the amount of data collected by some of the nodes was almost same but when evaluated on the diversity scale the value of data collected by them were greatly different and some of the nodes proved to be the better collector than the other one. Also it was deduced from the experiments that the data collected by a distributed system as a whole is more diverse than the data collection done by a standalone honeynet system. Hence this proves that distributed honeynet systems are able to collect much diverse data [40].

Chapter 3

PROBLEM STATEMENT

Nowadays, security system is very important to any organization to protect their data or any information kept in their computer from the intruders to access. Unauthorized user is able to connect to the organization's computers and control it in some form to view or access the files. Many of us know how to use the computer but do not have enough information to secure the computer especially for the system administrators.

Using malicious sites to launch attacks against client user applications is a growing threat in recent years. This led to emergence of new technologies to counter and detect this type of client-side attacks. One of these technologies is honeyclient. Honeyclients crawl the Internet to find and identify web servers that exploit client-side vulnerabilities.

This is exactly what we will be implementing to get the internal things about the client honeypots.

With the improvement of software security, attacks based on RPC vulnerabilities declined, however, attacks based on client application software vulnerabilities have increased. Such client application software includes web browsers, Email client and Office. The spread of malware using these software vulnerabilities has become a severe threat to today's Internet. In allusion to this kind of threat, we have tried to develop a prototype system to collect the internet malwares by actively visiting the malicious websites using client honeypots. This system can not only collect malware but also detect malicious website. Here when we are visiting the websites in a virtual machine, we monitor the activities such as file system, network monitor etc. The end results of the system are collected malware executable binaries, PCAP network data.

3.1 Goal of Client Honeypot

The ultimate goal of client honeypots is to detect and identify any malicious activity coming from the Internet. This ideal case of client honeypot can be summarized as follows:

- To study and understand the working of Honeyclient.
- To design and implement the virtual honeyclient on any open source Operating system.
- To deploy and test the working of virtual honeyclient to track the attackers.
- Client honeypot should detect any known and unknown threats against any client user application. Application can be any server/client based application. Client honeypot should be able to check various URLs (images, executable files, html, scripts, etc) . Ideal client honeypot has rate zero false positive.
- Client honeypot should detect the attacks in real-time.

Chapter 4

Implementation Details and Experimental Results

This chapter disusses about how the problem stated in previous chapter can be solved with the help of functional diagram of virtual client Honeypot.

Fig. 4.1 shows the process of malicious websites detection by honeyclients. Honeyclient should be provided with seeds to start the search. For example, we can start the search with "interesting" keywords or links extracted from phishing sites and spam mails. The next step is accessing the web to search for servers that exploit the client, to gather information about these websites exploits.

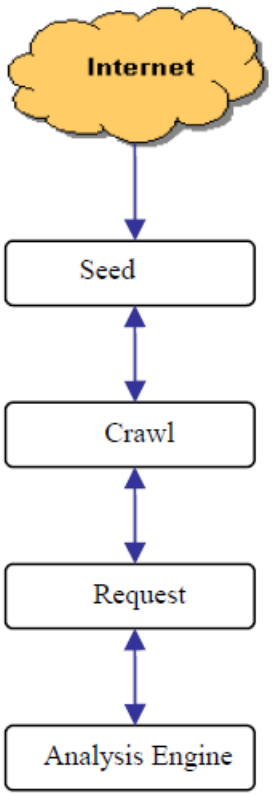

Figure 4.1 Process of Honeyclient [41]

A. Crawling Phase

Honeyclients are faced with crawling the web with its millions of servers. Honeyclient needs firstly to actively search a network to find servers to interact with. This phase is common for all honeyclients; we must firstly presort the parts of Internet we want to inspect.

This determines the search scopes of inspected web sites. Using crawling techniques should satisfy three requirements:

• Obtaining high speed.

• Avoiding overloading.

• Avoiding sample bias.

Crawling speed is dependent of hardware, network bandwidth, and crawling algorithm.

The second issue is information overloading. Modernweb sites are machine-generated sites which use the URL as a transport layer for information. Crawling should avoid crawling of the same content, and should keep the coordination of the crawler threads. Crawling policies are important to improve the performance of the overall crawl, and on the other hand not to overload the crawled hosts. For example, MonkeySpider uses URL normalization techniques to avoid unnecessary overhead, while downloading the same content multiple times. The third issue is sample bias [41]. There are various schemes can be used to obtain and presort website to inspect. A. Moshchuk T. Bragin, S. Gribble, and H. Levy presented a queuing scheme that depends on the premise that malicious websites contain particular type of contents such as pornography and create queues of web pages by querying search engines with query suspicious keywords or extracting links from suspicious sources, such as SPAM emails. However, this scheme will lead to missing many malicious websites, as attacker scould use less suspicious words to build their malicious websites. Moreover, this approach can not cover large scope of internet as many website will not be included in this scheme. The second method is web crawling. In this method, we use the hyperlink structure on retrieved documents to access other web pages. It is the primary means for web search engines to retrieve web pages for their index creation. However, such method has a shortcoming; since popular web pages are likely to be linked with higher rate, a random crawl would lead to bias into the sample. Various researches are concerned with addressing such

bias by statistical adjustments based on page popularity. While bias is reduced, it is not removed as these adjustments seem to leave bias toward pages with high number of links on other pages that point to this page, the so called in-degree. Further, this approach would miss pages for which no hyperlinks exist. The third method is by generating random IP address and checks the presence of web server, crawling the website hosted on the web server and selecting a page from it at random. This approach was presented by M. Pennock, S. Lawrence, and L. C. Giles, to evaluate the coverage of a web search engine's indices. This approach will create the most unbiased random sample from the solutions proposed. Nevertheless, some pages will be missed from the sample. In particular, pages that are not linked to from a starting page or on web servers on which no starting page exist. Pages extracted from link in SPAM message, IM messaging will also be missed by this approach. However, external spam message dataset archive can be used along with this approach.

B. Identification Phase

After sending request to the crawled sites, honeyclient analyzes the response to identify whether the servers are malicious or benign. Two approaches are used by honeyclients to identify malicious website:

• *Pattern-matching*: it is used by *low-interaction honeyclient*.

• *State Change Check (Integrity Check)*: it is used by *high-interaction honeyclient.*

Low-interaction honeyclients do not use full functional operating system or web browser, instead they use simulated client. Low-interaction honeyclients are often emulated web browsers, or web crawlers, which do have no or only limited abilities for attackers to interact with. Low-interaction honeyclients send HTTP requests to the web server, and detect malicious servers by applying signature-based or heuristic methods on the server response for a fast analysis. They can directly detect the security violation by pattern matching methods, which means applying static signature or heuristics based method on web server's response. *HoneyC* , *Monkey- Spider* and *PhoneyC* are examples of low interaction honeyclients which use pattern-matching. High-interaction honeyclients give an attacker the capability to interact with real system rather than simulation. They detect the security violations via state changes check; which means the need to monitor filesystem, registry entries, processes, network connection and physical resources such as memory and CPU, etc. State change checks should

give first insight into whether a system has been compromised. There are various honeyclients developed based on this approach such as *Capture-HPC* , *HoneyClient* and *HoneyMonkey*.

4.1 Design Architecture of Virtual Honeyclient

High-interaction honeyclients give an attacker the capability to interact with real system rather than simulation. They detect the security violations via state changes check; which means the need to monitor filesystem, registry entries, processes, network connection and physical resources such as memory and CPU, etc. State change checks should give first insight into whether a system has been compromised. There are various honeyclients developed based on this approach such as *Capture-HPC*, *HoneyClient* and *HoneyMonkey*. Fig. shows an overview of high-interaction honeypots. Via configuration front-end, the user can adjust various parameters, as keywords to search for web pages with search engines, depth and breadth of crawling, or number of URLs after which the client honeypots stop its execution. Browser simulator module simulates web browser application, URL analyzer should handle dialog box and other techniques could be used to evade honeypot. Integrity check module checks the state changes of the client system to detect any changes of the system. All log files are stored in remote database to enable centralized logging. Client honeypots operating in different networks can report their findings to central sites that can also correlate the data. The analysis front-end enables all data analysis, to help the operator to keep track of collected information [42].

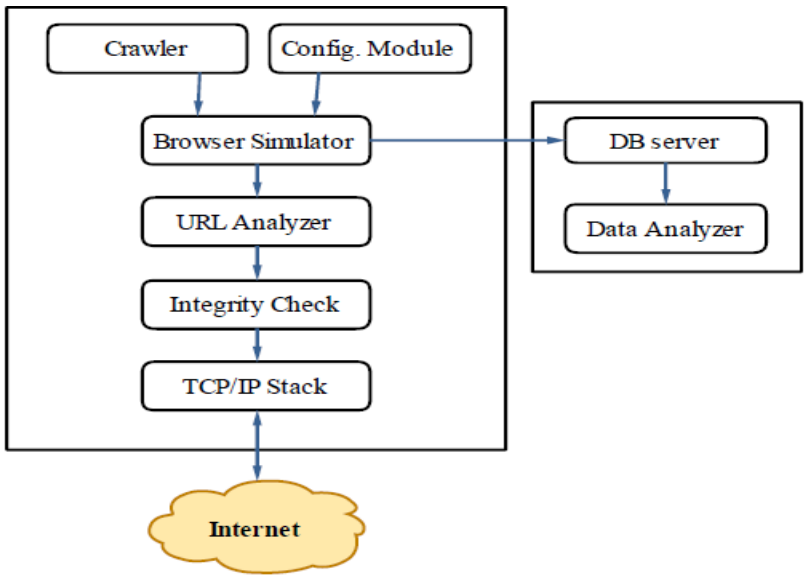

Figure 4.2 Overview of High Interaction Honeypots [42]

4.2 Functional Diagram client honeypot

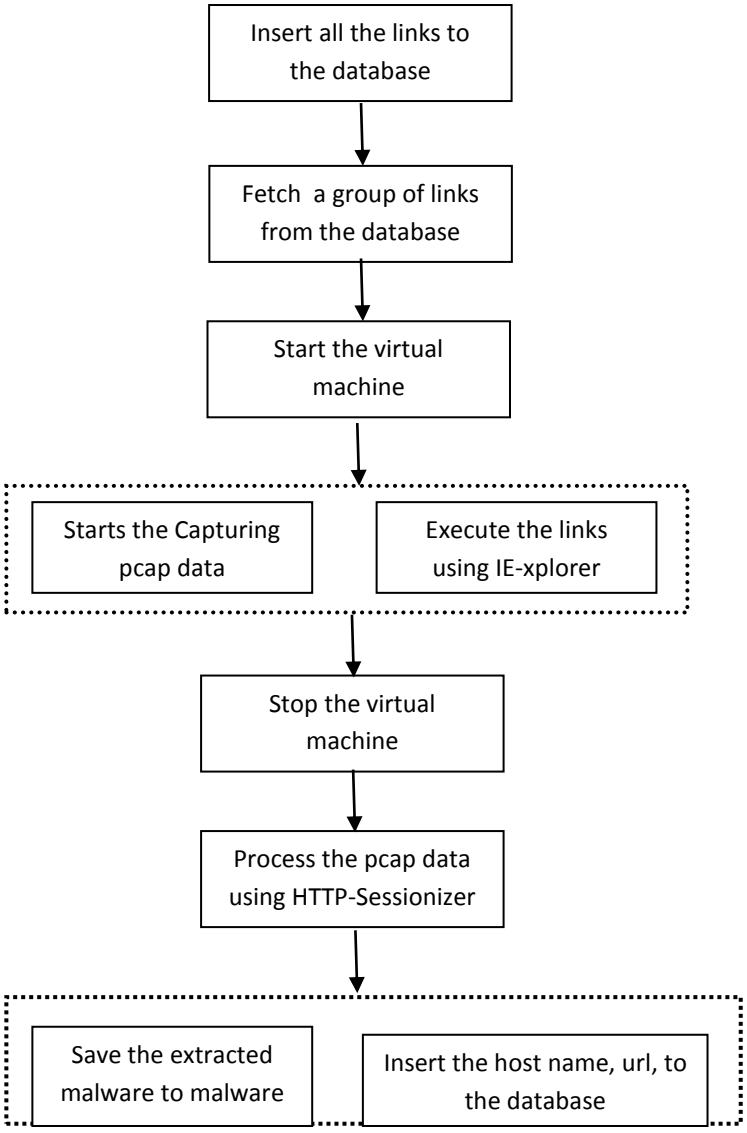

Figure 4.3 Functional Diagram of Honeyclient

In the implementation of virtual client honeypot, we have used linux red hat as base machine and Virtual Box based honeypot for browsing of URLs and monitoring file system, network activities. Firstly, we manually feed the URL's in the log file which we want to check for malwares or we can a crawler to collect web page URLs, and store them in a database. After that when we fetch the links from the database and start the virtual machine. The machine starts to open these fetched links one by one and MwWatcher tool infections. We have set the

execution of each site for 90 sec. Also we use the DCHSniffer for capturing PCAP data. After all the processing has been done virtual machine stops and all the executable and binary files be shown on the base machine with the URL from where they came. Then analysis and reporting, we are inserting the mailicious URLs into database. We have also used bridge-util is used for creation of bridge, gcc compiler is GNU C compiler used in linux platform, HTTP: sessionizer is for re-session of http communication and Fuse util is being used for virtual file system.

4.3 MAKING A START

I started by trying to implement "Honeyclient" because this is the most widely used honeypot and also the one most extensively documented. The aim was to start out with something, on which help is easily available, get the hang of how things work , see what initial set of problems I face and as I get comfortable with the whole technology and working move on to something more extensive.

My system consisted basically of a Redhat Linux-9 distribution running through virtualization software i.e. "Virtual box" Windows 7 professional having the following minimum configuration.

4.3.1 Design of Implemented Set up:

Base Machine (Window) ---> Window XP in Virtual Box.

Base Machine:

- Red Hat Enterprise edition 5
- Ntfs-3g
- Brigde-util
- Fuse util
- Mysql database
- HTTP:sessinonizer
- Gcc complier
- Tcpdump software

Virtual Machine:

- Window XP.
- Internet Explorer as client application
- Mwatcher for file system monitoring

4.3.2 Why Virtual Honeypot?

Well the reasons for opting for a virtual honeypot were obvious it requires fewer resources , meaning that all I would require would be one PC on which I could setup multiple honeypots independent of each other and probably after getting the insight even go for a virtual honeynet.

4.3.3 Features of Client Honeypot

Client honeypot should detect any known and unknown threats against any client user application.

Application can be any server/client based application. Client honeypot should be able to check various URLs (images, executable files, html, scripts, etc).

- Ideal client honeypot has rate zero false positive.
- Client honeypot should detect the attacks in real-time.
- Client honeypot should be able to dynamically modify the detection and security policy rules to fit the current situation .
- Network packets are being logged and dumps are being created
- Honeypot model is helpful in improving the defensive mechanism

4.3.4 MwWatcher

MwWatcher runs within the virtualized high-interaction honeypot and logs suspicious modifications to the filesystem. MwWatcher malware collection tool is a program which monitors honeypot file system changes in real time and catches potential malware. It currently only runs on a Win32 guest system. MwWatcher is one of the three malware collection tools implemented in the HoneyBow toolkit. It is based on the essential feature of honeypot – no production activity – and watches the file system for suspicious activity caused by malware infections in real time. The tool exploits a characteristic feature of propagating malware: when malware successfully exploits a vulnerable service then infects the honeypot. The malware sample will replicate, attach and stores itself in the file system of the victim. MwWatcher will then detect this change of the filesystem and obtain a binary copy of the malware sample. This sample is then moved to a hidden directory awaiting further collection by another tool called.

4.4 Performance and Results: In my thesis work we build a test setup “Client Based Honeypots” which shows the log file in the database log and we can check these logs on the base machine through which we check the pcap logs and dumps.

Step 1 In this step we give command “neat” to set Ip to the Base Machine

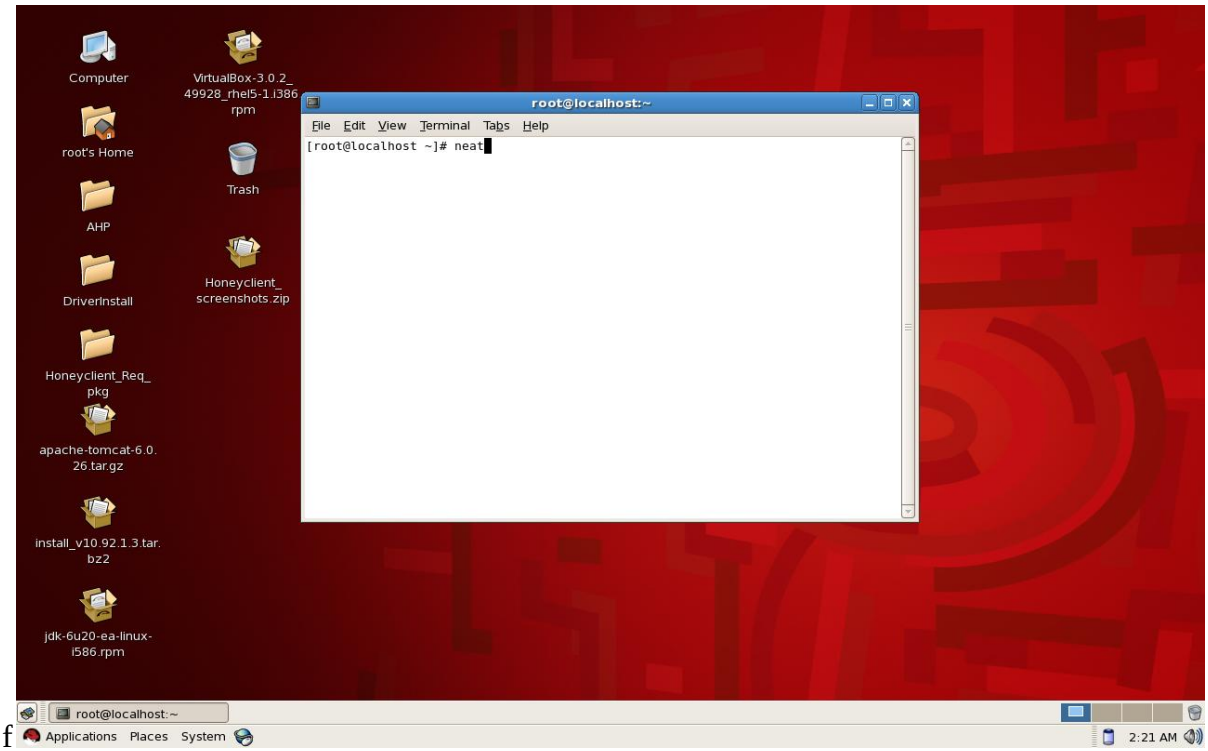

Figure 4.4 Command to open Ip Fields

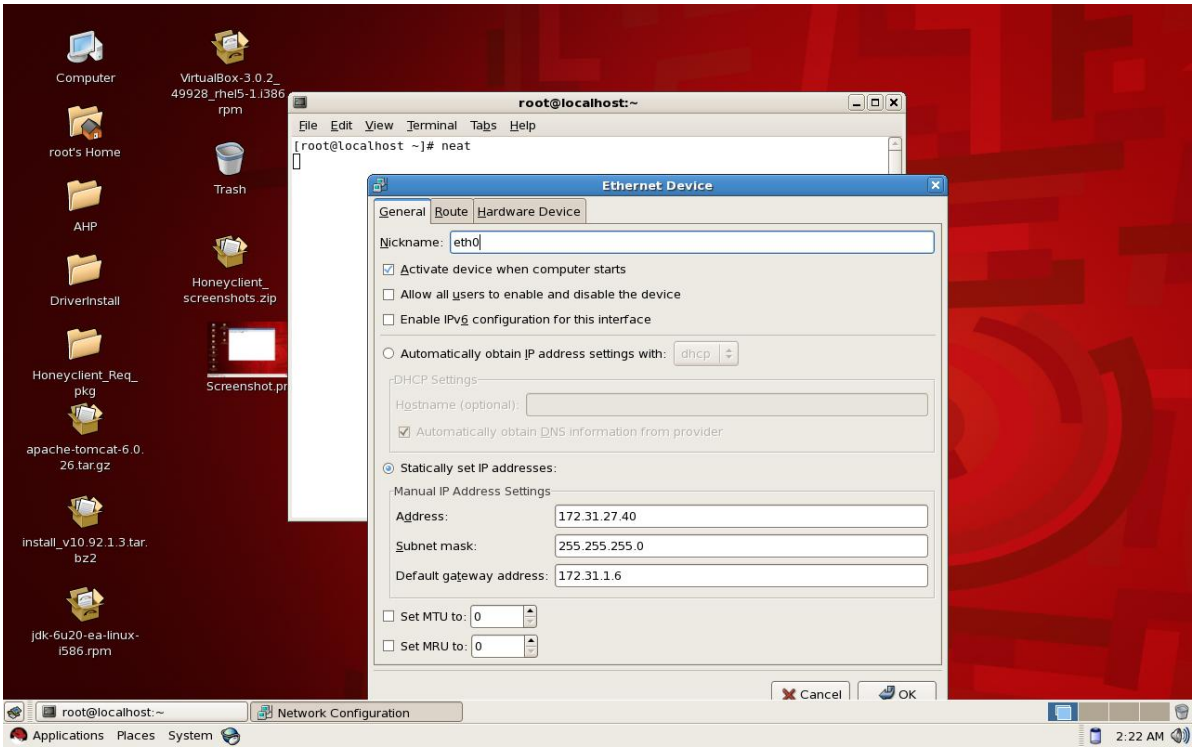

Figure 4.5 Setting Statically Ip Address

Step 2 In this Step we open the Sun virtual box and set Ip address to that machine also.

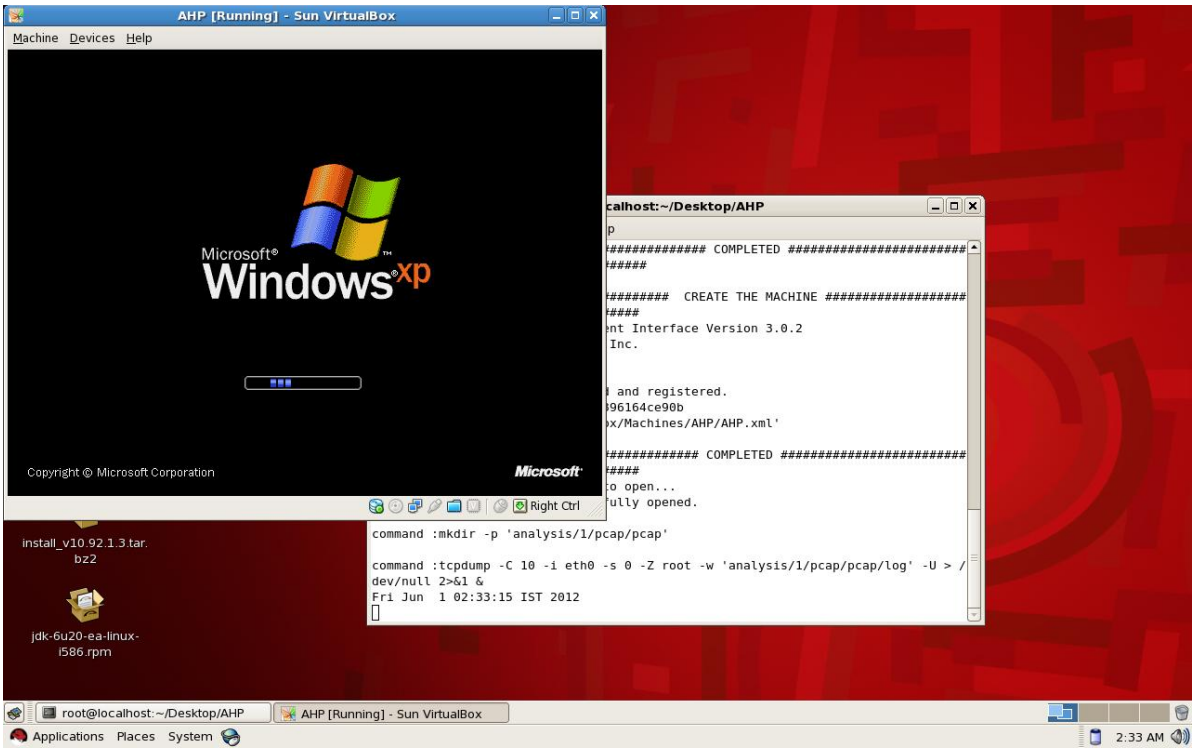

Figure 4.6 Opening Sun Virtual Box

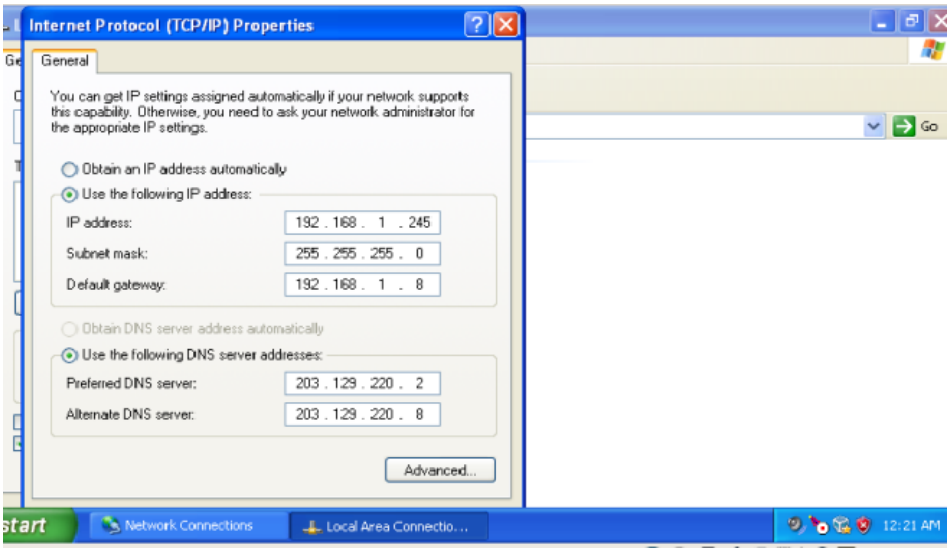

Figure 4.7 Setting IP Adress Statically In Virtual Machine

Step 3: In this step we used a command "cd /root/Desktop/AHP" in the root terminal to open the files in the AHP folder.

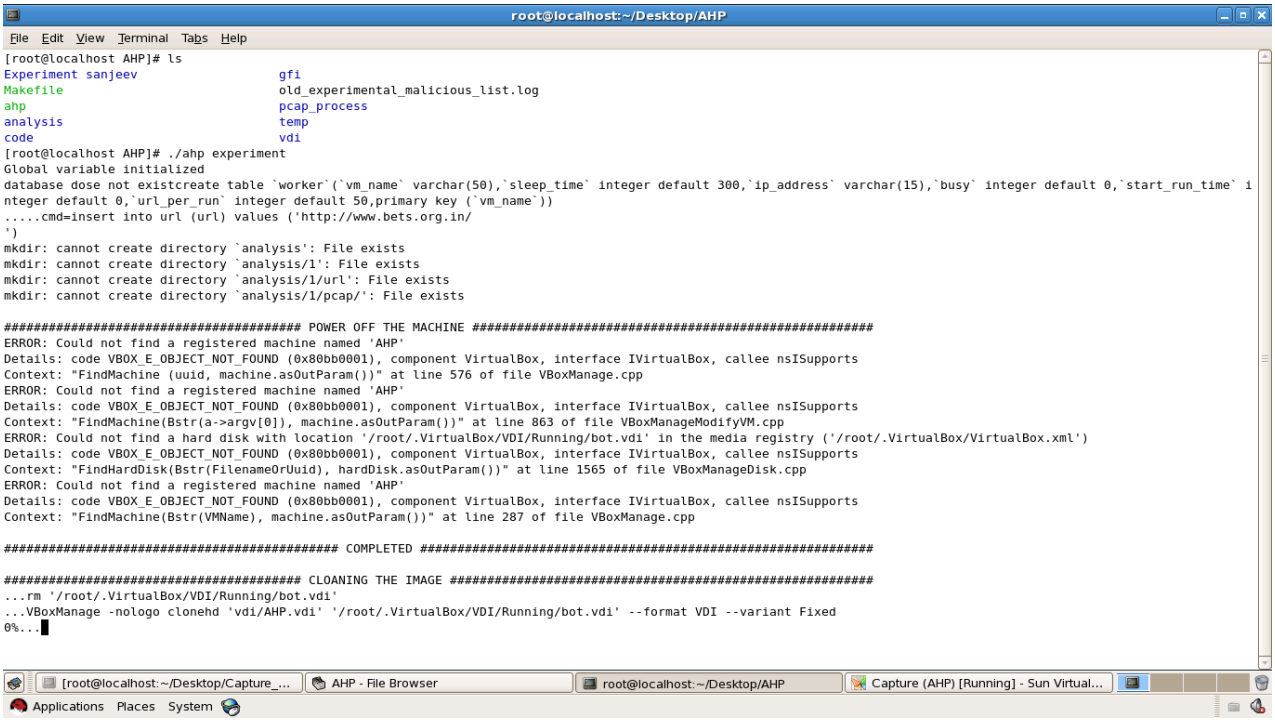

Figure 4.8 Command to open AHP folder

Step4: In the next step we used a command "./ahp experiment.log" to start the working of our test setup

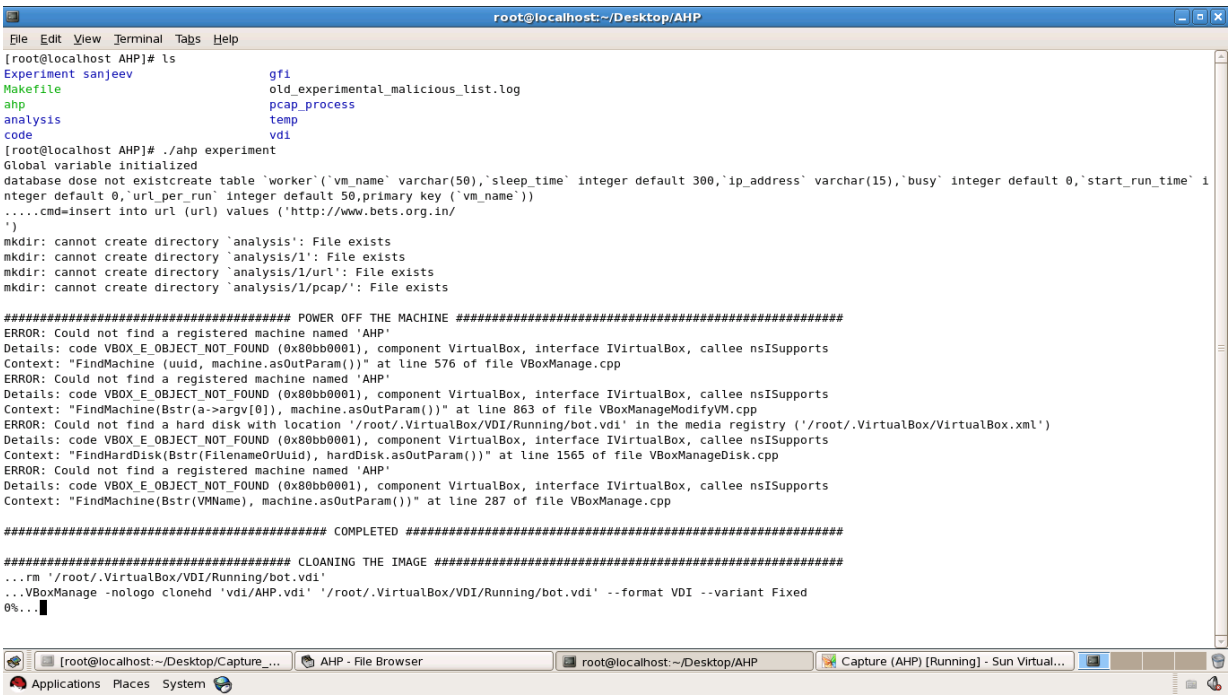

Figure 4.9 Opening file of AHP to generate setup

Step 5 : Processing to open the virtual machine

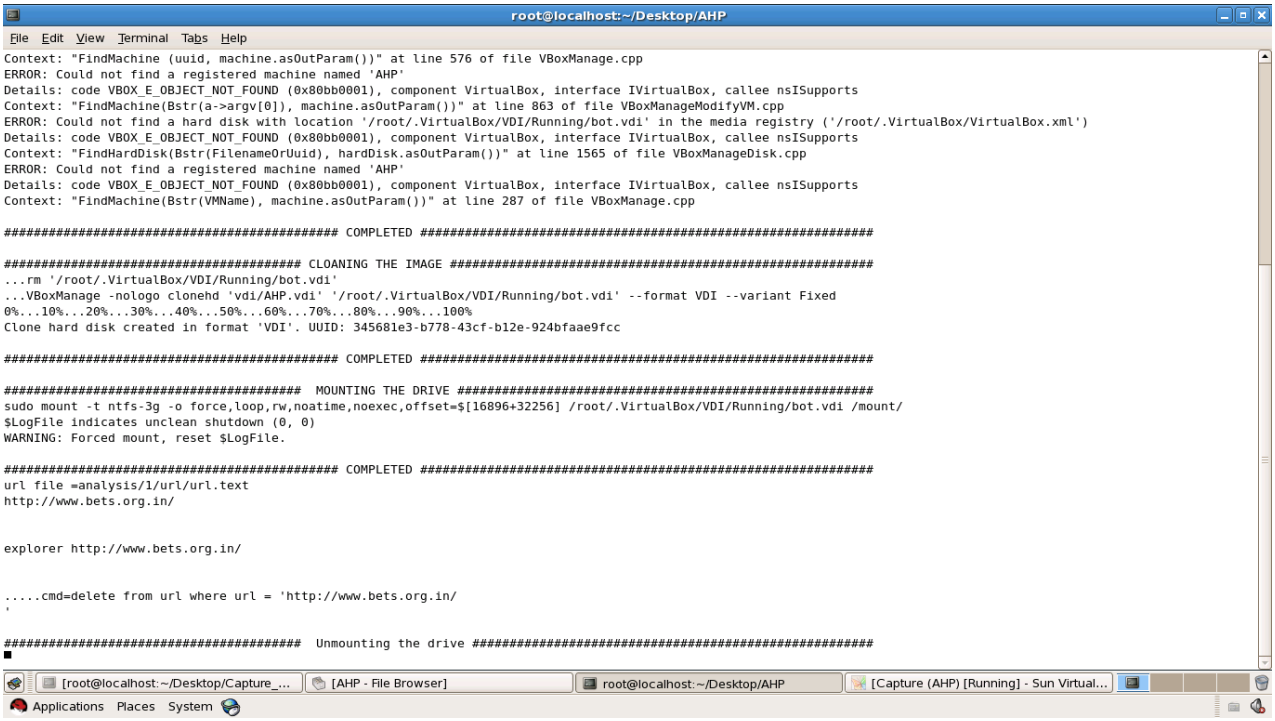

Figure 4.10 Processing to popup virtual box

Step 6 In this step we used a command "service mysqld start" in the root terminal and give username and password to open the database .

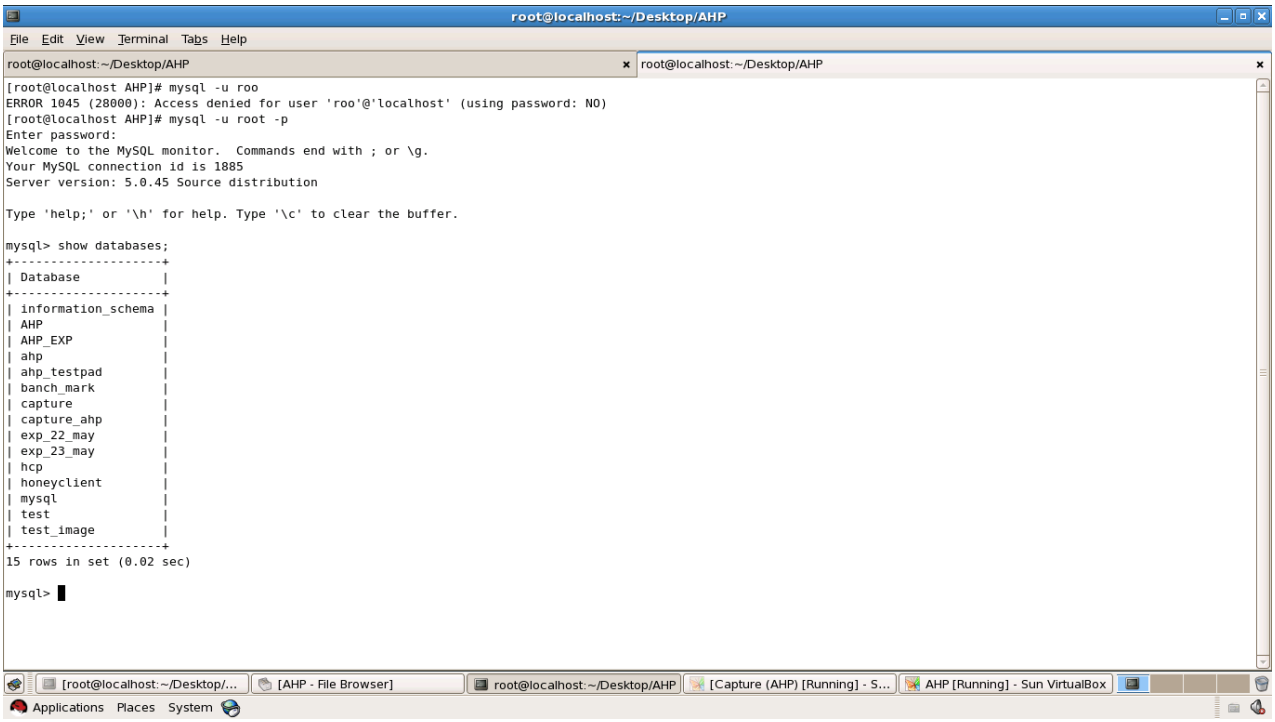

Figure 4.11 Opening SQL database

Step 7 In this step processing is done and setup is examining the urls saved in the experiment.log file.

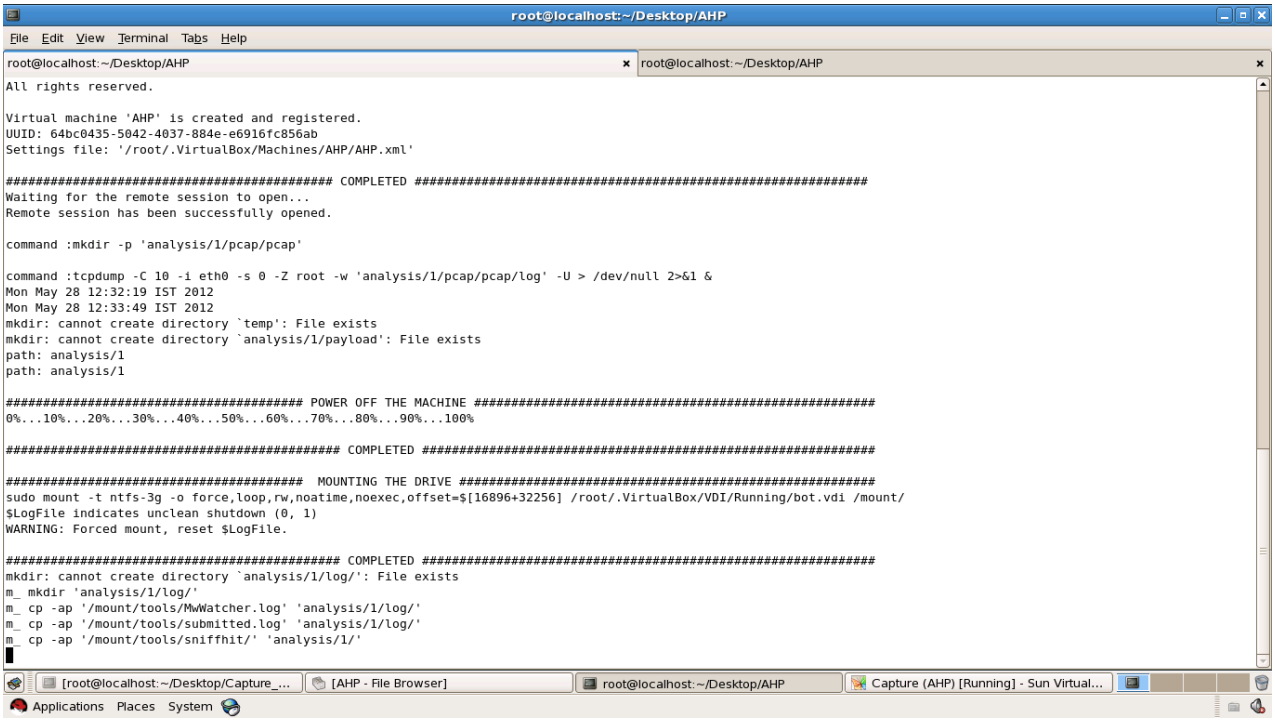

Figure 4.12 Examining URL's Stored in file

Step 8 In this step the processing is done and Sun virtual box is automatically come up.

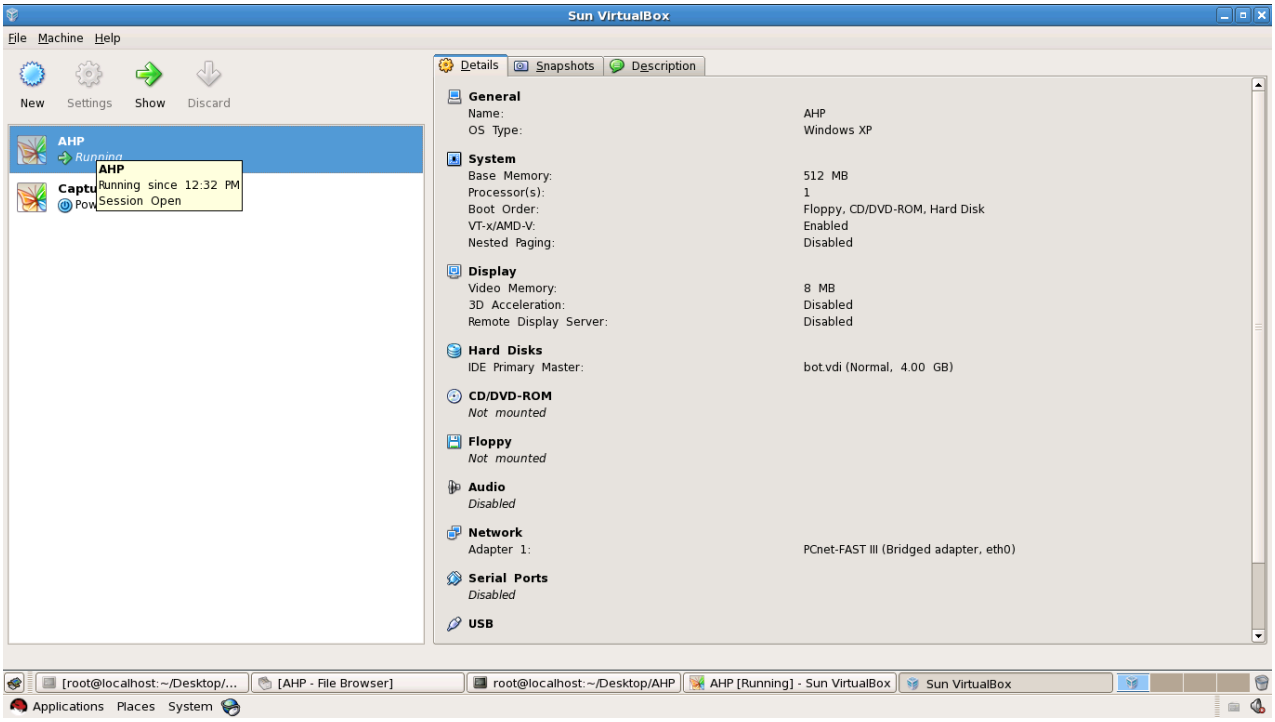

Figure 4.13 Popup of Virtual box automatically after processing is done

Step 9 Now the virtual machine opens the websites given in the database one by one if there is ip configured to that machine.

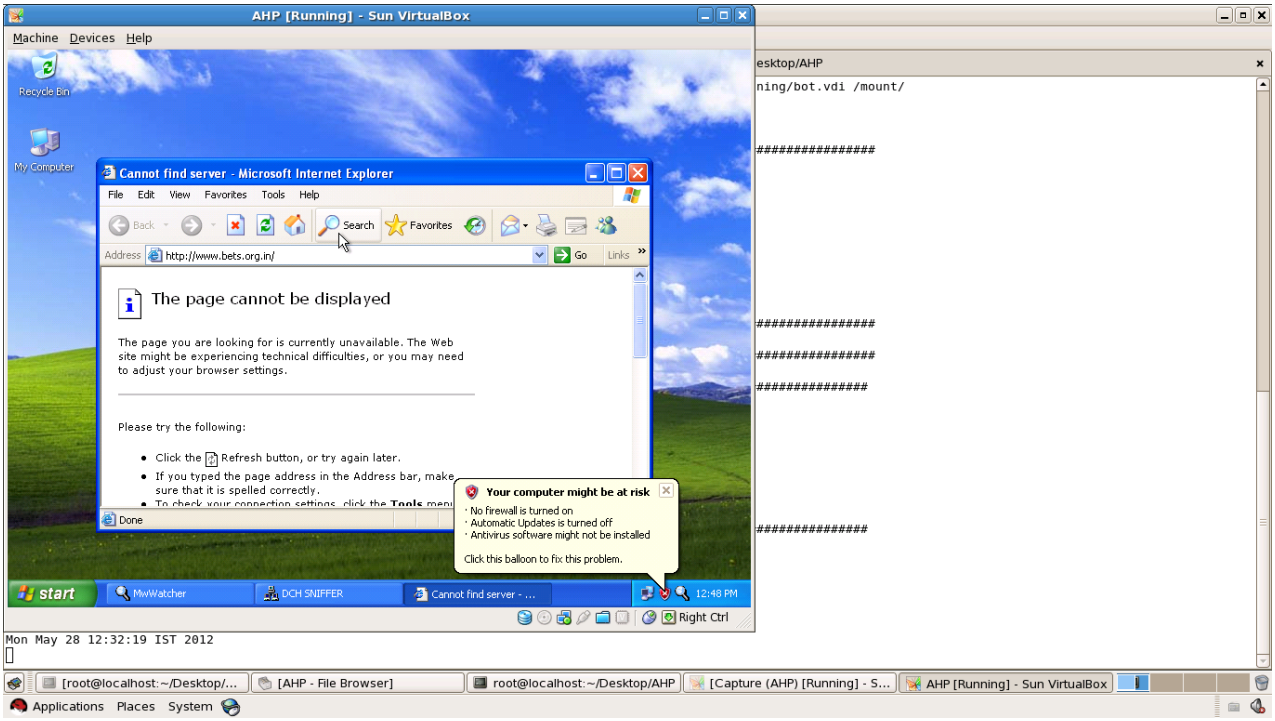

Figure 4.14 Opening sites automatically when window starts

Step 10 In this step execution of the URL will be started for 90 sec duration.

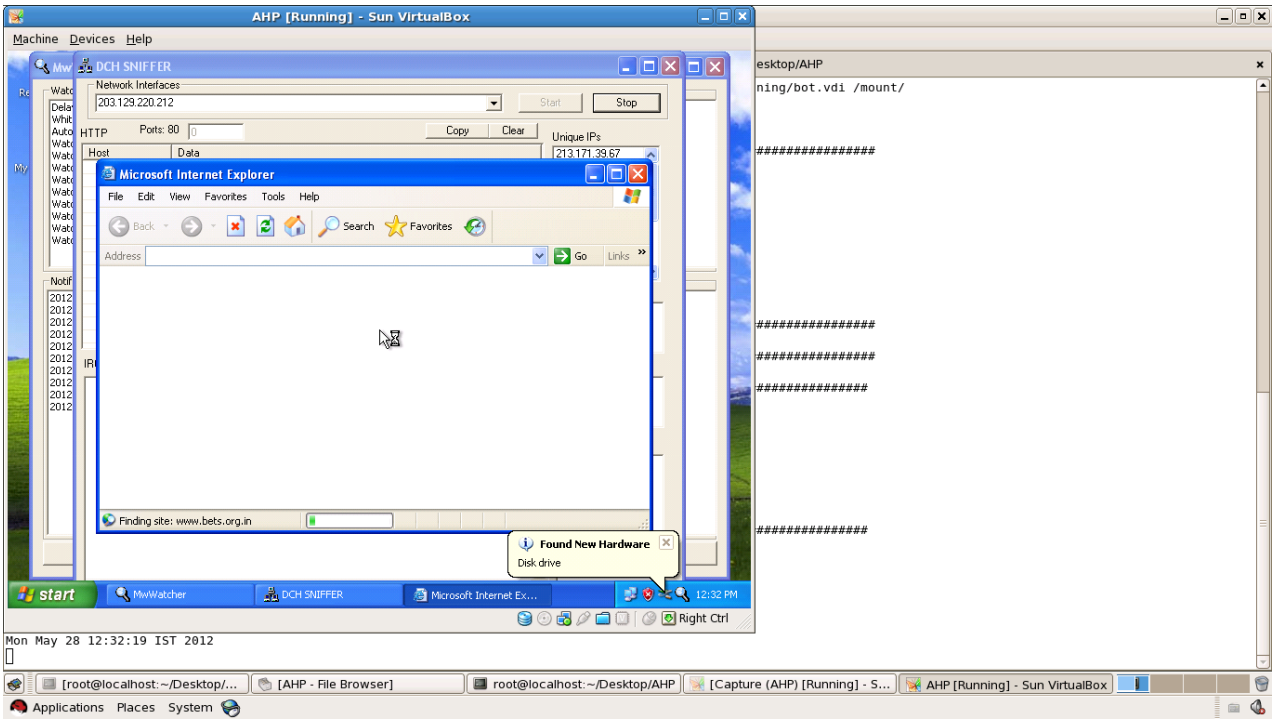

Figure 4.15 Execution of URL's for 90 sec

Step 11 In this step Mwatcher detect and collect internet malwares from the URLs given in the database.

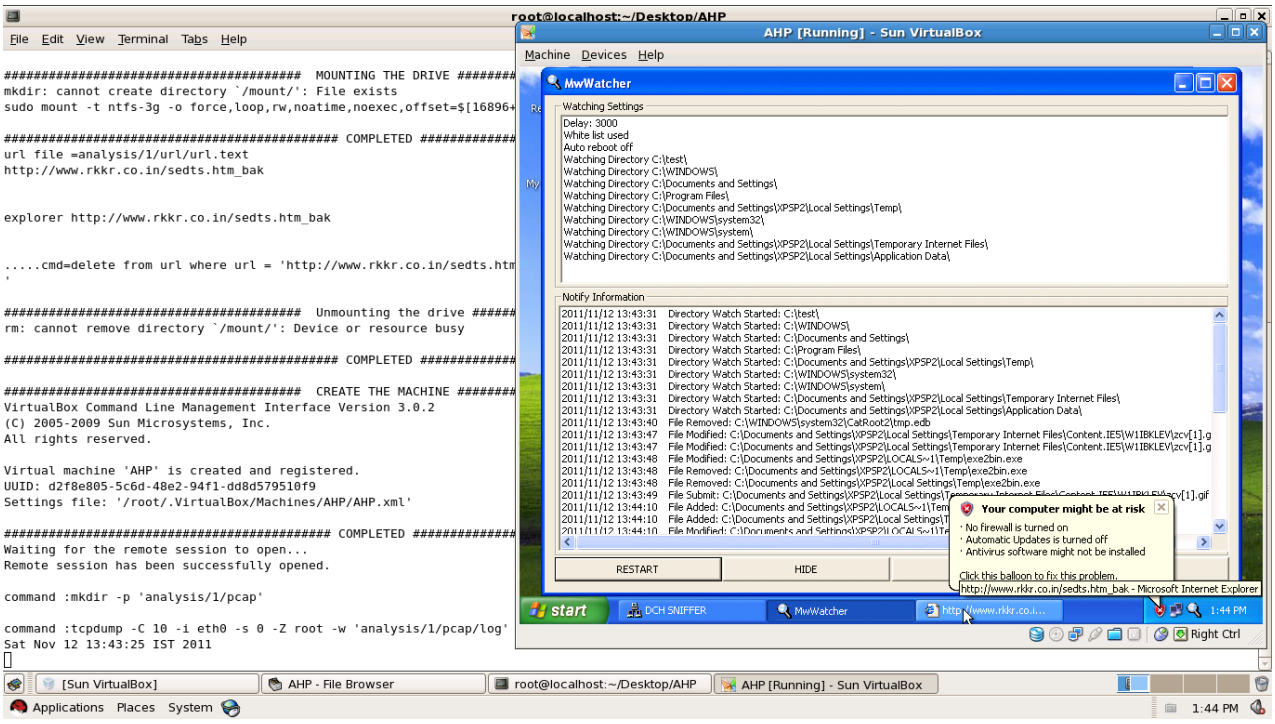

Figure 4.16 MwWatcher working for file system monitoring

Step 12 In this step we showed that DCHsniffer captures the packet.

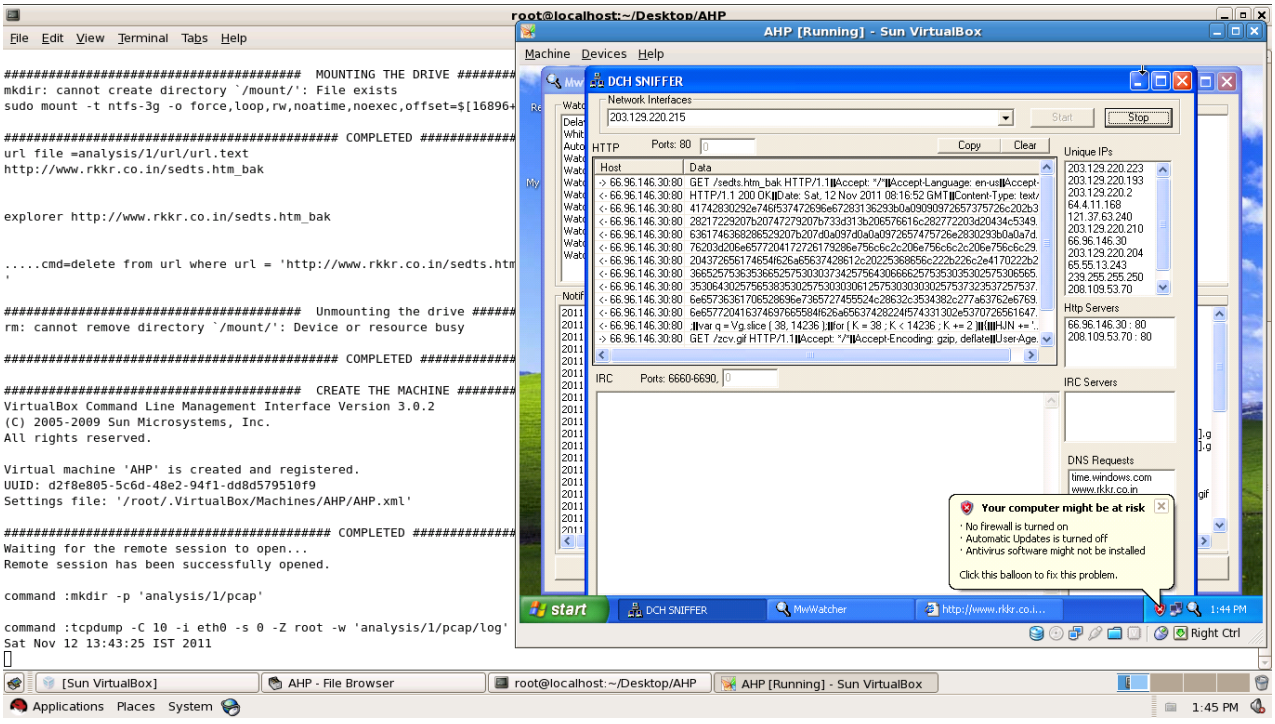

Figure 4.17 DCHsniffer capturing packets

Step 12 In this step we show the message that all urls given in the file are opened for 90 seconds,provide new urls to make the start.

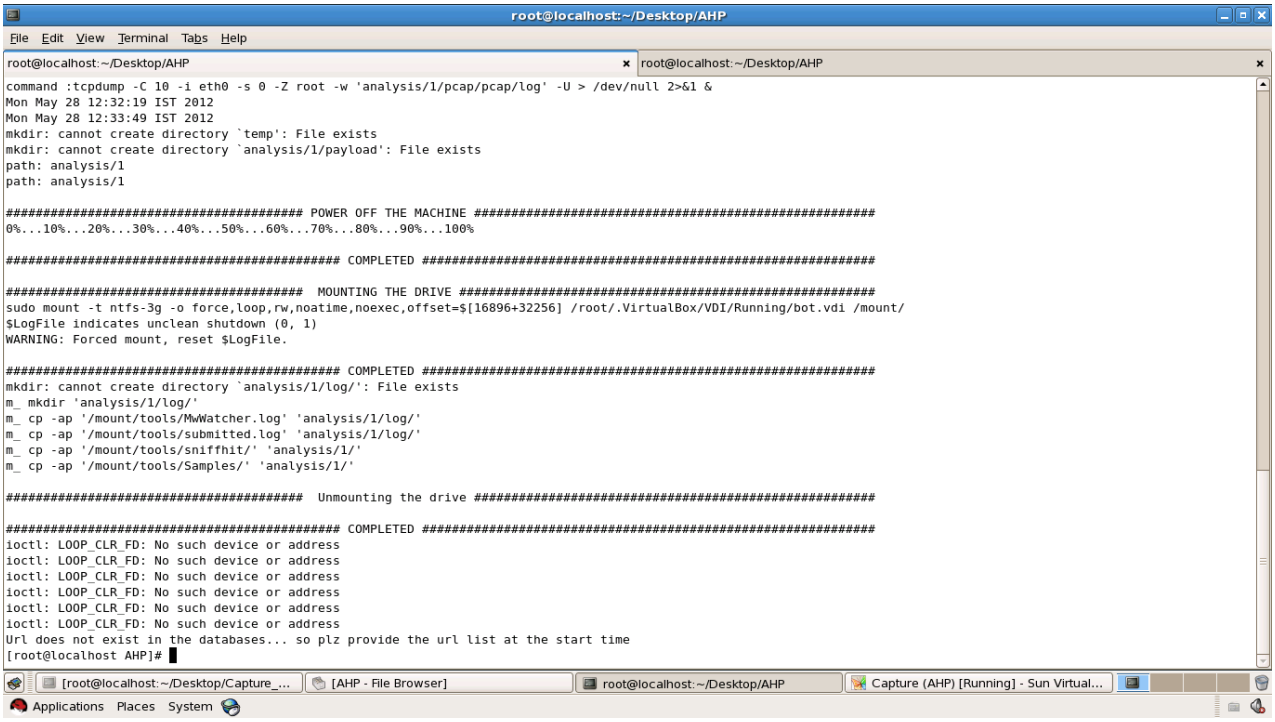

Figure 4.18 Showing message when all URL's opened

Step 13 In this step all the logs like PCAP data, malware collected etc will be in analysis folder.

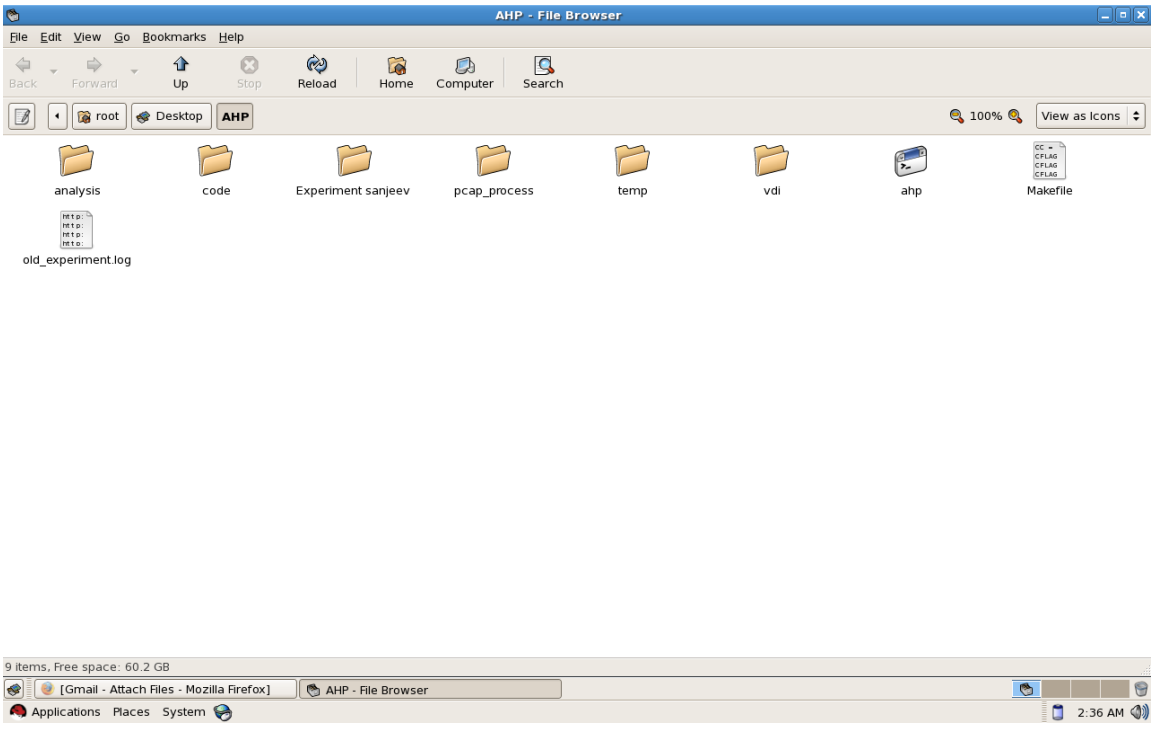

Figure 4.19 Opening Analysis Folder to check log

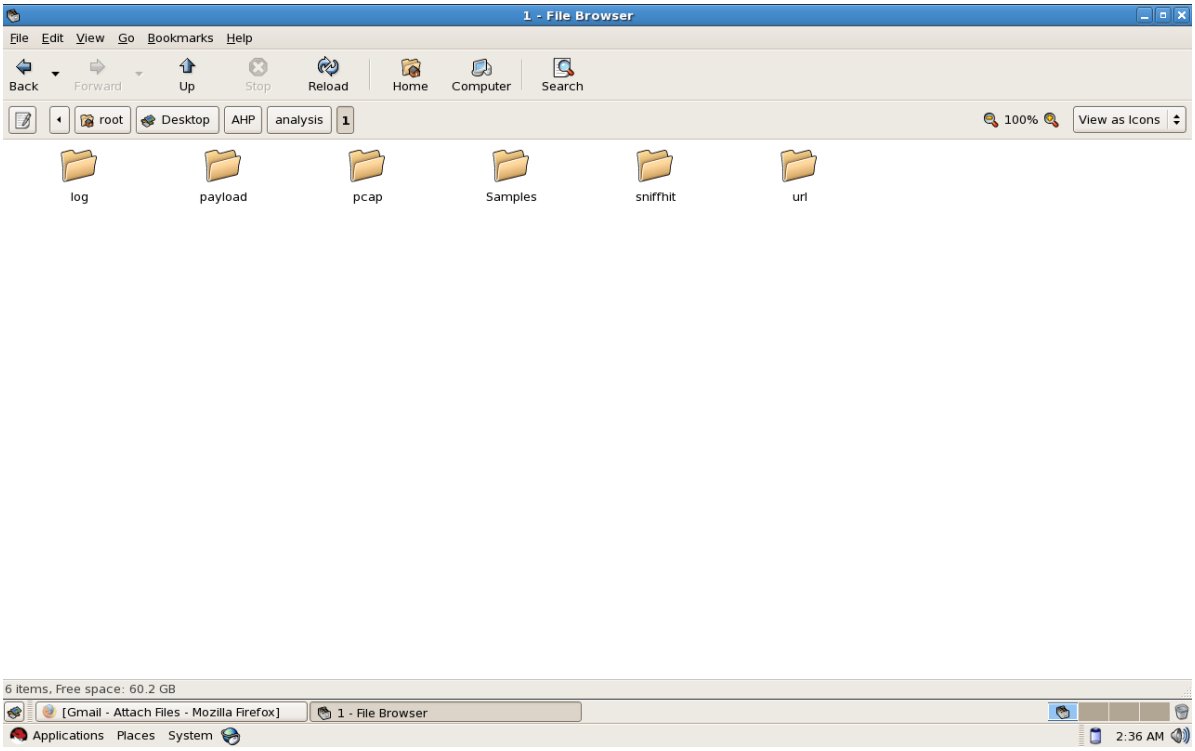

Figure 4.20 Analysis-→1

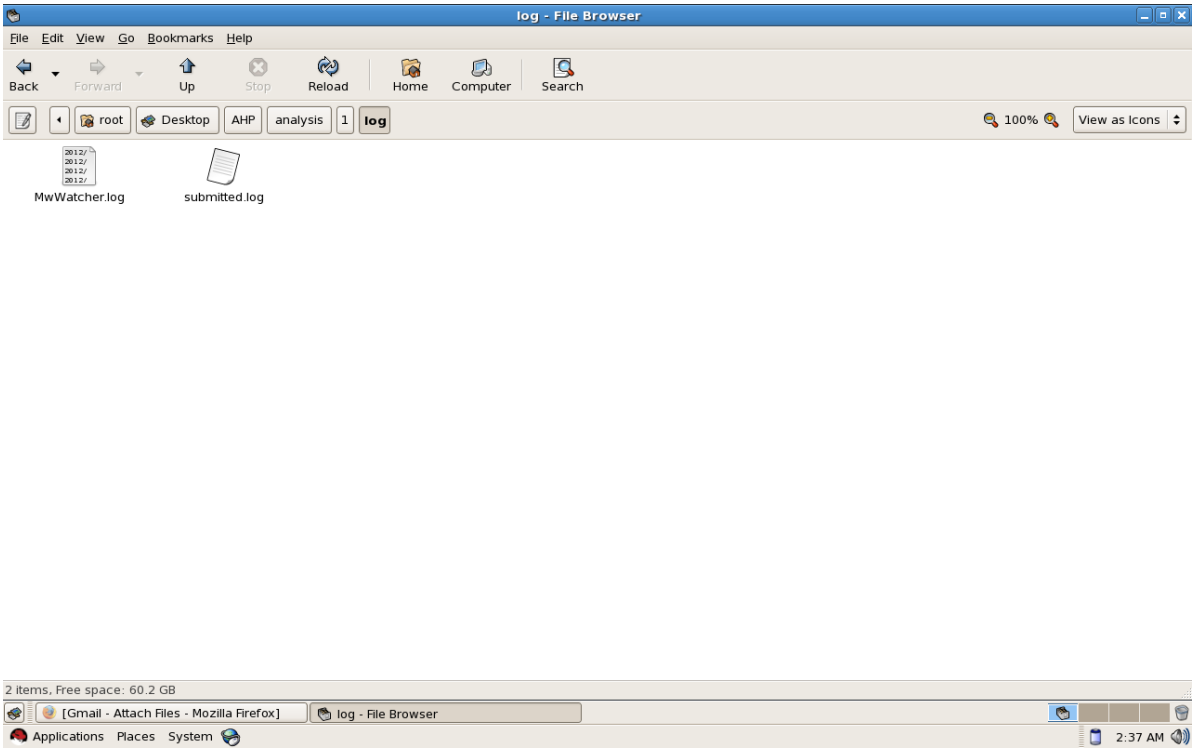

Figure 4.21 Analysis->1->log

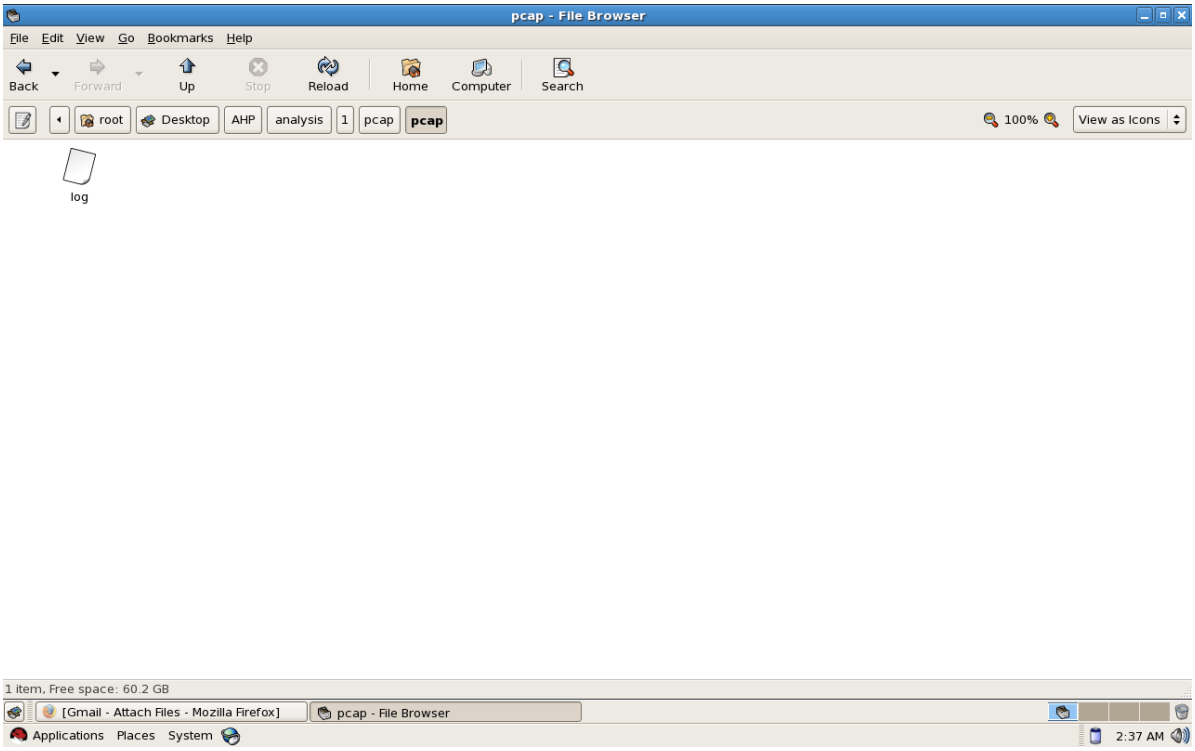

Figure 4.22 Packet captured log

4.5 Experimental Results of Honeyclient

url	**Stem**	**hostname**	**md5**
http://admarcontabil.sites.uol.com.br///live.txt	///live.txt	admarcontabil.sites.uol.com.br	cc4c77ee54de37e9089c7aae2e24d9a2
http://ew.correa.sites.uol.com.br///RITINHA.jpg	///RITINHA.jpg	ew.correa.sites.uol.com.br	5912d4f1845de44a4e5c9e9db891c65f
http://loys.com.br///oportunidade/images/01.jpg	///oportunidade/images/01.jpg	loys.com.br	3f7d7f857f13174261540d6db7c48e2d
http://cairujp.sites.uol.com.br///mynoticia/MSDOS.jjs	///mynoticia/MSDOS.jjs	cairujp.sites.uol.com.br	5148c280f7c3deee03993df5d34f748b
http://rudsonfr.sites.uol.com.br///album/album1/images/t/3.jjs	///album/album1/images/t/3.jjs	rudsonfr.sites.uol.com.br	de33c32e49679dcbf7535f9d8f6438e5
http://tomiya.sites.uol.com.br///mixirica.css	///mixirica.css	tomiya.sites.uol.com.br	461632d20fe61f26eace07401806f435
http://depaulamdp.sites.uol.com.br///aut.jpg	///aut.jpg	depaulamdp.sites.uol.com.br	337877a8689824558ba8c17a03763776

http://depaulamdp.sites.uol.com.br///ger.jpg	///ger.jpg	depaulamdp.sites.uol.com.br	3d703dc011fbde773e85372706aa74b1
http://pixwall.net///summer/XvidSetup.exe	///summer/XvidSetup.exe	pixwall.net	ae8621d33a5d184534bab844a0716d1b
http://magpietech.in///FacebookMessengerSetup.exe	///FacebookMessengerSetup.exe	magpietech.in	597284a2c42a074c304a0ffafe99620d
http://strandednaked.com///media/XvidSetup.exe	///media/XvidSetup.exe	strandednaked.com	ae8621d33a5d184534bab844a0716d1b
http://gucosilva.sites.uol.com.br///downloada.jpg	///downloada.jpg	gucosilva.sites.uol.com.br	5d1cdf7ff4c57503c2352f1d6bf3a149

Table 5.1 Experimental results of virtual client honeypot

Chapter 5

CONCLUSION AND FUTURE SCOPE

This chapter discusses the conclusions of work presented in this thesis. This chapter ends with a discussion of the future direction which can be taken further.

5.1 Conclusion

Computer networks have brought the world together by bridging the information gap among people. Network technology has undergone a revolution with better and faster ways of sending information between computers. Unfortunately security systems and policies to govern these networks have not progressed as the same speed. Today's network is very complex and the whole world is focusing on ease of use and functionality. This is diversity to our concern for the security towards the ease of use and increase of functionality. For hackers, these well-travelled paths make networks more vulnerable than ever before and with relatively little expertise hackers have significantly impacted the networks of leading brands or government agencies. Cyber crime is also no longer the prerogative of lone hackers or random attackers. Today disgruntled employees, unethical corporations, even terrorist organizations all look to the Internet as a portal to gather sensitive data and instigate economic and political disruption. With networks more vulnerable and hackers equipped to cause havoc, it's no surprise that network attacks are on the rise. So there is a huge need of detecting and preventing the threats and intrusion. In this work, we presented the Internet malware system using client-side honeypot. We use the active ability of client-side honeypot to collect malware that traditional honeypot cannot get in the Internet. We introduced the category of Internet malware, the client side attack techniques and overall framework of the system in detail. We mainly gave the design and implementation of client honeypots based malware collection. During the work done so far, client honeypot based solution is very useful to collect the internet malwares and to detect the malicious websites.

5.2 Thesis Contribution

a) In this thesis Honeypots introduction,architecture of client honeypot explained in detailed and Research and requirement of Client Honeypot as compare to traditional server honeypots.

b) Implementation design setup, Functional diagram, MwWatcher tool for file system monitoring is also explained.

c) The designed has been implemented and deployed on linux based client Honeypot to collect internet malwares.

5.3 Future Scope

Our developed Virtual Box powered Honeyclient is very useful for collection of internet malwares but it is having a limited capabilities or we can say that it is just a prototype. There is a requirement of integration of crawler as data acquirement, at present there is no such component in our developed module. Further there is also a possibility of addition of various client side applications such as firefox, pdf etc because currently we only using Internet Explorer for actively visiting the websites. And there is also a possibility of addition of automatically analysis of collected malwares. We can confirm that we cannot cover all the challenges such human user simulation, logic bomb, time triggered websites but we have developed a prototype solution to get better understanding of client honeypots**.**

LIST OF PUBLICATIONS

[1] Himani Gupta, Gurpal Singh, "Design And Implementation Of Linux Based Virtual Honeyclient", in IJARCET 2012 (International Journal Of Advanced Research In Computer Engineering And Technology), ISSN Bhopal section (JUNE 2012) (Status-Submitted).

REFERENCES

[1] Przemyslaw Kazienko, Piotr Dorosz "*Intrusion Detection Systems (IDS) Classification*" methods; techniques", June 2004.

[2] Krugel Christopher, Toth Thomas: "*A Survey on Intrusion Detection Systems*", TU Vienna, Austria 2000 7, 22-33.

[3] Juergen Heit, "*Intruders & Passwords*", Cryptographic protocols seminar, 20 June 2005.

[4] The Honeynet Project, " *Know Your Enemy : Honeynets*" , May 2005. http://www.honeynet .org/papers/honeynet/.

[5] Ram Kumar Singh, Prof. T. Ramanujam "*Intrusion Detection System Using Advanced Honeypots",* International Journal of Computer Science and Information Security, Vol. 2, No. 1, 2009

[6] Fordahl, Matthew "Honeynet Project: Security gurus study hackers" ,29 July, 2001 http://www.newstribune.com/stories/072901/wor_0729010050.asp

[7] Edward Amoroso. *Intrusion Detection: An Introduction to Internet Surveillance, Correlation, Trace Back, Traps and Responses*. Intrusion NetBooks, 1999.

[8] Theuns Verwoerd , "*Active Network Security*", Honours Report, 5 November,1999.

[9] Pikoulas J, Mannion M, Buchanan W, "*Software Agents and Computer Network Security*", IEEE Computer Magazine, pages 16-l 7, December 1997.

[10] Robert E. Mahan, "*Introduction to Computer & Network Security",* Version 2.1, CCIMB99-031, August 1999, pg. 14.

[11] Mario Guimaraes, Meg Murray "*Overview of Intrusion Detection and Intrusion Prevention"*, Information security curriculum development Conference by ACM (2008).

[12] Ahmed Patel, Qais Qassim, Christopher Wills " *A survey of intrusion detection and prevention systems"*, Information Management & Computer Security Journal (2010).

[13] Wayne Jansen, Peter Mell, Tom Karygiannis, Don Marks, "*Applying Mobile Agents to Intrusion Detection and Response",* National Institute of Standards and Technology Computer Security Division NIST Interim Report (IR) – 6416 October 1999.

[14] G. Cabri, L. Leonardi, F. Zambonelli, "*Mobile Agent Technology: Current Trends And Perspectives",* ACM Trans.on Computer Systems, vol. 2, pp. 39-59, February 1984.

[15] Shiv Shakti Srivastava, Nitin Gupta, Saurabh Chaturvedi, Saugata Ghosh, " *A Survey on Mobile Agent based Intrusion Detection System",* Advance Computing Conference, 2009.

IACC 2009. IEEE International, DOI: 10.1109/IADCC.2009.4809135

[16] Spitzner, L. The Value of Honeypots, Part One: "*Definitions and Values of Honeypots*", Security Focus, 2001.

[17] Feng Zhang, Shijie Zhou. Zhiguang Qin, Jinde Liu, "*Honeypot: a Supplemented Active Defense System for Network Security*", QOP 2005, 1st Workshop on Quality of Protection (co-located with ESORICS and METRICS), Sept. 15, Milan, Italy, 2005.

[18] Karthik Sadasivam, Banuprasad Samudrala, T. Andrew Yang, "*Design Of Network Security Projects Using Honeypots"* Proceedings of the Second IEEE International Information Assurance Workshop (IWIA'04), 2004.

[19] Iyatiti Mokube, Michele Adams, "*Honeypots: Concepts, Approaches, and Challenges",* Swiss Federal Institute of Technology, Zurich, 2002.

[20] Mahmoud T. Qassrawi*, "Client Honeypots: Approaches and Challenges",*Proceedings of the New trends in information Science and Service Science(NISS), 2010 4th International Conference on May 2010,19-25.

[21] http://www.oldenbourg-verlag.de/wissenschaftsverlag/client-honeypots/9783486705263

[22] Offensive-Security, Client Side Attacks, 2009 URL:http://www.offensive security.com/metasploitunleashed/ Client-Side-Attacks.

[23] MSDN Library, Understanding User-Agent Strings, [Online]. Available at: http://msdn.microsoft.com/en-us/library/ms537503(VS.85).aspx .

[24] S. Lim, "*Assessing the effect of honeypots on cyber-attackers*," M.S. thesis, Naval Postgraduate School, 2006.

[25] The New Zealand Honeynet Project, Capture-HPC -Capture - The High-interaction Honeyclient ,URL:http://www.nzhoneynet.org/capture.html.

[26] K. Wang. Honeyclient Development Project.URL:http://www.honeyclient.org/.

[27] Krisztion Piller,Sebastian Wolfgarten.Honeymonkeys- Chasing hackers with a bunch of monkeys.2005-12-30.http: //events.ccc.de/congress/2005/fahrplan/attachments/686-slides_honeymonkeys.pdf.

[28] Mpack web-based exploit tool [Online]. Available at: http://blogs.pandasoftware.com/blogs/images/PandaLabs/2007/05/11/MPack.pdf .

[29] C. Seifert, I. Welch, P. Komisarczuk, (2006). HoneyC - The Low-Interaction Client Honeypot [Online]. Available at: http://homepages.ecs.vuw.ac.nz/~cseifert.

[30] Monkey-Spider [Online]. Available at: http://pi1.informatik.unimannheim.de/filepool/publications/monkey-spider.pdf .

[31] http://www.honeynet.org/project/PhoneyC

[32] SpyBye [Online]. Available at: http://www.spybye.org.

[33] http://www.honeyspider.net/wp-content/uploads/2009/06/hsn-first2008-article-v02.pdf

[34] Strider HoneyMonkey Exploit Detection, http://research.microsoft.com/HoneyMonkey.

[35] C. Seifert, Improving Detection Speed and Accuracy with Hybrid Client Honeypots, Victoria University of Wellington, PhD thesis, 2008.

[36] Ram kumar Singh,Prof. T. Ramanujan, " *Intrusion Detection System Using Advanced Honeypots"*, Proceeding of International Journal of Computer Science and Information Security (IJCSIS), June 2009 Issue, Vol. 2, No. 1.

[37] Y.Alosefer and O.F Rana, "*Honeyware: a web based low interaction client honeypot*", Proceeding of Software testing,verification and validation workshops, third conference on april 2010,410-417.

[38] Xiaoyan sun,Yang Wang,Jie Ren, "*Collecting internet malwares based on client side honeypot*", Proceeding of Young Computer Scientist 9th international conference on Nov 2008.

[39] Mahmoud T.Qassrawi,HOngli Zhang, "*Using Honeyclients to detect Malicious Websites*", Proceedings of e-Business and information System security 2nd National conference on May 2010,1-6.

[40] Saurabh Chamotra,Rakesh Kumar,Raj kamal, "*Data diversity of a distributed honeynet based malware collection system*", Proceedings of Emerging Trends in Networks and computers communications International conference on April 2011.

[41] A. Ikinci. Monkey-spider: Detecting malicious web sites. Master's thesis, University of Mannheim, 2007.

[42] N. Provos and T. Holz, Virtual Honeypots: From Botnet Tracking to Intrusion Detection, Addison-Wesley, 2008, pp. 231–272.