Sudan University of Science and Technology

College of Graduate Studies

College of Computer Science & Information Technology

Department of Information Technology

Collection and Analysis of Attackers Data using Honeynet

"جمع وتحليل بيانات المهاجمين باستخدام الهوني نت"

A Thesis Submitted in Partial Fulfillment of the requirements for the degree of M.Sc.in Information Technology

By:
Solafa Salahaldin Ali Shaikhedris

Supervisor:
Dr. Faisal Mohammed Abdalla Ali

December 2018

Abstract

Honey pots and honey nets are secure unconventional tools to study techniques, methods, tools, and goals of attackers. Therefore, data analysis is an important part of honey pots and honey nets. Honey pots are devices deployed specifically to be a resource for the attack or compromising. Honeynets are deployed to collect information, namely the tools and tactics and motivations of the hacker’s community, and then this information is used to protect the organizations from different threats. This research aims to study the most frequent, new and automated attacks, moreover the behavior of the malicious attackers is analyzed as soon as they got manage to access a new host. Virtual honey net is implemented in order to capture the whole activities of the attackers. The collected data is analyzing using Wireshark in order to get massive information about hackers. In addition to this, the research explains a secure method to transfer collected data from honey pots to be Analyzed. Results were analyzed attacks targeting our honeynet over a period of 60 days, which made it obviously for us to know: Attacked/Probed ports and services, Attacker IP's, OS used in every packet that were captured, Packet length, format, and time.

المستخلص

مصائد مخترقي الشبكات هوني بوت وهوني نت هي أدوات أمنية غير تقليدية لدراسة تقنيات وأساليب وأدوات وأهداف المهاجمين، لذلك فإن تحليل البيانات جزء مهم منها. الهوني بوت عبارة عن أجهزة صممت لتكون مورداً للهجوم او الاختراق كما ان الهوني نت هي عبارة عن مصيدة مصممة لجمع المعلومات وتحديدًا أدوات وتكتيكات ودوافع مجتمع المخترقين، ومن ثم استخدام هذه المعلومات لحماية المنظمات من التهديدات المختلفة. يهدف هذا البحث إلى دراسة الهجمات الأكثر تكرارًا والجديدة والأوتوماتيكية. وعلاوة على ذلك، يُحلَّل سلوك المهاجمين الخبيثين بمجرد تمكنهم من الوصول إلى مضيف جديد، حيث يطبّق نظام الهوني نت الافتراضي لجمع البيانات و الأنشطة التي يقوم بها المهاجمين ومن ثم تحليل هذه البيانات باستخدام برنامج الويرشارك (Wireshark) للحصول على كم هائل من المعلومات عن المهاجمين إضافة إلى ذلك، يوضح هذا البحث طريقة آمنة لنقل البيانات المجمعة من مصائد مخترقي الشبكات إلى التحليل نفسه كما تم التوصل من خلال هذا البحث الى النتائج الآتية: معرفة كل من المنافذ (ports) والخدمات المستخدمة بواسطة المهاجمين، عناوين الانترنت المرسلة والمستقبلة للهجمات، وأنظمة التشغيل المستخدمة وحجم وزمن وصيغة الحزمة المرسلة.

Acknowledgments

This master thesis would not be successful without the help and support of a great number of people. First and foremost I would like to thank my supervisor Faisal Mohammed to have it made possible for me to work on my master thesis and his understanding and availability as well as his thoughtful guidance when things were turning complicated, and Without his efforts and patience this work would not be possible. Then**,** my gratitude also goes to all teachers of my master program who efficiently prepared me for this master’s thesis, in particular Doctor Mohssen Mohammed Zain for sharing his insights and knowledge about security.

I want to thank security team, especially Musab M. Mairgany and Doctor Mohssen Mohammed Zain for their support to honey pots problems.

I also want to thank deeply and greatly my aunt Thoria who motivated me to study the master program. Finally, I would like to thank my family for supporting me at Home all the time.

Contents:

CHAPTER I: Introduction

CHAPTER II: Literature Review

CHAPTER III: Methodology

CHAPTER IV: Implementation

CHAPTER V: Conclusion and Future work

List of tables:

List of figures:

List of appendixes:

Chapter I
Introduction

Chapter I Introduction

1-1 Introduction

Network Security becomes the keystone in modern societies because of new threats that arise every day. These threats demand advanced security solutions that are no more available by traditional tools. Traditional solutions, such as Firewalls and IDS (Intrusion Detection System), are unable to keep up with the evolution of the attackers and their techniques [1, 2], as they suffer from detecting and stopping unknown attacks and new attackers' methods. To achieve a good level of security, the security tools should not only be interested in the defense mechanisms, but also must rely on deceptive attackers and must have the initiative to disclose the attack when it occurs [2]. Security threats range from hacking intrusions, denial of service attacks to computer worms, viruses and more. We must understand that intrusion to a network or system can never be eliminated but it can be reduced. But the attackers aware of some vulnerabilities, so it is crucial to identify somehow their activities.

A honey pot is a system that is built and set up in order to be hacked, therefore it can be deployed to consume the attackers' resources and to waste their time on honey pots instead of attacking production systems. These purposes help to detect new attacks and learn more about attackers' techniques [2]. The key role of the Honey pot can be performed by any resource that can be used for observing hostile or unexpected activity. From the viewpoint that a honey pot does not advertise any resources for regular use, any interaction with the honey pot is assumed to be an intrusion and worthy of further investigation [3]. There are many advantages of Honey pots because of their simple concept that gives them powerful strengths However a Honey pot does not replace existing security technologies but can work alongside them tracking and capturing activity occurring on the system where it is deployed However it will only capture activity that is directed at the Honey pot itself The Honey pot can also be at risk of sabotage and could be used to attack other connected systems [4]. The only common feature of this resource is that it is not used for production purposes. The Honey pot is mostly specialized machine or software. One of the main problems of Internet security is the number of new, so far unknown threats due to existing security holes in the software – called "zero day threats" [5]. We also face the problem of automatically and reliably detecting previously unknown attacks which are known as zero-day attack [6].

Invaluable data upon security threats in the network can be extracted from the Honey pot systems.

From this perspective, we need to find new approaches to protect information and infrastructure of the organizations. One of the effective approaches to protect them is the concept of honey nets.

In this research we will Carry out various analyses based on the collected data to better understand threats to characterize attack processes also Analyze the behavior of malicious attackers once they manage to get access and compromise a target High-interaction honeypots[7] (honeynet).

1-2 Problem statements

The main challenge in network security is keeping up with all threat types that arise every day. Traditional security mechanisms such as firewalls and Intrusion Detection System (IDS) do not provide detection for new attacks or helping in learning new attackers' techniques, which is result in worse situation that reveals to us these serious questions:

Who is trying to compromise our system?

How can we discover the unknown attacks and how can we avoid them in the future?.

1-3 Objectives of research

The main purpose of undertaking this work is to shed light on the following:

- Knowing trends in the attacks domain and learn how to protect one.
- Knowing one's enemies.
- Catch next tools (worm…).
- Make the environment more secure by Detection of new attacks.
- Get prepared in case of attacks on operational networks.

Honeypots can also be used to catch hackers while they are in the network and to redirect hackers from the actual production systems to the honey pot system [9].

1-4 Methodology

Build honeynet system architecture in VMware (virtual machine) environment to perform the data collection using sebek software, snort software for detection, and alert the administrator, then analyzes the information collected by a honeynet system using weirshark program also Nmap software used for port scanning.

1-5 Research content

The remainder of this research is organized as follows: Chapter II presents an overview of literature review and related works (honey pot technologies and theory). Chapter III discusses in detail the (methodology) system's architecture and the software, which have been used. The implementation is presented in Chapter IV we build a virtual honey net. It is a network of virtual computers, whose operating systems are configured to become honey pots. All these virtual computers are running on one physical machine. At the end, chapter V concludes the research and highlights the future work.

Chapter II

Literature Review

Chapter II Literature Review

2-1 Introduction

The revolution in Information Technology has provided a flood of assets in the form of applications and services. Enterprises have based their entire business models on top of these assets. Networks have evolved from low speed half duplex links to full duplex, multi-homed, self-convergent, gigabyte streams, controlled by advanced protocols. The security of the available applications and services accessible over these networks currently represents a major challenge to the IT industry. Each day, exploits, worms, viruses and buffer overflows severely threaten the IT infrastructure and associated business assets along with mission critical systems. By learning the tactics and techniques used by malicious *black hats* crackers, we can secure our IT assets and infrastructure. Honey pots provide a means to study black-hat techniques and tactics through which they have been able to gain illegitimate access to system resources along with methods for analyzing the tools that they use to obtain this access [9].

2-2 Network security

Network security comprises a wide range of concepts, provisions, and policies adopted by a network administrator or some management authority to prevent and monitor unauthorized access, misuse, modification, or denial of a computer network and network-accessible resources [10].

Computer networks allow communicating faster than any other facilities. These networks allow the user to access local and remote databases. It is impossible to protect every system on the network. In industries, the network and its security are important issues, as a breach in the system can cause major problems. The security of network is required for improvement of the industries which are dependent on the internet to enhance the business and providing services on the network so security of network is primary concern of the industries for securing the critical information [11].

There is a significant lack of security methods that can be easily implemented. There exists a “communication gap” between the developers of security technology and developers of networks. Although there isn't a methodology to manage the complexity of security requirements. When considering about network security, it should be emphasized that the complete network is secure. It does not only concern with the security in the computers at each end of the communication chain [12].

Network security is the process by which digital information assets are protected. The main goals of security are to protect confidentiality, maintain integrity, and ensure availability. With this in mind, it is imperative that all networks must be protected as much as possible from threats and vulnerabilities for a business to achieve its fullest potential.

A threat refers to anything that has the potential to cause serious harm to a computer system. It is something that may or may not happen but has the potential to cause serious damage. Threats can lead to attacks on computer systems, networks, and more. Typically, these threats are persistent due to vulnerabilities, which can arise from misconfigured hardware or software, poor network design, inherent technology weaknesses, or end-user carelessness [10].

2-3 Intrusion Detection System

Intrusion detection is a set of techniques and methods that are used to detect suspicious activity both at the network and host level. Intrusion Detection System or IDS is software, hardware or combination of both used to detect intruder activity. Snort is an open source IDS available to the public. Intrusion detection systems fall into two basic categories:

- Signature-based intrusion detection systems.
- Anomaly detection systems.

2-3-1 Signature-Based intrusion Detection System:

Intruders have signatures, like computer viruses, that can be detected using software. You try to find data packets that contain any known intrusion-related signatures or anomalies related to Internet protocols. Based upon a set of signatures and rules, the detection system is able to find and log suspicious activity and generate alerts.

2-3-2 Anomaly Detection System:

Anomaly-based intrusion detection usually depends on packet anomalies present in protocol header parts. In some cases, these methods produce better results compared to signature-based IDS [13].

2-4 Types of Threats

In general, there are two types of attackers: the kind which wants to compromise as many systems as possible and the kind which wants to compromise a specific system or systems of high value [8]. Most threats tend to fall into one of these two categories.

2-4-1 Script kiddies: These types of attackers usually depend on scripted attacks. Sometimes, these attackers have certain requirements, such as hacking systems with a fast connection to the Internet or a large hard drive for storing files. In general, however, all they care about are numbers.

2-4-2 Advanced black hat: These types of attackers focus on targets of choice and may want to compromise a specific system or systems of high value. These individuals are most likely highly experienced and knowledgeable attackers.

Not only can they penetrate highly secured systems, their actions are difficult to detect and trace. Advanced black hats make little “noise” when attacking systems and they excel at covering their tracks. Even if you have been successfully attacked by such a skilled black hat, you may never even be aware of it [8, 10].

2-5 Types of Attacks

Here we are presenting some basic class of attacks which can be a cause for slow network performance, uncontrolled traffic, viruses etc. Attacks to network from malicious nodes. Attacks can be categorized in two:

"Passive" when a network intruder intercepts data traveling through the network, and "Active" in which an intruder initiates commands to disrupt the network's normal operation.

2-5-1 Active attack

Some active attacks are spoofing attack, Wormhole attack, Modification, Denial of services, Sinkhole, and Sybil attack [12].

- Spoofing

When a malicious node miss-present his identity, so that the sender change the topology

- Modification

When malicious node performs some modification in the routing route, so that sender sends the message through the long route. This attack cause communication delay occurred between sender and receiver.

- Wormhole

This attack is also called the tunneling attack. In this attack an attacker receives a packet at one point and tunnels it to another malicious node in the network. So that a beginner assumes that he found the shortest path in the network.

- Denial of services

In denial of services attack, malicious node sending the message to the node and consume the bandwidth of the network. The main aim of the malicious node is to be busy the network node. If a message from unauthenticated node will come, then receiver will not receive that message because he is busy and beginner has to wait for the receiver response.

- Sinkhole

Sinkhole is a service attack that prevents the base station from obtaining complete and correct information. In this attack, a node tries to attract the data to it from his all neighboring node. Selective modification, forwarding or dropping of data can be done by using this attack.

- Sybil

This attack related to the multiple copies of malicious nodes. The Sybil attack can be happen due to malicious node shares its secret key with other malicious nodes. In this way the number of malicious node is increased in the network and the probability of the attack is also increases. If we used the multipath routing, then the possibility of selecting a path malicious node will be increased in the network.

2-5-2 Passive attack

The names of some passive attacks are traffic analysis, Eavesdropping, and Monitoring [12].

- Traffic analysis

In the traffic analysis attack, an attacker tries to sense the communication path between the sender and receiver. An attacker can found the amount of data which is travel from the route of sender and receiver. There is no modification in data by the traffic analysis.

- Eavesdropping

This is a passive attack, which occurred in the mobile ad-hoc network. The main aim of this attack is to find out some secret or confidential information from communication. This secrete information may be privet or public key of sender or receiver or any secrete data.

- Monitoring

In this attack in which attacker can read the confidential data, but he cannot edit the data or cannot modify the data.

2-6 Zero-Day Attack

Zero-day vulnerabilities are software vulnerabilities for which no patch or fix has been publicly released. The term zero-day refers to the number of days a software vendor has known about the vulnerability. Attackers use zero day vulnerabilities to go after organizations and targets that diligently stay current on patches; those that are not diligent can be attacked via vulnerabilities for which patches exist but have not been applied. [14]
Wikipedia defines "zero-day virus" as "a previously unknown computer virus or other malware for which specific anti-virus software signatures are not yet available". According to *Wikipedia*, "a zero-day attack or threat is a computer threat that tries to exploit computer application vulnerabilities that are unknown to others or undisclosed to the software developer". There is also a notion of a *vulnerability window* which is the time between the first exploitation of vulnerability and when software developers start to develop a countermeasure to that threat. These definitions evaluate time points such as the attack release and the moment when the very first easing is available [13].

2-7 Security services

X.800 defines a security service as a service provided by a protocol layer of communicating open systems, which ensures adequate security of the systems or of data transfers. In addition, the RFC 2828 defines security services as a processing or communication service that is provided by a system to give a specific kind of protection to system resources. Security Services implement security policies and are implemented by security mechanisms. X.800 divides these services into five categories:

2-7-1 Authentication: The assurance that the communicating entity is the one that it claims to be.

- Peer Entity Authentication: Used in association with a logical connection to provide confidence in the identity of the entities connected.

- Data Origin Authentication: In a connectionless transfer, provides assurance that the source of received data is as claimed.

2-7-2 **Access control**: The prevention of unauthorized use of a resource.

2-7-3 **Data confidentiality**: The protection of data from unauthorized disclosure.

2-7-4 **Data integrity**: The assurance that data received are exactly as sent by an authorized entity.

2-7-5 **Nonrepudiation**: Provides protection against denial by one of the entities involved in a communication of having participated in all or part of the communication [15].

2-8 Protection tools

2-8-1 Firewall

A firewall is a technology that protects your organization by controlling what traffic can flow where. These are used as an access control tool. Firewalls are most commonly deployed around an organization's perimeter to block unauthorized activity.

2-8-2 IDS

An intrusion detection system (IDS) is a device or software application that monitors network or system activities for malicious activities or policy violations and produces reports to the administrators.

2-8-3 IPS

An intrusion prevention system (IPS) consists of network security appliances that monitor network or system activities for malicious activity, attempting to block or stop the activity, and report it.

2-8-4 IDPS

A combination of the IDS and IPS is called an intrusion detection and prevention system (IDPS). Most organizations now use IDPS products because they offer great defense mechanisms. The main functions of the IDPSs are focused on identifying possible incidents, logging information about them, attempting to stop them, and report them to security administrators [10].

2-8-5 IP protection

The standard Internet communication protocol is completely unprotected, allowing hosts to inspect or modify data which is in transit. Adding IPSec to systems will overcome this limitation by providing strong encryption, integrity, authentication and replay protection.

Internet Protocol Security (**IP sec**) is a collection of protocols designed by the Internet Engineering Task Force to provide security for a packet at the network layer. IPSec is an end-to-end security scheme operating in the Internet Layer of the Internet Protocol Suite. It can be used in protecting data flows between a pair of hosts (*host-to-host*), between a pair of security gateways (*network-to-network*), or between a security gateway and a host (*network-to-host*). IPSec helps to create authenticated and confidential packet for the IP layer [16].

2-9 Concept of honey pot

A honey pot is security resource whose value lies in being probed, attacked, or compromised [8, 10]. Honey pot is a monitored system on the Internet serving the purpose of attracting and trapping attackers who attempt to penetrate the protected servers on a network. In general, any network activities observed at honey pots are considered suspicious, and it is possible to capture the latest intrusions based on the analysis of these activities. [6]. Honey pots 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 [17].

A Honey pot is a software based security device deployed to attract hackers by displaying services and open ports which are potentially vulnerable while the attackers are diverted their activities can then be monitored and analyzed to identify current attack methods and trends [18].

This means that whatever we designate as a honey pot, our expectations and goals are to have the system probed, attacked, and potentially exploited. Honey pots are different from most security tools in that they can take on different manifestations. Most of the security technologies used today was designed to address specific problems, but the honey pots are not limited to solving a single, specific problem. Instead, honey pots are a highly flexible tool that can be applied to a variety of different situations.

2-10 Types of honey pots

Honey pots can be classified based on their arrangement and based on their level of Interaction, Based on Implementation and base on purpose [17]:

2-10-1 By level of interaction

1 Low Interaction

Low-interaction honey pots are the simplest in terms of implementation and typically are the easiest to install, configure, deploy, and maintain because of their simple design and basic functionality. These honey pots merely emulate a variety of services. So, the attacker is limited to interact with these pre designated services [10].

In fact, the main function of the low-interaction honey pots is detection, specifically of unauthorized scans or unauthorized connection attempts.

2 High Interaction

The high-interaction honey pots are different from low-interaction honey pots in terms of implementation and collecting information. High-interaction honey pots utilize actual operating systems rather than emulations. As actual operating systems are used in the high-interaction honey pots, the attacker obtains a more realistic experience, and we are able to gather more information about intended attacks. This makes high-interaction honey pots useful when one wishes to capture details of vulnerabilities or exploits that are not yet known to the public [10].

3 Medium-Interaction Honey pots

Medium-interaction honey pots try to combine the benefits of both approaches (low interaction and high interaction) with regard to botnet detection and malware collection while removing their shortcomings. The medium-interaction honey pots do not aim at fully simulating a full operational system environment or implement all details of an application protocol. The role of medium-interaction honey pots do is provide sufficient responses that known exploits await on certain ports that will trick them into sending their payload [10].

2-10-2 By Implementation

1 Physical

- Real machines
- Own IP Addresses
- Often high-interactive

2 Virtual

Simulated by other machines that: Respond to the traffic sent to the honey pots May simulate a lot of (different) virtual honey pots at the same time [17].

2-10-3 By purpose

1 Production

Production honey pots are what most people typically think of as a honey pot. They add value to the security of a specific organization and help mitigate risk

[8]. Production honey pots usually are easier to build and deploy than research honey pots because they require less functionality, they also generally give us less information about the attacks or the attackers [8, 10].

Normally, production honey pots are low-interaction honey pots, which are easier to arrange [17].

2 Research

Research honey pots are designed to gain information about the black hat community. Their primary mission is to research the threats organizations may face. If production honey pots are similar to law enforcement, then you can think of research honey pots as counterintelligence, used to gain information on the bad guys. This information lets us better understand who our threats are and how they operate [8]. Research honey pots are complex to deploy and maintain, capture extensive information, and are used primarily by research, military, or government organizations [17].

2-11 Honey net

The concept of a honey net is simple: building a network of standard production systems, just as we could find in most organizations today, and putting this network of systems behind some type of access control device (such as a firewall) and watching what happens [10, 8]. Two or more honey pots on a network form a honey net [11].

Honey net is a network of high-interaction honey pots. It is used in large networks in which one honey pot is not sufficient to monitor all kind of system and network activities [19].

Honey nets represent the extreme of high-interaction honey pots. Not only they provide the attacker with a complete operating system to attack and interact with, they may also provide multiple honey pots. Honey nets are nothing more than a variety of standard systems deployed within a highly controlled network. By their nature, these systems become honey pots, since their value is in being probed, attacked, or compromised. The controlled network captures all the activity that happens within the Honey net and decreases the risk by containing the attacker's activity [8]. It extends the concept of a single Honey pot to a highly controlled network of Honey pots. A Honey net is a specialized network architecture configured in a way to achieve [9]: Data Control, Data Capture and Data Collection.

Data Control is what mitigates risk. It controls the attacker's activity by limiting what can happen inbound and outbound. The risk is that once an attacker compromises a system within the Honey net, they can use that system to attack other non-Honey net systems, such as organizations on the Internet. The attacker has to be controlled so they cannot do that. They can attack other systems within the Honey net, but we have to protect non-Honey net systems.

Data Capture is what collecting all the activity that happens inbound, outbound, or within the Honey net.

Data Collection is only for organizations that have multiple Honey nets logically or physically distributed around the world have to collect all of the captured data and store it in a central location [20].

A Honey net is a network designed to be compromised, not to be used for production traffic. Any traffic entering or leaving the network is suspicious by definition [20].

2-12 Virtual honeynet

Virtualization is a technology that allows running multiple virtual machines on a single physical machine. Each virtual machine can be an independent Operating System installation. Running concurrently with others this is achieved by sharing the machine’s physical resources such as CPU, memory, storage and peripherals through specialized software across multiple environments. This reduces project hardware costs [9].

A virtual honey net is a solution that allows you to run everything you need on a single computer. We use the term virtual because different operating systems have the “appearance” of running on their own independent computers, which are not real machines. These solutions are possible because of virtualization software that allows running multiple operating systems at the same time on the same hardware. Virtual honey nets are not a radically new technology; they simply take the concept of honey net technologies and implement them into a single system [10].

Virtual Honey nets combine all the elements of a Honey net onto one physical system. Not only are all three requirements of Data Control, Data Capture, and Data Collection met, but the actual honey pots themselves run on the single system. One such example is the use of VMware. VMware allows various operating systems to run at the same time on the same system [20].

2-13 Related works

Based on honey pot techniques researchers have developed many methods and tools for the collection of attacker's data. The paper of **Experiences with a Generation III Virtual Honey net** [9] this paper proposes a methodology for establishing a virtual Honey net on a VMware Server running Honeywell CDROM Room. The implementation is specific to a Linux based host having a single physical network interface card. In addition, the honey net project **Know Your Enemy: Honey nets** [20] the purpose of this paper was to discuss what a Honey net is, its value, how it works, and the risks/issues involved. These papers are considered as main sources of our work provide useful guidelines for the implementation of honey nets and practically experimental tools which have been used in different honey net projects. Among them there are some honey pot projects which are related to our work.

One of the main references which we used often was research outcomes of **A Hybrid Honey pot Scheme for Distributed Denial of Service Attack** [2] the main goal of this work was presents a hybrid honey pot scheme that combines low and high interaction honey pots to mitigate the shortcomings of both types. The low interaction honey pots are used to emulate operating systems and services, and for any outbound connection, they act as a proxy to forward the packets to real systems in high interaction honey pot. The scheme is tested by applying Distributed Denial of Service attack (DDOS) against the system, and a significant enhancement to the system security is achieved [2]. This design is achieved by implementing Honey systems, and making these systems look like physical machines; this Honeyed provide the first level of security. The second level is the honey nets, which are placed behind the Honeyed. So as we can see this work provides many levels of security that decrease the probability of an attacker targeting production systems. **But** they do not analyze the gathered information in the honeynet [2].

Also we benefited from the research paper of **Data Collection and Data Analysis in Honey pots and Honey nets [1]** this paper combination of two research areas, namely the data analysis in honey pots and honey nets and the incident taxonomy. Research in area of the data analysis has resulted in a number of papers, discussing specific topics, concerning design of data analysis and analysis of specific aspect of attack [1]. The main goal of this paper was to allow easy analysis of data from low-interaction and medium-interaction honey pots. **Unfortunately**, it does not focus on high-interaction honey pots and issues of secure connection between honey pots and itself.

Another interesting paper **Honey pots in network security** [11] this paper discusses a proposed model for honey pot to solve the problem of small scale industries which is the hybrid structure of Snort, Nmap, Xprobe2, and P0f. This model captures the activities of attackers and maintains a log for all these activities. The kind of honey pot has been used are Honeyed and Tcpdump to simulate set of services such as HTTP, POP3, FTP, TELNET So these are the main services where the honey pot can work for and provide the security for the network from the hackers[11]. In addition, this paper discussed the tool that is Honey pot for the Intruder Detecting services and discusses the various effects of the intruder attack on the web services. **However**, we can see some deficiencies in this paper comparing it with ours such as the comparison between the existing method and the proposed method should have to be done. Also there should be the need of the implementations of the some algorithms and the techniques like connection tracking, protocol analysis and the pattern detection, as they don't have simulation for the whole network or system because they use low interaction honey pot (Honeyed).

Finally none of these works deployed high-interaction honey pots such as honey nets that interact with real system and control, capture then analysis data and collect massive of data that we need it, which has been done in our work.

In order to improve intrusion detection systems and extend the scalability and flexibility of the honey pots. This approach will be helpful when we designed our own virtual honeynet architecture.

Table 2-1 comparative between related works

The name of paper	The level of Honeypot used	Deficiency
Experiences with a Generation III Virtual Honey net	High interaction	The implementation was limited to a Linux based only
Know Your Enemy: Honey nets	High interaction	There were no information about the analysis
A Hybrid Honey pot Scheme for Distributed Denial of Service Attack	Low and high interaction	they do not analyze the gathered information in the honeynet

Data Collection and Data Analysis in Honey pots and Honey nets	Low and medium interaction	does not focus on high-interaction honey pots and issues of secure connection between honey pots and itself
Honey pots in network security	Low interaction	It does not have simulation for the whole network or system because they use low interaction honey pot (Honeyed).

Chapter III

Methodology

Chapter III Methodology

3-1 Introduction

Honey pots are closely supervised decoys that are employed in a network to read the track of hackers and to alert network administrators of a possible intrusion. Using honey pots provides a cost-effective solution to increase the security structure of an organization. Even though they are not a panacea for security breaches, they are useful as a tool for network adaption and intrusion detection [17]. Honey pots are being used increasingly by organizations to detect the existence of attackers. This means that the defenders can keep the attackers ring fenced where they can do little harm and learn more about the tactics that are currently deployed in order to fine tune their defenses appropriately, There are many advantages of Honey pots because of their simple concept that gives them powerful strengths However a Honey pot does not replace existing security technologies but can work alongside them to track and capture activity that occur on the system where it is deployed, However it will only capture activity that is directed at the Honey pot itself. The Honey pot can also be at risk of sabotage and could be used to attack other connected systems [18].

3-2 Honey net Architectures

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

3-2-1 Generation I

Gen I Honey net was developed in 1999 by the Honey net Project. The architecture was simple with a firewall aided by IDS as the gateway and Honey pots placed behind it. This architecture required 2 interfaces on the Honeywell gateway, one faces the external network and the other faces the Honey pot's internal network.

3-2-2 Generation II & III

Change in architecture was brought about by the introduction of a single device that handles the data control and data capture mechanisms of the Honey net called the IDS Gateway or the Honeywell. This is implemented as a transparent bridge. Gen II and Gen III Honey nets have the same architecture, with the only difference being improvements in deployment and management in Gen III Honey nets along with the addition of a Sebeka server built in the gateway – this is known as the Honeywell [19].

Below you can see some specific advantages and disadvantages of the virtual honey nets compare to traditional honey nets [19].

Table 3-1 Virtual Honey net

Advantages	Disadvantages
• reduced cost • easier maintenance • Portable	• Fingerprinting risk • Limitation of OS according to hardware ,and Virtualization software

3-3 Implemented Architecture

Build honey net system architecture in VMware environment to perform the data collection using sebek software and snort software for detection and alerting the administrator, then analyze the information collected by a honey net system using weir shark program.

The implemented architecture is illustrated in Figure 3-1 below, and the components of figure 3-1 are:

1- The public internet which we are connected to.

2- Our only physical device that contains the virtual honey net system.

3- Honeywell Roo Virtual machine, Honeywell gateway for high-interaction honey pots.

4- Virtual honey pot with Ubuntu operating system.

5- Virtual honey pot with windows XP operating system.

6- A remote management machine to remotely control the collection of attack data and to monitor the activities and processes on the honey pots.

7- The attacker who is attempts to login into our system through the physical PC.

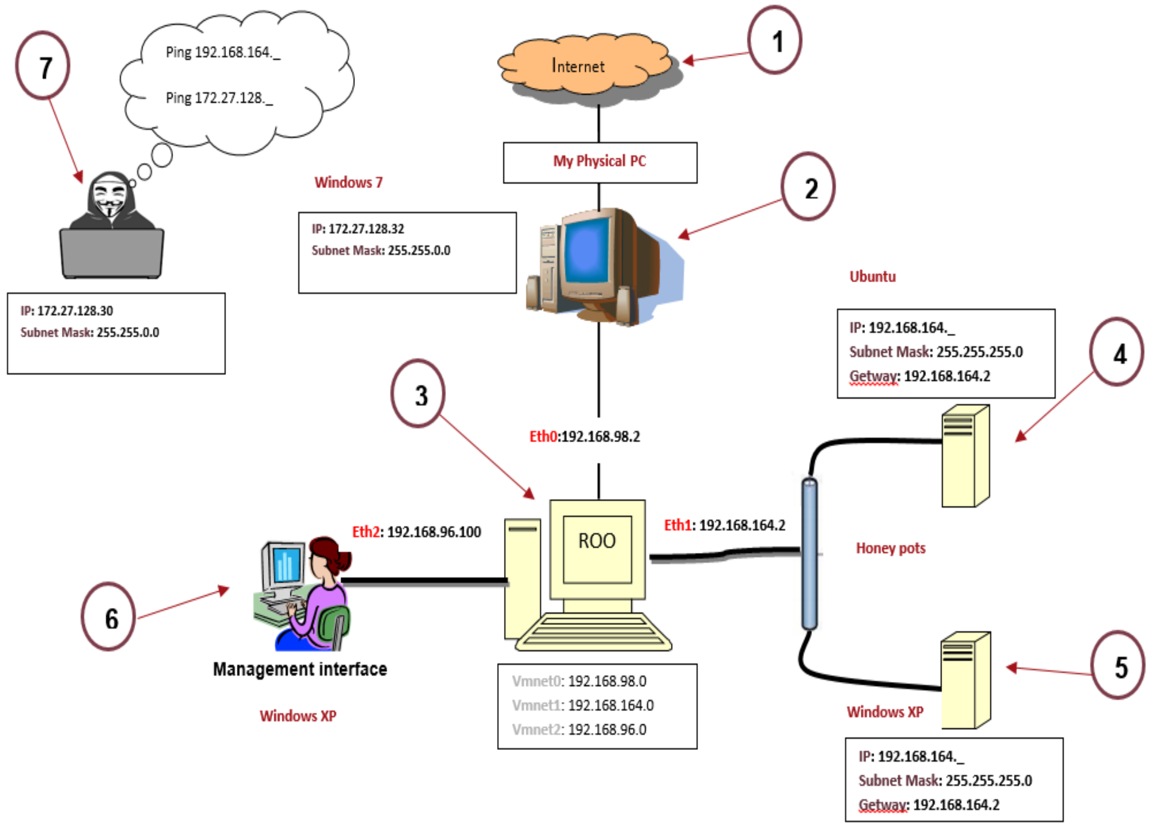


Figure 3-1 virtual honeynet

In this implementation, we used only one physical machine, which contains the virtual honey pots, and a remote management machine to remotely control the collection of attack data and to monitor the activities and processes on the honey pots. All of the honey pots are deployed and configured on the virtual machines.

In our implemented architecture, we used Honeywell CDROM Roo version 1.4. Honeywell CDROM is a bootable CDROM operating system built on CentOS for installing, deploying and maintaining a Honey net. The purpose of the Honeywell CDROM is to automate the installation and maintenance of a Honey net and provide data analysis support for all activity within the Honey net [9].

As we mentioned before, Honeywell is a main component of the GEN III honey net, since it provides the main tasks of the honey net such as Data Capture, Data Control and Data Analysis [19]. Thus this system gives to administrators more flexibility and easy control. Honeywell Roo includes the following security tools [19, 9]:

- **Tcpdump:** Packet analyzer.
- **Sebek:** Data capture tool.
- **Snort**: Intrusion Detection System (IDS).

- **Snort inline**: Snort inline is basically a modified version of Snort that accepts packets from iptables. It then uses new rule types (drop, sdrop, reject) to tell iptables whether the packet should be dropped, rejected, modified, or allowed to pass based on a snort rule set. It is an Intrusion Prevention System (IPS) that uses existing Intrusion Detection System (IDS) signatures to make decisions on packets that traverse snort_inline [21].
- **Iptables:** Iptables is a Linux firewall integrated into the kernel. It is a generic table structure for the definition of rule sets. Each rule within an IP table consists of a number of classifiers (iptables matches) and one connected action (iptables target) [21].
- **Hflow2**: A data correlation tool for Honey net data analysis.
- **P0f:** Passive OS fingerprinting tool.
- **Walleye Web Interface:** a web based interface for Honeywell configuration, administration and data analysis.
- **Argus:** Argus is a real time flow monitor that is designed to perform comprehensive IP network traffic auditing [21].
- **Menu:** Graphical menu developed by the Honey net Project to maintain and control a running Honeywell [21].

3-4 VMware networking

According to our virtual honey net architecture, Honeywell must have three virtual NICs:

- eth0 – connected to the external network, bridged to host’s eth0 physical NIC.
- eth1 – connected to the honey net through vmnet1 host-only virtual interface.
- eth2 – interface for remote administration and management, through vmnet2 host-only.

3-5 Walleye Web Interface

Walleye Web Interface is a web-based Graphical User Interface that is used for Honeywell configuration, administration and data analysis [19]. We used this web interface in order to analyze the inbound and outbound traffic through a web browser client by typing https://192.x.x.x (where 192.x.x.x is the Public IP address).For security reason, this interface is accessible over port 443(HTTS). Walleye has two main functionalities: Data Analysis and

System Administration. The Data Analysis is used for analyzing real-time flows, overview of incoming and outgoing flows; Sebek based data, alert flows by the Snort IDS and activity summary per day. Walleye also provides downloading the packet data in pap format which we used for further analysis with Wire shark on our remote management machine [19].

3-6 Sebek

Currently the state of the art in honey net area is a technology called Sebek. It can log almost all actions performed by a user on the honey pot and send the record to a central collecting server. The main unique features of Sebek are [22]:

- Records all text typed in a terminal.
- saves all tools transferred from or to the host even if they are deleted afterwards.
- Records all binaries executed on the host even if they are deleted afterwards.
- can filter what should be recorded (so the analysts do not get drowned in the log files).
- hides the network activity related to Sebek from all users on the network.

3-7 Configuring virtual high-interaction honey pots

In this section, we will talk about setting and configuration of virtual high-interaction honey pots in our virtual honey net implementation. We set up the virtual honey pots in two steps. In the first step, we installed the operating systems on our virtual machines, which is very similar to installing an operating system on a physical machine. In the second step, we installed additional software and vulnerable applications in order to attract more attackers to the honey pots and collect information about them. Since we deployed our high-interaction honey pots using virtualization software it was not so easy to make concrete decision on choosing the operating system for the honey pots. To choose the "right" operating system depends on the need and the available software/hardware version and parameters. We decided to use Windows and Linux operating systems in order to study attacks against different operating systems and the tools, methods used by attackers.

Therefore we used a Sebek Client as a hidden data capture tool to log attackers activities on the Windows honey pot and send them to the Sebeka Server which is located on Honeywell (Roo) machine. Sebek can be hard detectable by attackers.

3-7-1 Honey pot 1 -Ubuntu

Ubuntu is one of the most widely used Linux Distribution, which provides a good platform to study zero-day attacker exploits [19].

3-7-2 Honey pot 2 – Windows

As a second virtual high-interaction honey pot we choose a guest machine that was running Windows XP. To offer some “honey” for the attackers, we set up an insecure web server and deployed common applications and services with some open ports on the Windows honey pot.

3-8 Scanning the Internet ports using Nmap

Nmap is a free, open-source port scanner available for both UNIX and Windows. It has an optional graphical front-end, NmapFE, and supports a wide variety of scan types, each one with different benefits and drawbacks [23].

Nmap is most often used by network administrators and IT security professionals to scan enterprise networks, looking for live hosts, specific services, or specific operating systems. Part of the beauty of Nmap is its ability to create IP packets from scratch and send them out utilizing unique methodologies to perform the above-mentioned types of scans and more [24].

Table 3.2 Common Port

Port	Service	Protocol
20/21	FTP	TCP
22	SSH	TCP
23	Telnet	TCP
25	SMTP	TCP
53	DNS	TCP/UDP
67/68	DHCP	UDP
69	TFTP	UDP
80	HTTP	TCP
110	POP3	TCP
161/162	SNMP	UDP
443	HTTPS	TCP

3-9 Wire shark

Wire shark is the world's most popular network analyzer. Available for free to all as an open source tool, Wire shark runs on a variety of platforms and offers the ideal 'first responder' tool for IT professionals. It is maintained by an active community of developers from all over the world [25].

3-10 The honey pot architecture implementation

In this section, we will describe how to set up the honey net. We will also indicate the issues that are involved during the deployment of the honey net.

Table 3.3 List of hardware and software resources used in our implementation

Software tools, OS	Description	Specifications
Windows 10 Pro	physical machine	Processor: intel(R) core(TM)i3 CPU M380 @ 2.53GHz Installed memory(RAM): 4.00GB (3.80 GB usable) System type: 64-bit operating system
Linux, Honeywell Roo (CENTOS 6)	Virtual machine, Honeywell gateway for high-interaction honey pots	Roo 1.4 RAM:1GB ; Hard Disk:20GB Network:3 [1 –bridged(eth0),2- host-only(eth1, eth2)], eth --> vmnet0
Linux, Ubuntu 8.04 (Gutsy)	Virtual honey pot	RAM:1GB Hard Disk:40 GB Network: host-only vmnet1
Windows XP (Gutsy)	Virtual honey pot	RAM:512MB Hard Disk:40 GB Network: host-only vmnet1

Windows XP Professional (Gutsy)	Remote Management machine	RAM:512MB Hard Disk:40 GB Network: host-only vmnet2
Sebek	Data Capture tool based in client-server architecture (Server on Honeywell gw, Clients on virtual honey pots)	Sebek Server 3.0.2(Honeywell) Sebek 3.2.0b client (Linux) Sebek 3.0.5 client (Windows)

In Table 3.3you can see the list of hardware and software resources which we have used in the implementation. As hardware, we use one physical machine and four virtual machines: two are used for the honey pots, and another one is used as Honeywell (Roo) host for the virtual honey net, and the last one is used as the remote management machine

Chapter IV

Implementation

Chapter IV Implementation

4-1 Introduction

We decided to deploy high-interaction honeypots, to collect high-level information about the attacks, and monitor the activities carried out by different kind attackers. In this chapter we present the whole architecture used in our work In the virtual honeynet Attackers without any restrictions can get access to high-interaction honeypots which have high fidelity. By using a virtual honeynet, we can reduce the cost of deploying honeypots. All of the honeypots are deployed and configured on the virtual machines. There is a disadvantage of the implemented architecture is that it has no filter mechanism for the incoming traffic. That means that, no port-scan attempts or connections to closed ports will be filtered, and high interaction honeypots can become overwhelmed by receiving a large volume of such traffic. It can also receive uninterested traffic such as unestablished TCP connections that have been received many times before. That is why we propose to filter out the zero-day attack by then generate signatures for them as a future work.

Installation of the Honeywall is very easy, because the “Honeynet project” provides the Honeywall CD-ROM which can setup within a few minutes. All the captured data in the Honeywall will be written to a database.

Data Capture collects data of activities of the attacker. The simplest mechanism for Data Capture is to capture all incoming and outgoing traffic. This can be done by tools like Tcpdump or Wireshark. These tools simply log all packets passing through the network interfaces and write them into a database for later analysis. Moreover, all events that are logged by an installed IDS or IPS are also logged into a database.

4-2 VMware

VMware is a company providing many products focused on virtualization. Its main products are: VMware Workstation, VMware Player and VMware ESX server. In our implementation, we used VMware workstation version 14

.

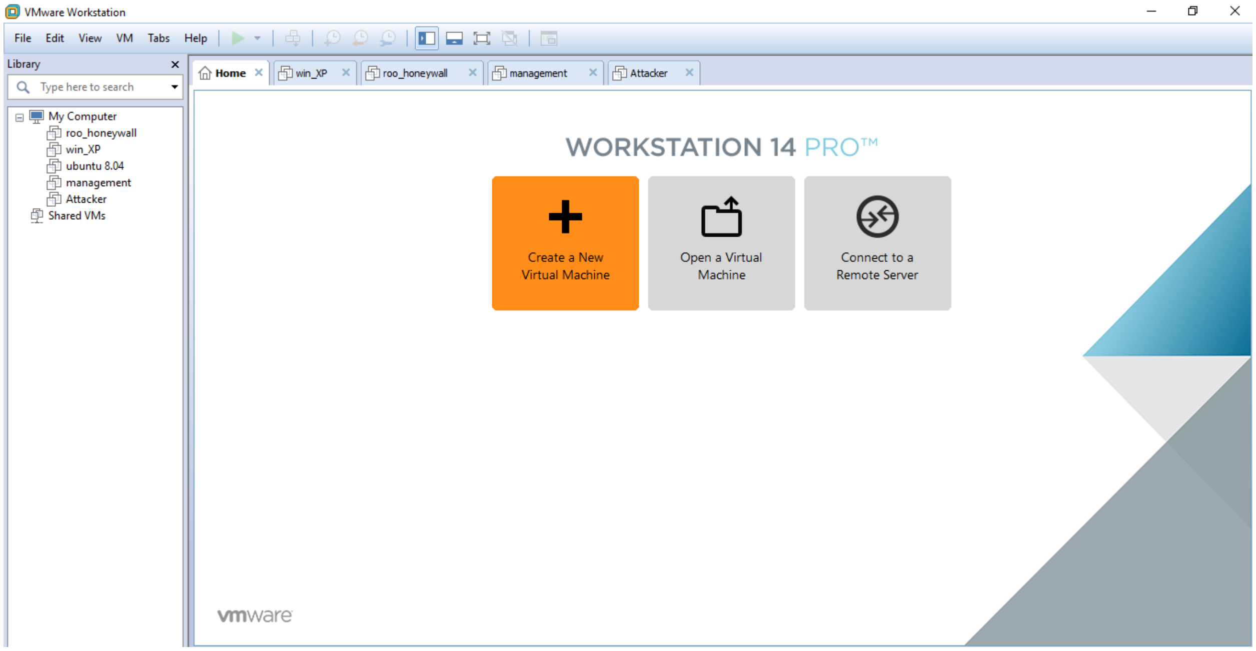


Figure 4-1 Installed virtual machines on VMware workstation 14 PRO

As it has been shown in figure 4-1 VMware workstation allows simultaneously running four virtual machines with using less hardware resources. In our implementation, we installed VMware workstation version 14.0 and created four virtual machines upon it: Honey wall, remote management machine and two Honey pots (Windows, Ubuntu). You can see these virtual machines in Figure 4-1. Honey wall boots from an iso-image Honey wall Roo 1.4 and its operating system is based on CentOS 5. It is configured with 1GB RAM, 40 GB storage and three network interfaces: two host-only interfaces and one bridged interface. Linux-based Ubuntu Honey pot was configured with 1GB RAM, 40 GB storage and one host-only network interface, but Windows Honey pot with 512 MB RAM, 40 GB storage and one host-only network interface.

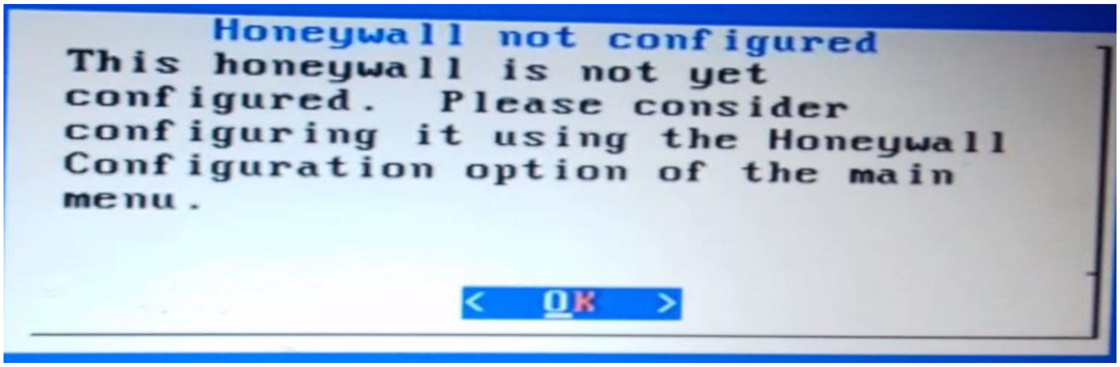
4-3 Honeywall configuration

Honeywall installation is fully automated. We downloaded the latest release of Honey wall-Roo 1.4 ISO image from the https://projects.honeynet.org/honeywall/ and burned it to the CD-ROM that is bootable. Then we booted our virtual machine off this CD-ROM .Once the installation was completed, the new Linux-based OS was booted and configuration process was started [19].

There are two methods for configuring the Honey wall:

- Dialog menu interface. It is more common configuration method which is mostly used in the initial 32 setup of Honey wall. This interface is opened automatically during the first login as root after the installation. It's also possible to run it by typing the command *menu* from console.

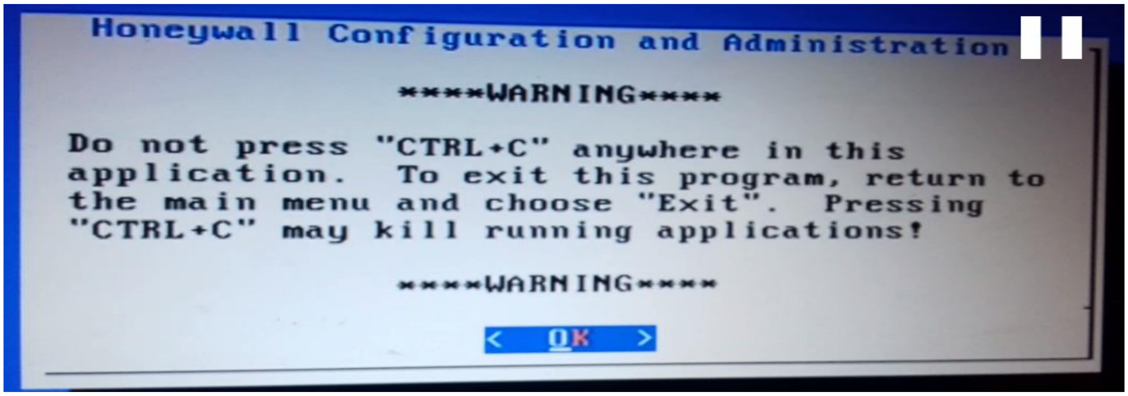
$su - // to access as root to the configuration if it's not work type "menu"



```
Honeywall not configured
This honeywall is not yet
configured.  Please consider
configuring it using the Honeywall
Configuration option of the main
menu.

< OK >
```

Figure 4-2 Honey wall configuration

Figure 4-2 is the first window of Honeywall configuration that tell you Honeywall is not yet configured and you have to press OK to continue.



```
Honeywall Configuration and Administration

*****WARNING*****

Do not press "CTRL+C" anywhere in this
application.  To exit this program, return to
the main menu and choose "Exit".  Pressing
"CTRL+C" may kill running applications!

*****WARNING*****

< OK >
```

Figure 4-3 Honey wall configuration

Figure 4-3 is the second window of configuring Honeywall that warning you from pressing (CTRL+C) to exit from configuration also you have to press OK to continue.

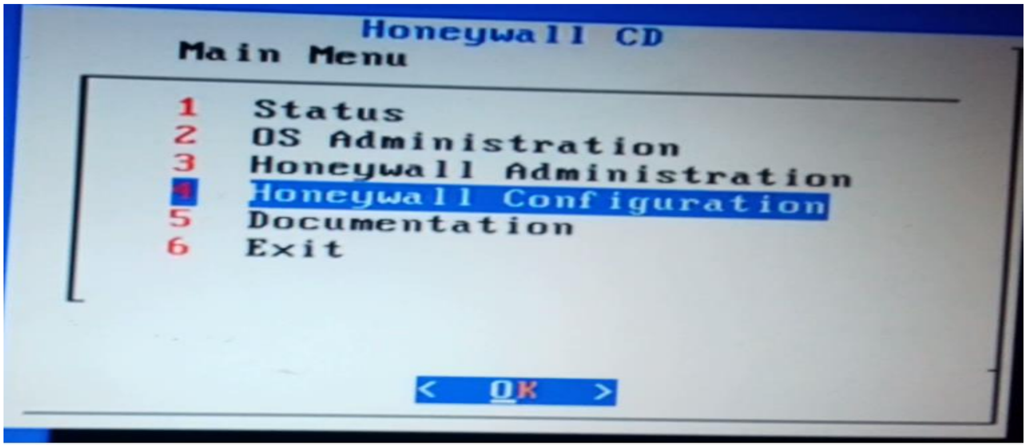
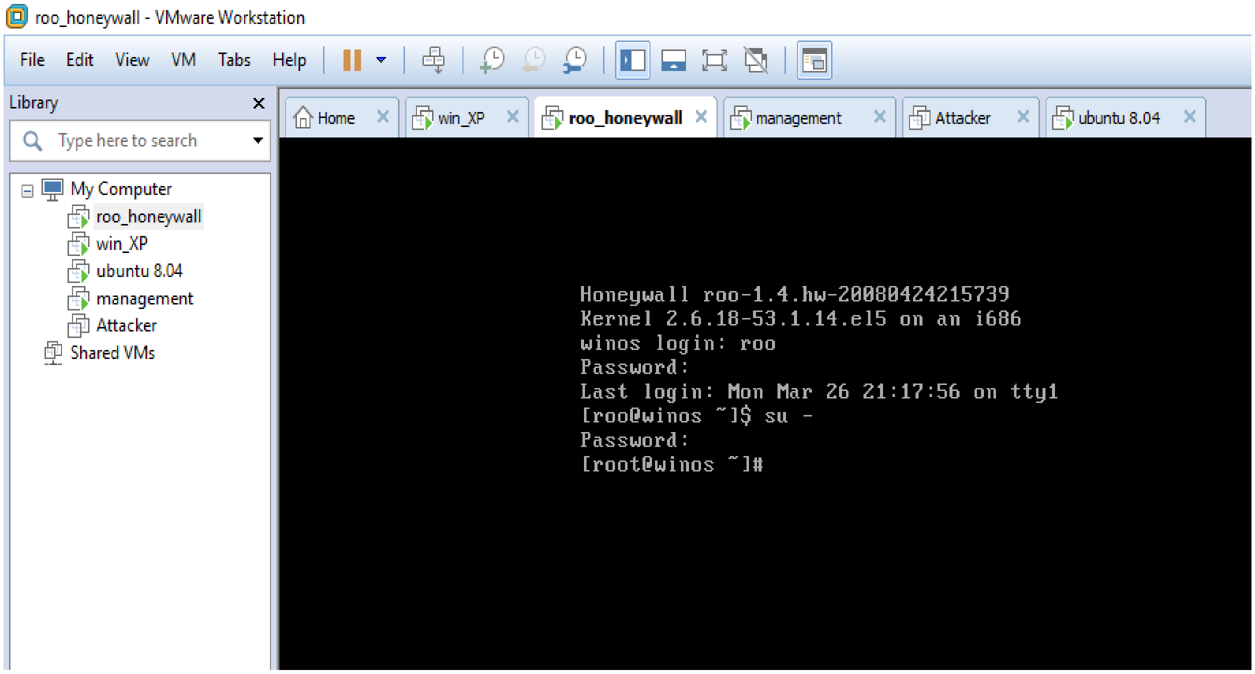


Figure 4-4 the main menu of Honey wall configuration

Figure 4-4 is the main menu of Honeywall configuration you have to choose the option Honeywall configuration and press OK to start the configuration.

- Manually create Honeywall. conf. Honey wall Roo comes with default configuration file, Honeywall. conf, which is ASCII text file.

4-4 Login Roo



```
Honeywall roo-1.4.hw-20080424215739
Kernel 2.6.18-53.1.14.el5 on an i686
winos login: roo
Password:
Last login: Mon Mar 26 21:17:56 on tty1
[roo@winos ~]$ su -
Password:
[root@winos ~]#
```

Figure 4-5 login Roo

Honey wall comes with two user accounts (*Roo* and *root*) which they are used to login to the system and Walleye GUI.

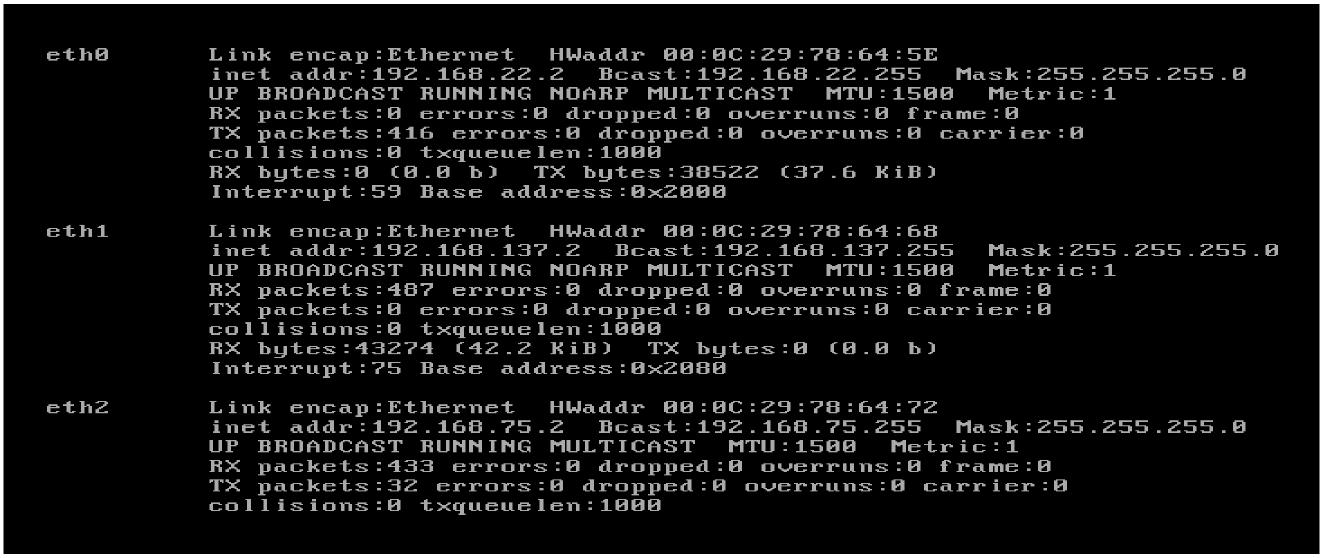
As it has been illustrated in figure 4-5 *Root* login is not accessible by default so one has to login as *Roo* and then use *"su -"* command to have root access.

When you access root then you can check your network cards by typing the following command

#

```
[root@winos ~]# ifconfig
```

However the result was the following:



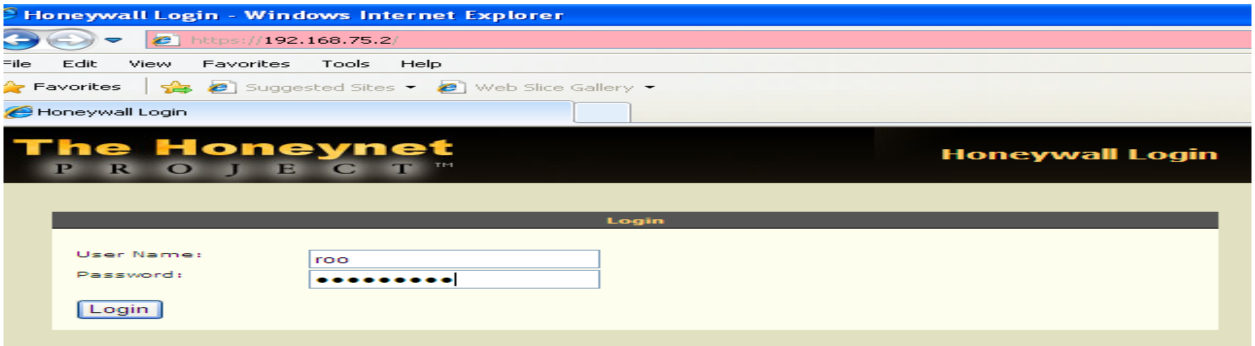
```
eth0      Link encap:Ethernet  HWaddr 00:0C:29:78:64:5E
          inet addr:192.168.22.2  Bcast:192.168.22.255  Mask:255.255.255.0
          UP BROADCAST RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:416 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:38522 (37.6 KiB)
          Interrupt:59 Base address:0x2000

eth1      Link encap:Ethernet  HWaddr 00:0C:29:78:64:68
          inet addr:192.168.137.2  Bcast:192.168.137.255  Mask:255.255.255.0
          UP BROADCAST RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:487 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:43274 (42.2 KiB)  TX bytes:0 (0.0 b)
          Interrupt:75 Base address:0x2080

eth2      Link encap:Ethernet  HWaddr 00:0C:29:78:64:72
          inet addr:192.168.75.2  Bcast:192.168.75.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:433 errors:0 dropped:0 overruns:0 frame:0
          TX packets:32 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
```

Figure 4-6 network cards

As it has been showed in figure, 4-6 each Ethernet has specific static IP in the range of it is network and Hardwar address.

4-5 Walleye Web Interface

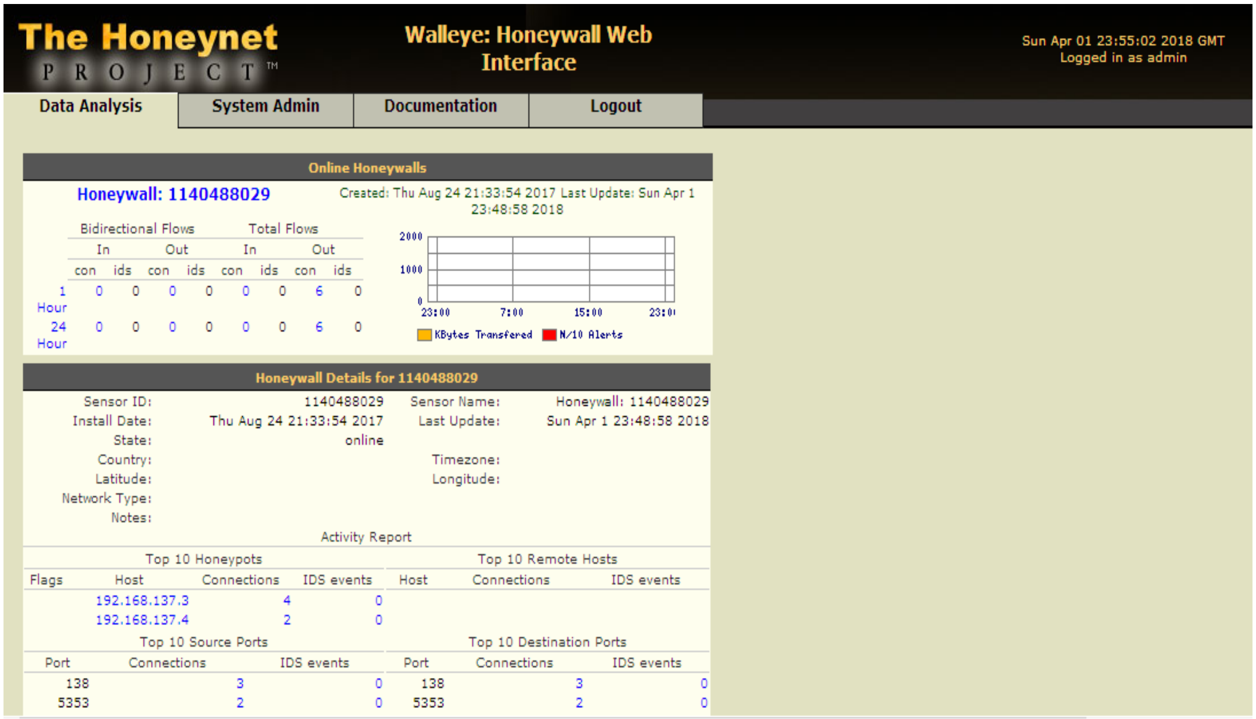
Walleye also provides downloading the packet data in pcap format which we used for further analysis with Wire shark on our remote management machine. System Administration allows remote Honey wall administration, and also provides access to the Honey wall configuration, thus all the settings can be updated from this interface too.



User Name: roo
Password: ●●●●●●●●●●

Figure 4-7 Honey wall login

Figure 4-7 is the main screen of Honeywall to login, so it will open walleye, which is Honeywall web interface.



Sun Apr 01 23:55:02 2018 GMT
Logged in as admin

Online Honeywalls

Honeywall: 1140488029

Created: Thu Aug 24 21:33:54 2017 Last Update: Sun Apr 1 23:48:58 2018

	Bidirectional Flows				Total Flows			
	In		Out		In		Out	
	con	ids	con	ids	con	ids	con	ids
1 Hour	0	0	0	0	0	0	6	0
24 Hour	0	0	0	0	0	0	6	0

Honeywall Details for 1140488029

Sensor ID:	1140488029	Sensor Name:	Honeywall: 1140488029
Install Date:	Thu Aug 24 21:33:54 2017	Last Update:	Sun Apr 1 23:48:58 2018
State:	online		
Country:		Timezone:	
Latitude:		Longitude:	
Network Type:			
Notes:			

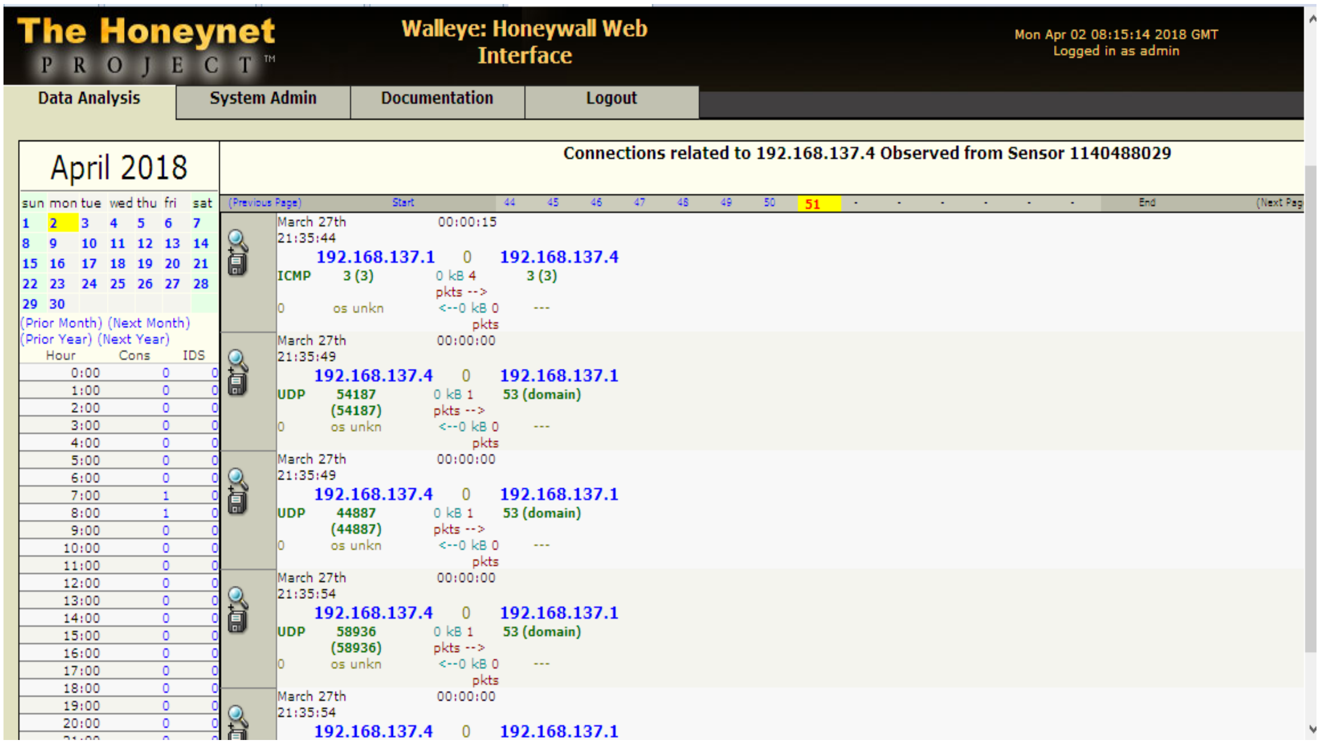
Activity Report

Top 10 Honeypots				Top 10 Remote Hosts		
Flags	Host	Connections	IDS events	Host	Connections	IDS events
	192.168.137.3	4	0			
	192.168.137.4	2	0			

Top 10 Source Ports			Top 10 Destination Ports		
Port	Connections	IDS events	Port	Connections	IDS events
138	3	0	138	3	0
5353	2	0	5353	2	0

Figure 4-8 Honey wall web interface

In Figure 4-8 you can see the examination of all connections for one day: aggregated connections to each honey pot per day, number of IDS events, and most connected destination and source IP addresses and ports. Moreover we can analyze one connection in detail for the particular IP address. In the following screen you can see that each line contains information about protocol type, number and bytes of the packets, and OS type of source IP address of the connection.



Mon Apr 02 08:15:14 2018 GMT
Logged in as admin

April 2018

sun	mon	tue	wed	thu	fri	sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

(Prior Month) (Next Month)
(Prior Year) (Next Year)

Hour	Cons	IDS
0:00	0	0
1:00	0	0
2:00	0	0
3:00	0	0
4:00	0	0
5:00	0	0
6:00	0	0
7:00	1	0
8:00	1	0
9:00	0	0
10:00	0	0
11:00	0	0
12:00	0	0
13:00	0	0
14:00	0	0
15:00	0	0
16:00	0	0
17:00	0	0
18:00	0	0
19:00	0	0
20:00	0	0

Connections related to 192.168.137.4 Observed from Sensor 1140488029

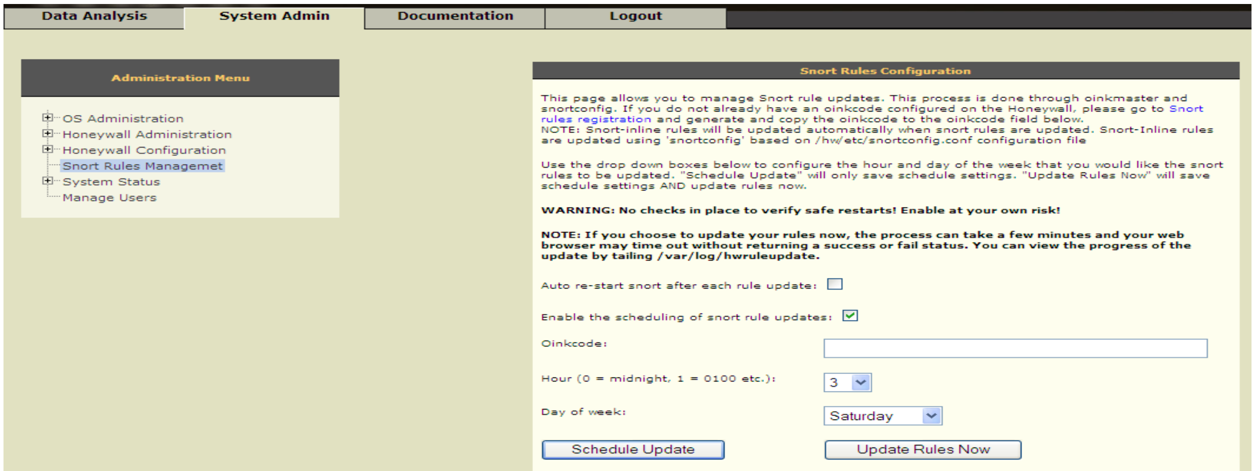
(Previous Page) Start 44 45 46 47 48 49 50 51 - - - - - - End (Next Pag

```
March 27th        00:00:15
21:35:44
     192.168.137.1     0     192.168.137.4
ICMP      3 (3)      0 kB 4       3 (3)
                     pkts -->
0         os unkn    <--0 kB 0    ---
                          pkts
March 27th        00:00:00
21:35:49
     192.168.137.4     0     192.168.137.1
UDP       54187      0 kB 1     53 (domain)
          (54187)    pkts -->
0         os unkn    <--0 kB 0    ---
                          pkts
March 27th        00:00:00
21:35:49
     192.168.137.4     0     192.168.137.1
UDP       44887      0 kB 1     53 (domain)
          (44887)    pkts -->
0         os unkn    <--0 kB 0    ---
                          pkts
March 27th        00:00:00
21:35:54
     192.168.137.4     0     192.168.137.1
UDP       58936      0 kB 1     53 (domain)
          (58936)    pkts -->
0         os unkn    <--0 kB 0    ---
                          pkts
March 27th        00:00:00
21:35:54
     192.168.137.4     0     192.168.137.1
```

Figure 4-9 Detailed information for a flow

Figure 4-9 showed detailed information about the packet captured.

4-6 Snort IDS and Snort-inline IPS



Snort Rules Configuration

This page allows you to manage Snort rule updates. This process is done through oinkmaster and snortconfig. If you do not already have an oinkcode configured on the Honeywall, please go to Snort rules registration and generate and copy the oinkcode to the oinkcode field below.
NOTE: Snort-inline rules will be updated automatically when snort rules are updated. Snort-Inline rules are updated using 'snortconfig' based on /hw/etc/snortconfig.conf configuration file

Use the drop down boxes below to configure the hour and day of the week that you would like the snort rules to be updated. "Schedule Update" will only save schedule settings. "Update Rules Now" will save schedule settings AND update rules now.

WARNING: No checks in place to verify safe restarts! Enable at your own risk!

NOTE: If you choose to update your rules now, the process can take a few minutes and your web browser may time out without returning a success or fail status. You can view the progress of the update by tailing /var/log/hwruleupdate.

Auto re-start snort after each rule update: ☐

Enable the scheduling of snort rule updates: ☑

Oinkcode:

Hour (0 = midnight, 1 = 0100 etc.): 3

Day of week: Saturday

Figure 4-10 snort rule

Snort as an Intrusion Detection and Prevention System is integrated into Honeywall 1.4. It is an open-source IDS, rule- and signature-based engine that can be run in one of the following modes:

- Sniffer Mode

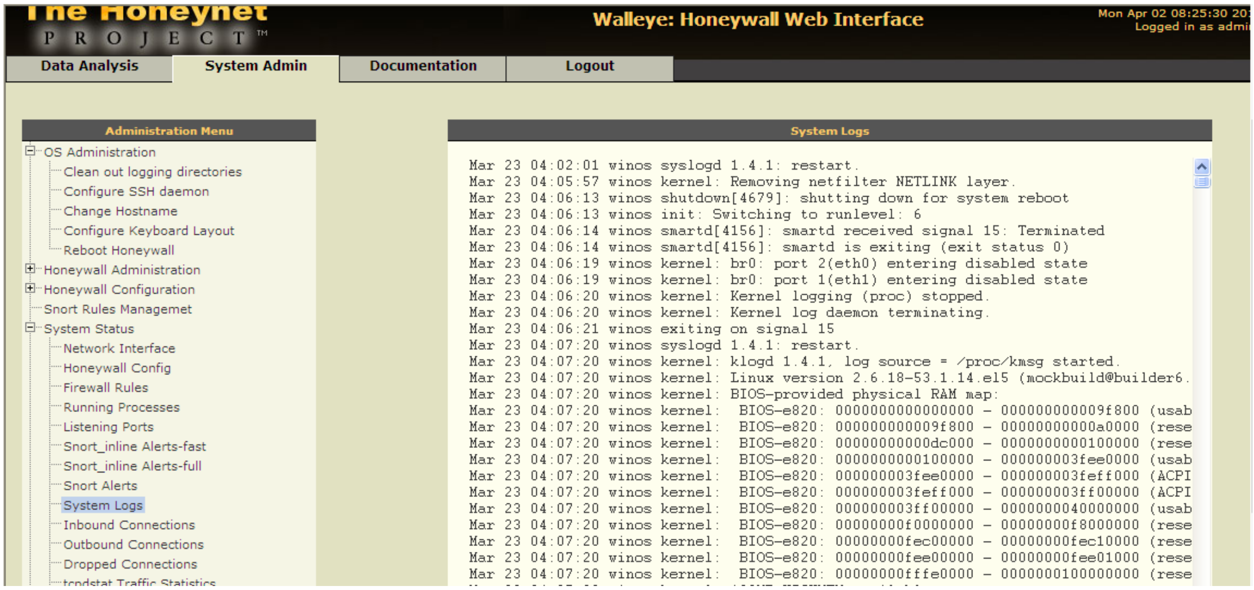
In this mode Snort is used as packet sniffer and displays IP headers on the screen.

- Logger Mode

All packets are logged into the file and can be used for further analysis.

- Network Intrusion Detection Mode

The core mode of Snort. All incoming packets will be analyzed based on the user-defined rules and signatures. Snort will log, detect and alert if there is any anomaly detection in the packets then Inline Mode. In this mode, Snort acts as an Intrusion Prevention System (IPS) which is called Snort-inline. It resides on the Honeywall where the packets are analyzed and monitored using iptables in order to control outgoing packets from the honeypots.

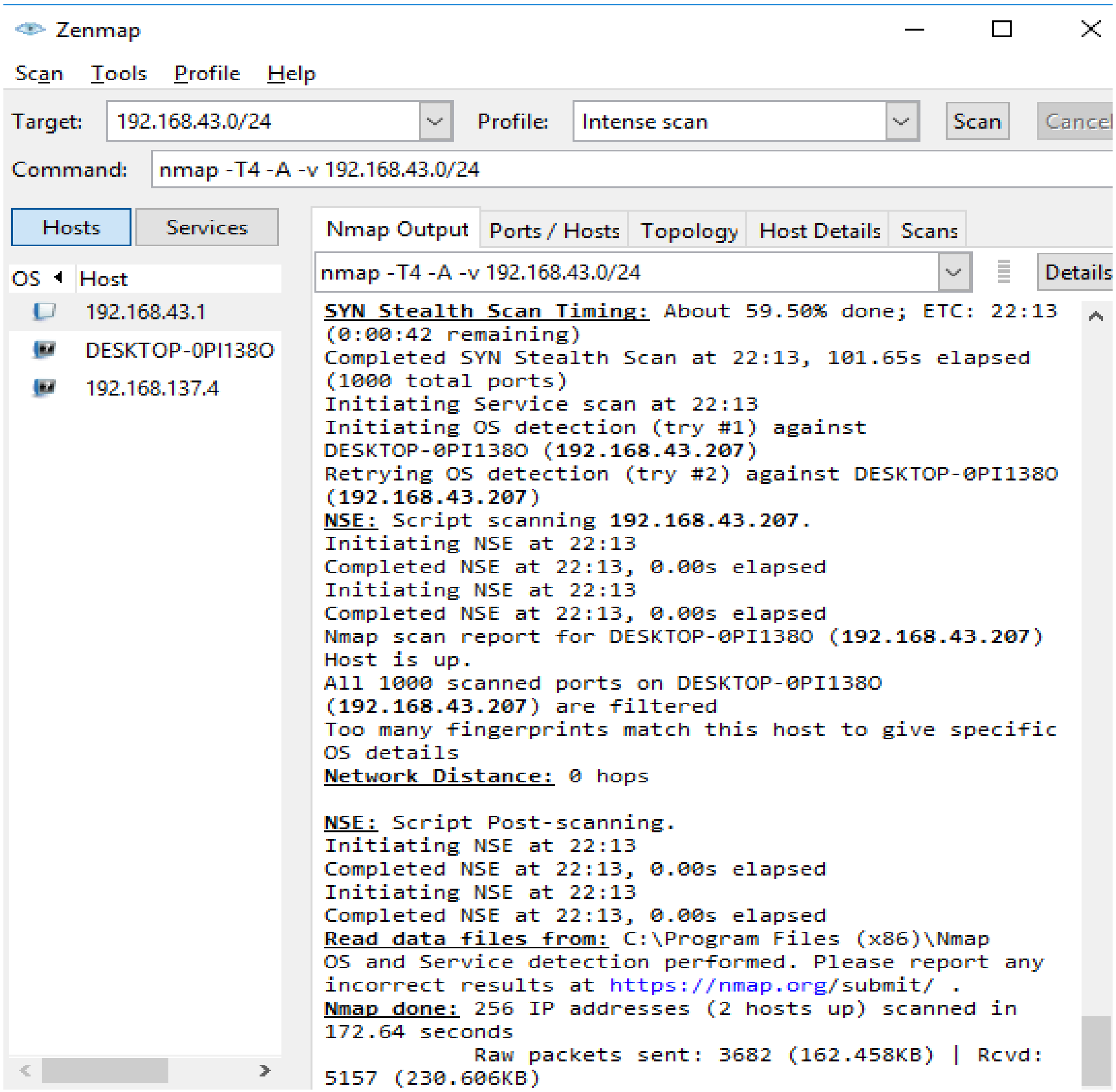


```
Mar 23 04:02:01 winos syslogd 1.4.1: restart.
Mar 23 04:05:57 winos kernel: Removing netfilter NETLINK layer.
Mar 23 04:06:13 winos shutdown[4679]: shutting down for system reboot
Mar 23 04:06:13 winos init: Switching to runlevel: 6
Mar 23 04:06:14 winos smartd[4156]: smartd received signal 15: Terminated
Mar 23 04:06:14 winos smartd[4156]: smartd is exiting (exit status 0)
Mar 23 04:06:19 winos kernel: br0: port 2(eth0) entering disabled state
Mar 23 04:06:19 winos kernel: br0: port 1(eth1) entering disabled state
Mar 23 04:06:20 winos kernel: Kernel logging (proc) stopped.
Mar 23 04:06:20 winos kernel: Kernel log daemon terminating.
Mar 23 04:06:21 winos exiting on signal 15
Mar 23 04:07:20 winos syslogd 1.4.1: restart.
Mar 23 04:07:20 winos kernel: klogd 1.4.1, log source = /proc/kmsg started.
Mar 23 04:07:20 winos kernel: Linux version 2.6.18-53.1.14.el5 (mockbuild@builder6.
Mar 23 04:07:20 winos kernel: BIOS-provided physical RAM map:
Mar 23 04:07:20 winos kernel:  BIOS-e820: 0000000000000000 - 000000000009f800 (usab
Mar 23 04:07:20 winos kernel:  BIOS-e820: 000000000009f800 - 00000000000a0000 (rese
Mar 23 04:07:20 winos kernel:  BIOS-e820: 00000000000dc000 - 0000000000100000 (rese
Mar 23 04:07:20 winos kernel:  BIOS-e820: 0000000000100000 - 000000003fee0000 (usab
Mar 23 04:07:20 winos kernel:  BIOS-e820: 000000003fee0000 - 000000003feff000 (ACPI
Mar 23 04:07:20 winos kernel:  BIOS-e820: 000000003feff000 - 000000003ff00000 (ACPI
Mar 23 04:07:20 winos kernel:  BIOS-e820: 000000003ff00000 - 0000000040000000 (usab
Mar 23 04:07:20 winos kernel:  BIOS-e820: 00000000f0000000 - 00000000f8000000 (rese
Mar 23 04:07:20 winos kernel:  BIOS-e820: 00000000fec00000 - 00000000fec10000 (rese
Mar 23 04:07:20 winos kernel:  BIOS-e820: 00000000fee00000 - 00000000fee01000 (rese
Mar 23 04:07:20 winos kernel:  BIOS-e820: 00000000fffe0000 - 0000000100000000 (rese
```

Figure 4-11 system log file

4-7 Port Scanning

Port Scanning is often used by system administrators to verify the security level of their networks and by attackers to find the vulnerabilities.



Target: 192.168.43.0/24

Profile: Intense scan

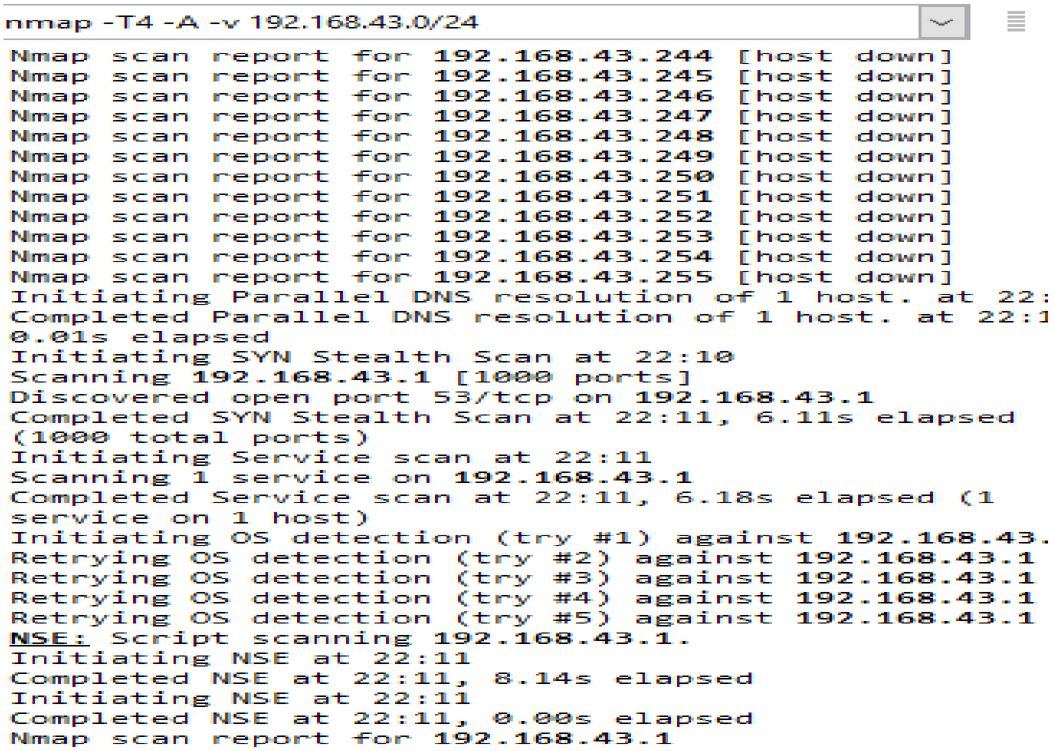
Command: nmap -T4 -A -v 192.168.43.0/24

192.168.43.1
DESKTOP-0PI138O
192.168.137.4

```
nmap -T4 -A -v 192.168.43.0/24

SYN Stealth Scan Timing: About 59.50% done; ETC: 22:13
(0:00:42 remaining)
Completed SYN Stealth Scan at 22:13, 101.65s elapsed
(1000 total ports)
Initiating Service scan at 22:13
Initiating OS detection (try #1) against
DESKTOP-0PI138O (192.168.43.207)
Retrying OS detection (try #2) against DESKTOP-0PI138O
(192.168.43.207)
NSE: Script scanning 192.168.43.207.
Initiating NSE at 22:13
Completed NSE at 22:13, 0.00s elapsed
Initiating NSE at 22:13
Completed NSE at 22:13, 0.00s elapsed
Nmap scan report for DESKTOP-0PI138O (192.168.43.207)
Host is up.
All 1000 scanned ports on DESKTOP-0PI138O
(192.168.43.207) are filtered
Too many fingerprints match this host to give specific
OS details
Network Distance: 0 hops

NSE: Script Post-scanning.
Initiating NSE at 22:13
Completed NSE at 22:13, 0.00s elapsed
Initiating NSE at 22:13
Completed NSE at 22:13, 0.00s elapsed
Read data files from: C:\Program Files (x86)\Nmap
OS and Service detection performed. Please report any
incorrect results at https://nmap.org/submit/ .
Nmap done: 256 IP addresses (2 hosts up) scanned in
172.64 seconds
           Raw packets sent: 3682 (162.458KB) | Rcvd:
5157 (230.606KB)
```

Figure 4-12 Nmap port scanning output

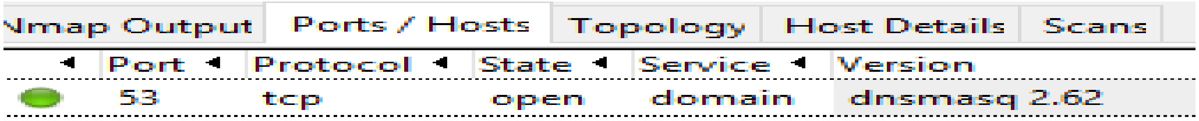
Port scanning doesn’t mean scanning only TCP ports. Nevertheless, UDP port scanning is used more often as well. The basic attacks we observed from the snort log files were TCP, UDP and ICMP port scans. These scans have been performed by different scanner tools such as NMAP.



```
nmap -T4 -A -v 192.168.43.0/24
Nmap scan report for 192.168.43.244 [host down]
Nmap scan report for 192.168.43.245 [host down]
Nmap scan report for 192.168.43.246 [host down]
Nmap scan report for 192.168.43.247 [host down]
Nmap scan report for 192.168.43.248 [host down]
Nmap scan report for 192.168.43.249 [host down]
Nmap scan report for 192.168.43.250 [host down]
Nmap scan report for 192.168.43.251 [host down]
Nmap scan report for 192.168.43.252 [host down]
Nmap scan report for 192.168.43.253 [host down]
Nmap scan report for 192.168.43.254 [host down]
Nmap scan report for 192.168.43.255 [host down]
Initiating Parallel DNS resolution of 1 host. at 22:
Completed Parallel DNS resolution of 1 host. at 22:1
0.01s elapsed
Initiating SYN Stealth Scan at 22:10
Scanning 192.168.43.1 [1000 ports]
Discovered open port 53/tcp on 192.168.43.1
Completed SYN Stealth Scan at 22:11, 6.11s elapsed
(1000 total ports)
Initiating Service scan at 22:11
Scanning 1 service on 192.168.43.1
Completed Service scan at 22:11, 6.18s elapsed (1
service on 1 host)
Initiating OS detection (try #1) against 192.168.43.
Retrying OS detection (try #2) against 192.168.43.1
Retrying OS detection (try #3) against 192.168.43.1
Retrying OS detection (try #4) against 192.168.43.1
Retrying OS detection (try #5) against 192.168.43.1
NSE: Script scanning 192.168.43.1.
Initiating NSE at 22:11
Completed NSE at 22:11, 8.14s elapsed
Initiating NSE at 22:11
Completed NSE at 22:11, 0.00s elapsed
Nmap scan report for 192.168.43.1
```

Figure 4-13 Nmap scanning ports for hosts

In figure 4-13 it has been checking if there are open ports in the whole hosts in our network range 192.168.43.0/24.



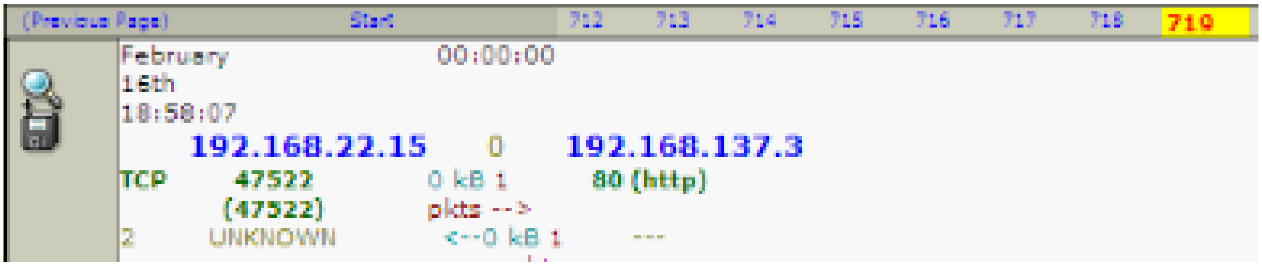
	Port	Protocol	State	Service	Version
●	53	tcp	open	domain	dnsmasq 2.62

Figure 4-14 open port

They have been sending empty UDP datagram's to the target port. If the port is closed, then the attacker will receive "ICMP Port Unreachable" message. If open or filtered, then an error message will be sent back or incoming datagram simply will be ignored.

4-8 Data analysis

Before starting to analyze attackers, the best way is first to study how or where they begin to launch attacks. Most of the experiment results show that they normally start with information gathering about their targets, and then determine what vulnerabilities exist before starting to exploit. As in our experiment, most attacks initially involved collecting the information. They usually use different port and vulnerability scanners, such as Nmap to find open ports and vulnerabilities of their victims, and then start to exploit the existing vulnerabilities. He first probes the port 80, and then tries to get access into the honey pot machine.



```
(Previous Page)        Start        712  713  714  715  716  717  718  719
February               00:00:00
16th
18:58:07
      192.168.22.15       0       192.168.137.3
TCP       47522           0 kB 1        80 (http)
         (47522)          pkts -->
2        UNKNOWN          <--0 kB 1       ---
```

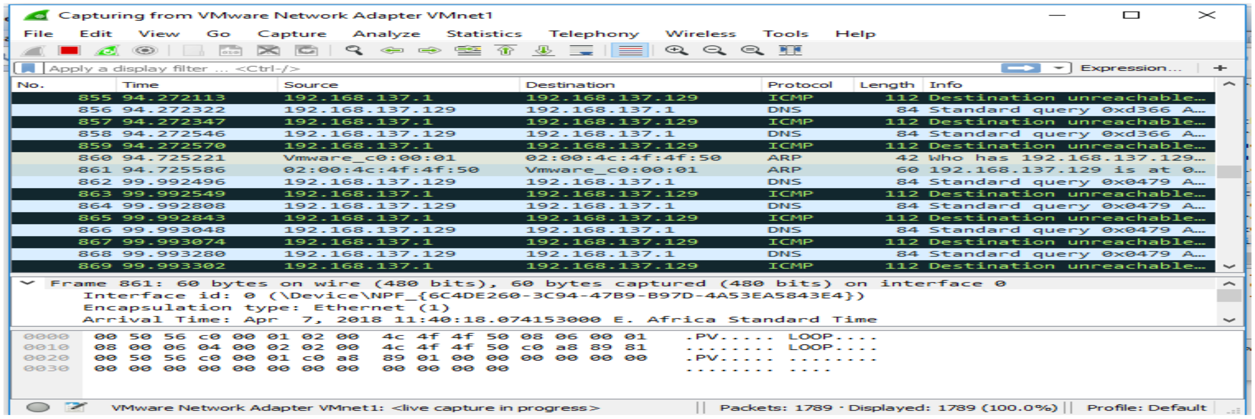
Figure 4-15 packet captured in walleye

In order to find out how the attacks were launched, to determine the motivation and intention of the attackers, to use the knowledge to prevent future occurrences, we decided first to use Wire shark analysis. Our analysis is entirely based on the collected log files from the honey pots as it has been shown in figure 4-15.

4-9 Observed attack cases

In this section we will talk about the attacks against our honey pots and give some examples of them. An important fact is that we could observe malicious connections immediately after launching the honey pots in the network.

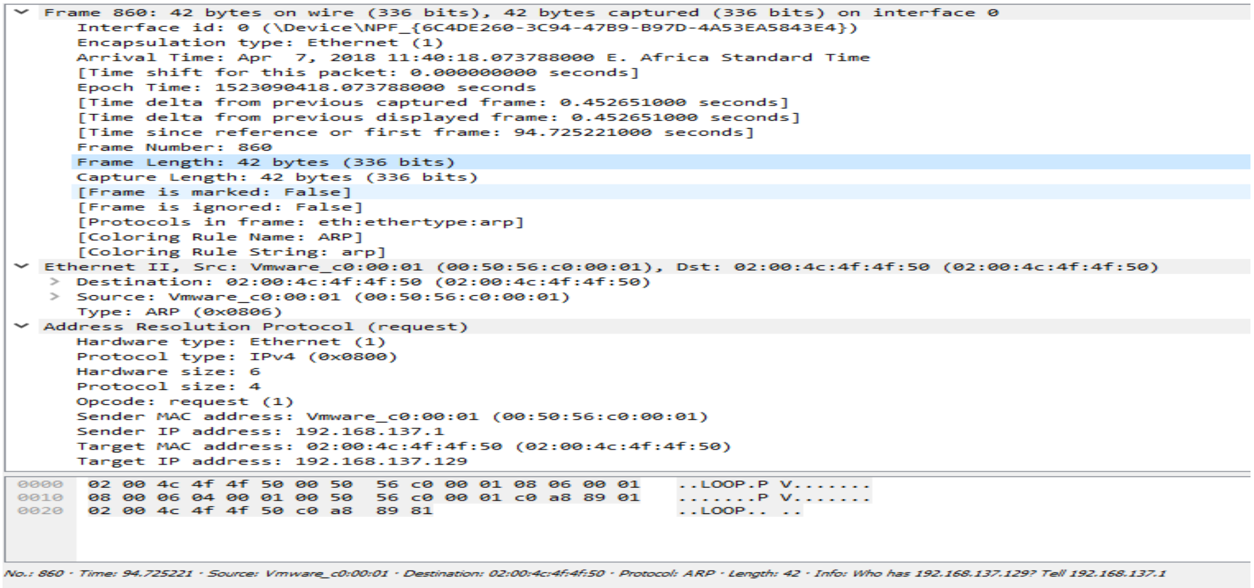
Most common activities which we observed were TCP, UDP and ICMP port scans by intruders in order to check the vulnerabilities on the operating systems.



Capturing from VMware Network Adapter VMnet1
File
Edit
View
Go
Capture
Analyze
Statistics
Telephony
Wireless
Tools
Help
Apply a display filter ... <Ctrl-/>
Expression...

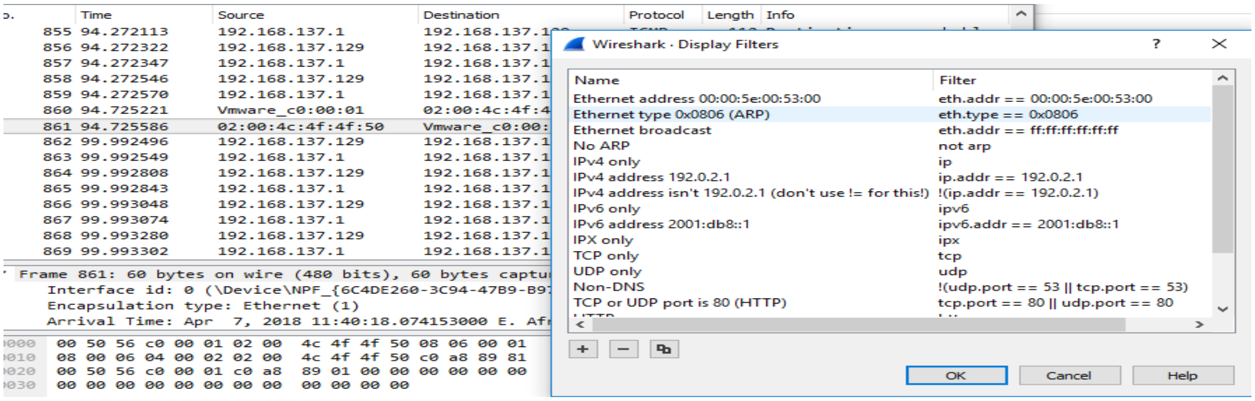
```
No.   Time       Source              Destination          Protocol Length Info
 855 94.272113   192.168.137.1       192.168.137.129      ICMP      112 Destination unreachable…
 856 94.272322   192.168.137.129     192.168.137.1        DNS        84 Standard query 0xd366 A…
 857 94.272347   192.168.137.1       192.168.137.129      ICMP      112 Destination unreachable…
 858 94.272546   192.168.137.129     192.168.137.1        DNS        84 Standard query 0xd366 A…
 859 94.272570   192.168.137.1       192.168.137.129      ICMP      112 Destination unreachable…
 860 94.725221   Vmware_c0:00:01     02:00:4c:4f:4f:50    ARP        42 Who has 192.168.137.129…
 861 94.725586   02:00:4c:4f:4f:50   Vmware_c0:00:01      ARP        60 192.168.137.129 is at 0…
 862 99.992496   192.168.137.129     192.168.137.1        DNS        84 Standard query 0x0479 A…
 863 99.992549   192.168.137.1       192.168.137.129      ICMP      112 Destination unreachable…
 864 99.992808   192.168.137.129     192.168.137.1        DNS        84 Standard query 0x0479 A…
 865 99.992843   192.168.137.1       192.168.137.129      ICMP      112 Destination unreachable…
 866 99.993048   192.168.137.129     192.168.137.1        DNS        84 Standard query 0x0479 A…
 867 99.993074   192.168.137.1       192.168.137.129      ICMP      112 Destination unreachable…
 868 99.993280   192.168.137.129     192.168.137.1        DNS        84 Standard query 0x0479 A…
 869 99.993302   192.168.137.1       192.168.137.129      ICMP      112 Destination unreachable…

Frame 861: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
    Interface id: 0 (\Device\NPF_{6C4DE260-3C94-47B9-B97D-4A53EA5843E4})
    Encapsulation type: Ethernet (1)
    Arrival Time: Apr  7, 2018 11:40:18.074153000 E. Africa Standard Time

0000  00 50 56 c0 00 01 02 00  4c 4f 4f 50 08 06 00 01   .PV..... LOOP....
0010  08 00 06 04 00 02 02 00  4c 4f 4f 50 c0 a8 89 81   ........ LOOP....
0020  00 50 56 c0 00 01 c0 a8  89 01 00 00 00 00 00 00   .PV..... ........
0030  00 00 00 00 00 00 00 00  00 00 00 00               ........ ....

VMware Network Adapter VMnet1: <live capture in progress> || Packets: 1789 · Displayed: 1789 (100.0%) || Profile: Default
```

Figure 4-16 Packet captured in Wire shark



```
Frame 860: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
    Interface id: 0 (\Device\NPF_{6C4DE260-3C94-47B9-B97D-4A53EA5843E4})
    Encapsulation type: Ethernet (1)
    Arrival Time: Apr  7, 2018 11:40:18.073788000 E. Africa Standard Time
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1523090418.073788000 seconds
    [Time delta from previous captured frame: 0.452651000 seconds]
    [Time delta from previous displayed frame: 0.452651000 seconds]
    [Time since reference or first frame: 94.725221000 seconds]
    Frame Number: 860
    Frame Length: 42 bytes (336 bits)
    Capture Length: 42 bytes (336 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:arp]
    [Coloring Rule Name: ARP]
    [Coloring Rule String: arp]
Ethernet II, Src: Vmware_c0:00:01 (00:50:56:c0:00:01), Dst: 02:00:4c:4f:4f:50 (02:00:4c:4f:4f:50)
  > Destination: 02:00:4c:4f:4f:50 (02:00:4c:4f:4f:50)
  > Source: Vmware_c0:00:01 (00:50:56:c0:00:01)
    Type: ARP (0x0806)
Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: Vmware_c0:00:01 (00:50:56:c0:00:01)
    Sender IP address: 192.168.137.1
    Target MAC address: 02:00:4c:4f:4f:50 (02:00:4c:4f:4f:50)
    Target IP address: 192.168.137.129

0000  02 00 4c 4f 4f 50 00 50  56 c0 00 01 08 06 00 01   ..LOOP.P V.......
0010  08 00 06 04 00 01 00 50  56 c0 00 01 c0 a8 89 01   .......P V.......
0020  02 00 4c 4f 4f 50 c0 a8  89 81                     ..LOOP.. ..

No.: 860 · Time: 94.725221 · Source: Vmware_c0:00:01 · Destination: 02:00:4c:4f:4f:50 · Protocol: ARP · Length: 42 · Info: Who has 192.168.137.129? Tell 192.168.137.1
```

Figure 4-17 Packet analysis using Wire shark

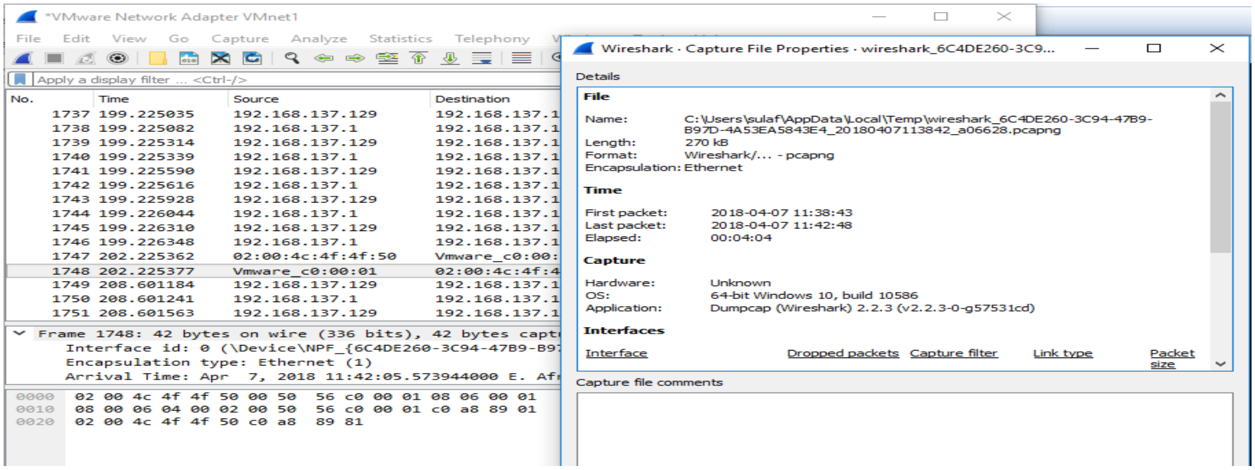


```
o.     Time        Source             Destination        Protocol  Length  Info
  855 94.272113   192.168.137.1      192.168.137.1
  856 94.272322   192.168.137.129    192.168.137.1
  857 94.272347   192.168.137.1      192.168.137.1
  858 94.272546   192.168.137.129    192.168.137.1
  859 94.272570   192.168.137.1      192.168.137.1
  860 94.725221   Vmware_c0:00:01    02:00:4c:4f:4
  861 94.725586   02:00:4c:4f:4f:50  Vmware_c0:00:
  862 99.992496   192.168.137.129    192.168.137.1
  863 99.992549   192.168.137.1      192.168.137.1
  864 99.992808   192.168.137.129    192.168.137.1
  865 99.992843   192.168.137.1      192.168.137.1
  866 99.993048   192.168.137.129    192.168.137.1
  867 99.993074   192.168.137.1      192.168.137.1
  868 99.993280   192.168.137.129    192.168.137.1
  869 99.993302   192.168.137.1      192.168.137.1

Frame 861: 60 bytes on wire (480 bits), 60 bytes captu
    Interface id: 0 (\Device\NPF_{6C4DE260-3C94-47B9-B9
    Encapsulation type: Ethernet (1)
    Arrival Time: Apr  7, 2018 11:40:18.074153000 E. Af

000  00 50 56 c0 00 01 02 00  4c 4f 4f 50 08 06 00 01
010  08 00 06 04 00 02 02 00  4c 4f 4f 50 c0 a8 89 81
020  00 50 56 c0 00 01 c0 a8  89 01 00 00 00 00 00 00
030  00 00 00 00 00 00 00 00  00 00 00 00
```

Wireshark · Display Filters

Name	Filter
Ethernet address 00:00:5e:00:53:00	eth.addr == 00:00:5e:00:53:00
Ethernet type 0x0806 (ARP)	eth.type == 0x0806
Ethernet broadcast	eth.addr == ff:ff:ff:ff:ff:ff
No ARP	not arp
IPv4 only	ip
IPv4 address 192.0.2.1	ip.addr == 192.0.2.1
IPv4 address isn't 192.0.2.1 (don't use != for this!)	!(ip.addr == 192.0.2.1)
IPv6 only	ipv6
IPv6 address 2001:db8::1	ipv6.addr == 2001:db8::1
IPX only	ipx
TCP only	tcp
UDP only	udp
Non-DNS	!(udp.port == 53 || tcp.port == 53)
TCP or UDP port is 80 (HTTP)	tcp.port == 80 || udp.port == 80

Figure 4-18 Display filter using Wire shark



*VMware Network Adapter VMnet1
File
Edit
View
Go
Capture
Analyze
Statistics
Telephony
Apply a display filter ... <Ctrl-/>

```
No.     Time          Source             Destination
   1737 199.225035   192.168.137.129    192.168.137.1
   1738 199.225082   192.168.137.1      192.168.137.1
   1739 199.225314   192.168.137.129    192.168.137.1
   1740 199.225339   192.168.137.1      192.168.137.1
   1741 199.225590   192.168.137.129    192.168.137.1
   1742 199.225616   192.168.137.1      192.168.137.1
   1743 199.225928   192.168.137.129    192.168.137.1
   1744 199.226044   192.168.137.1      192.168.137.1
   1745 199.226310   192.168.137.129    192.168.137.1
   1746 199.226348   192.168.137.1      192.168.137.1
   1747 202.225362   02:00:4c:4f:4f:50  Vmware_c0:00:
   1748 202.225377   Vmware_c0:00:01    02:00:4c:4f:4
   1749 208.601184   192.168.137.129    192.168.137.1
   1750 208.601241   192.168.137.1      192.168.137.1
   1751 208.601563   192.168.137.129    192.168.137.1

Frame 1748: 42 bytes on wire (336 bits), 42 bytes capt
    Interface id: 0 (\Device\NPF_{6C4DE260-3C94-47B9-B9
    Encapsulation type: Ethernet (1)
    Arrival Time: Apr  7, 2018 11:42:05.573944000 E. Af

0000  02 00 4c 4f 4f 50 00 50  56 c0 00 01 08 06 00 01
0010  08 00 06 04 00 02 00 50  56 c0 00 01 c0 a8 89 01
0020  02 00 4c 4f 4f 50 c0 a8  89 81
```

Wireshark · Capture File Properties · wireshark_6C4DE260-3C9...

Details

File

Name: C:\Users\sulaf\AppData\Local\Temp\wireshark_6C4DE260-3C94-47B9-B97D-4A53EA5843E4_20180407113842_a06628.pcapng
Length: 270 kB
Format: Wireshark/... - pcapng
Encapsulation: Ethernet

Time

First packet: 2018-04-07 11:38:43
Last packet: 2018-04-07 11:42:48
Elapsed: 00:04:04

Capture

Hardware: Unknown
OS: 64-bit Windows 10, build 10586
Application: Dumpcap (Wireshark) 2.2.3 (v2.2.3-0-g57531cd)

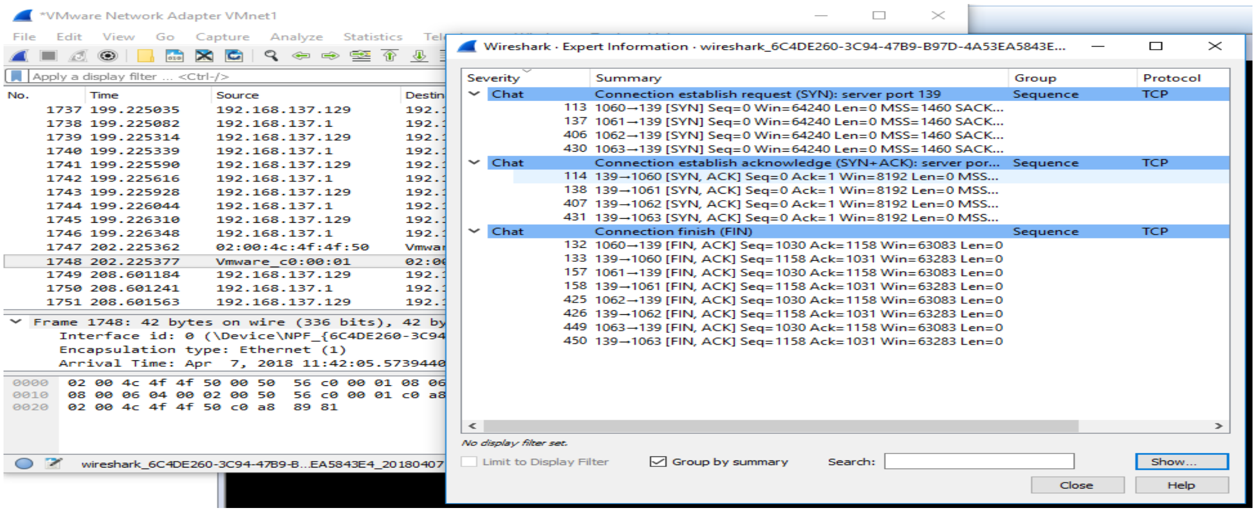
Interfaces

Interface	Dropped packets	Capture filter	Link type	Packet size

Capture file comments

Figure 4-19 Capture file properties using Wire shark

The details information about captured file has been showed in figure 4-19 such as file name, length, format, time, the application used and OS.


*VMware Network Adapter VMnet1
File Edit View Go Capture Analyze Statistics Tel
Apply a display filter … <Ctrl-/>
Wireshark · Expert Information · wireshark_6C4DE260-3C94-47B9-B97D-4A53EA5843E...
No display filter set.
□ Limit to Display Filter
☑ Group by summary
Search:
Show…
Close
Help

```
No.     Time         Source              Destin
   1737 199.225035   192.168.137.129     192.1
   1738 199.225082   192.168.137.1       192.1
   1739 199.225314   192.168.137.129     192.1
   1740 199.225339   192.168.137.1       192.1
   1741 199.225590   192.168.137.129     192.1
   1742 199.225616   192.168.137.1       192.1
   1743 199.225928   192.168.137.129     192.1
   1744 199.226044   192.168.137.1       192.1
   1745 199.226310   192.168.137.129     192.1
   1746 199.226348   192.168.137.1       192.1
   1747 202.225362   02:00:4c:4f:4f:50   Vmwar
   1748 202.225377   Vmware_c0:00:01     02:00
   1749 208.601184   192.168.137.129     192.1
   1750 208.601241   192.168.137.1       192.1
   1751 208.601563   192.168.137.129     192.1

Frame 1748: 42 bytes on wire (336 bits), 42 by
    Interface id: 0 (\Device\NPF_{6C4DE260-3C94
    Encapsulation type: Ethernet (1)
    Arrival Time: Apr  7, 2018 11:42:05.5739440

0000  02 00 4c 4f 4f 50 00 50  56 c0 00 01 08 06
0010  08 00 06 04 00 02 00 50  56 c0 00 01 c0 a8
0020  02 00 4c 4f 4f 50 c0 a8  89 81

wireshark_6C4DE260-3C94-47B9-B...EA5843E4_20180407
```

```
Severity   Summary                                                       Group      Protocol
  Chat     Connection establish request (SYN): server port 139           Sequence   TCP
       113 1060→139 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK...
       137 1061→139 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK...
       406 1062→139 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK...
       430 1063→139 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK...
  Chat     Connection establish acknowledge (SYN+ACK): server por...     Sequence   TCP
       114 139→1060 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS...
       138 139→1061 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS...
       407 139→1062 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS...
       431 139→1063 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS...
  Chat     Connection finish (FIN)                                       Sequence   TCP
       132 1060→139 [FIN, ACK] Seq=1030 Ack=1158 Win=63083 Len=0
       133 139→1060 [FIN, ACK] Seq=1158 Ack=1031 Win=63283 Len=0
       157 1061→139 [FIN, ACK] Seq=1030 Ack=1158 Win=63083 Len=0
       158 139→1061 [FIN, ACK] Seq=1158 Ack=1031 Win=63283 Len=0
       425 1062→139 [FIN, ACK] Seq=1030 Ack=1158 Win=63083 Len=0
       426 139→1062 [FIN, ACK] Seq=1158 Ack=1031 Win=63283 Len=0
       449 1063→139 [FIN, ACK] Seq=1030 Ack=1158 Win=63083 Len=0
       450 139→1063 [FIN, ACK] Seq=1158 Ack=1031 Win=63283 Len=0
```

Figure 4.20 Expert information about connection

4-10 Results and dissection

As a result of comparison this research with other related works [9, 20] that had been mentioned in chapter 2 this research deployed high-interaction honeypots (A virtual honeynet) that interact with real system control, capture and analysis data also collect massive data that are need it, moreover the whole activities from different honeypots were captured by sebek database software, furthermore the captured data were analyzed with weirshark software program.

The concrete result of this thesis can be summarized into the following:

1. Created a platform for the next generation high-interaction honeypots that automates the procedure of capturing zero-day attacks.

2. Developed a technique that transparently enables desktop systems to act as honeypots. Our technique is able to overcome the issues of honeypot avoidance, and can detect the exploits, to protect others systems from attacks.

3. Reduces project hardware costs because virtual Honeynets combine all the elements of a Honeynet onto one physical system.

Finally, we have analyzed attacks targeting our honeynet over a period of 60 days, which made it obviously for us to know the attacked/probed ports and services, attacker IP's (figure 4-15), the OS used in every packet that were captured as it has been showed in figure 4-19, also the Packet length, format, and time (figure 4-19).

Chapter V

Conclusion and Future work

Chapter V Conclusion and Future work

5-1 Conclusion

The honey pots and honey nets are unconventional tools. A Honey net is a network designed to be compromised, not to be used for production traffic. Any traffic entering or leaving the network is suspicious by definition.

Honey pots can be used for research, gathering information on threats therefore it's better to understand and defend against them. The main purpose of this research is to collect data and subsequently analyze it to learn information about attackers and their methods, tools and objectives.

In this thesis, a honey net architecture has been implemented and used for collecting attacker's data and tracking activities carried out by them. Then it has been analyzed. The aim was to study the attackers‟ skill and knowledge based on this analysis. It appears that most of the observed attacks were automated and carried out by script kiddies.

We hope that this work will help organizations to select proper protection mechanism for their networks by evaluating the impact of detected attacks, and taking into consideration the attacker's skill and knowledge level.

5-2 Future work

1- As a future work, we have proposed an improved a honeynet architecture that captured all attackers activities, that can be enhanced to take the data, which we have captured and compare it with database signatures in order to filter out the zero-day attack by then generate signatures for them.

2- In addition, I would like to work on Software to map attackers with respect to targets (using Google API's).

Reverences

Reverences

[1] P. Sokol, P. Pekarčík, and T. Bajtoš, "Data collection and data analysis in honeypots and honeynets," Proceedings of the Security and Protection of Information. University of Defence, 2015.

[2] H. Sallowm, M. Assora, M. Alchaita, and M. Aljnidi, "A Hybrid Honeypot Scheme for Distributed Denial of Service Attack, "2017.

[3] C. Moore and A. Al-Nemrat, "An analysis of honeypot programs and the attack data collected," in *International Conference on Global Security, Safety, and Sustainability*, 2015.

[4] I. Welch, X. Gao, and P. Komisarczuk, "Detecting heap-spray attacks in drive-by downloads: Giving attackers a hand," in Local Computer Networks (LCN), 2013 IEEE 38th Conference on, 2013.

[5] K. Cabaj and P. Gawkowski, "HoneyPot systems in practice," Przegląd Elektrotechniczny, vol. 91, pp. 63-67, 2015.

[6] M. M. Mohammed, "Automated signature generation for zero-day polymorphic worms using a double-honeynet," University of Cape Town, 2012.

[7] E. Alata, M. Dacier, Y. Deswarte, M. Kaaâniche, K. Kortchinsky, V. Nicomette, et al., "Collection and analysis of attack data based on honeypots deployed on the Internet," in Quality of Protection, ed: Springer, 2006.

[8] L. Spitzner, *Honey pots: tracking hackers* vol. 1: Addison-Wesley Reading, 2003.

[9] F. H. Abbasi and R. Harris, "Experiences with a generation iii virtual honeynet," in Telecommunication Networks and Applications Conference (ATNAC), 2009 Australasian, 2009.

[10] M. Mohammed and A.-S. K. Pathan, Automatic defense against zero-day polymorphic worms in communication networks: Auerbach Publications, 2016.

[11] R. Karthikeyan, D. T. Geetha, K. Shyamamol, and G. Sivagami, "Advanced Honey Pot Architecture for Network Threats Quantification," the international journal of Engineering and Techniques, vol. 3, pp. 2395-1303, 2017.

[12] M. V. Pawar and J. Anuradha, "Network security and types of attacks in network," Procedia Computer Science, vol. 48, pp. 503-506, 2015.

[13] R. R. Patel, C. S. Thaker, and H. B. Patel, "Detecting Zero-Day Attack Signatures using Honeycomb in a Virtualized Network," 2011.

[14] L. Ablon and A. Bogart, Zero Days, Thousands of Nights: The Life and Times of Zero-Day Vulnerabilities and Their Exploits: Rand Corporation, 2017.

[15] http://elearningatria.files.wordpress.com/2013/10/ise-viii-information-and-network-security-06is835-notes.pdf, time: 7/3/2018 5:30:00 pm.

[16] A. Singh and M. Gahlawat, "Internet Protocol Security (IPSec)," International Journal of Computer Networks and Wireless Communications, vol. 2, pp. 717-721, 2012.

[17] Y. M. Mali, M. Raj, and A. T. Gaykar, "Honeypot: a tool to track hackers," IRACST-Engineering Science and Technology: An International Journal (ESTIJ), vol. 4, 2014.

[18] K. Gary and G. Diane, "Analysis of Attacks Using a Honeypot, "2014.

[19] V. Aliyev, "Using honeypots to study skill level of attackers based on the exploited vulnerabilities in the network Department," Thesis memory of Computer Science and Engineering Division of Computer Security Chalmers University of technology Göteborg, Sweden, 2010.

[20] T. Verwoerd and R. Hunt, "Intrusion detection techniques and approaches," Computer communications, vol. 25, pp. 1356-1365, 2002.

[21] C. S. Date, "ReSIST: Resilience for Survivability in IST," Interaction, p. 1, 2008.

[22] T. Kouba, "Virtual honeynet with simulated user activity," 2009.

[23] A. J. Bennieston, "Nmap-a stealth port scanner," ed, 2004.

[24] C. Carthern, W. Wilson, R. Bedwell, and N. Rivera, "Introduction to Network Penetration Testing," in Cisco Networks, ed: Springer, 2015.

[25] L. Chappell and G. Combs, Wireshark network analysis: the official Wireshark certified network analyst study guide: Protocol Analysis Institute, Chappell University, 2010.

Appendix A:

Sample of ASCII text file

# /usr/local/bin/hwctl -s -p /etc/honeywall.conf #

```
#$Id: honeywall.conf 4552 2006-10-17 01:06:51Z esammons $


 #############################################
 #
 # Copyright (C) <2005><The Honeynet Project>
 #
 # This program is free software; you can redistribute it and/or modify
 # it under the terms of the GNU General Public License as published by
 # the Free Software Foundation; either version 2 of the License, or (at
 # your option) any later version.
 #
 # This program is distributed in the hope that it will be useful, but
 # WITHOUT ANY WARRANTY; without even the implied warranty of
 # MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
 # General Public License for more details.
 #
 # You should have received a copy of the GNU General Public License
 # along with this program; if not, write to the Free Software
 # Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
 # USA
 #
 #############################################

 #
 # This file is the Honeywall import file (aka "honeywall.conf").
 # It is a list of VARIABLE=VALUE tuples (including comments as
 # necessary, # such as this) and whitespace lines.
 #
 # note: DO NOT surround values in quotation marks
 #
 #####################################################################

 ############################
 # Site variables that are  #
 # global to all honeywalls #
 # at a site.               #
 ############################

 # Specify the IP address(es) and/or networks that are allowed to connect
 # to the management interface.  Specify any to allow unrestricted
access.
 # [Valid argument: IP address(es) | IP network(s) in CIDR notation |
any]
HwMANAGER=any

 # Specify the port on which SSHD will listen
 # NOTE: Automatically aded to the list of TCP ports allowed in by
IPTables
 # [Valid argument: TCP (port 0 - 65535)]
HwSSHD_PORT=22

 # Specify whether or not root can login remotely over SSH
 # [Valid argument: yes | no]
HwSSHD_REMOTE_ROOT_LOGIN=yes

 # NTP Time server(s)
 # [Valid argument: IP address]
HwTIME_SVR=


 ############################
 # Local variables that are #
 # specific to each         #
 # honeywall at a site.     #
 ############################

 # Specify the system hostname
 # [Valid argument: string ]
HwHOSTNAME=winos

 # Specify the system DNS domain
 # [Valid argument: string ]
HwDOMAIN=localhost

 #Start the Honeywall on boot
 # [Valid argument: yes | no]
HwHONEYWALL_RUN=yes

 # To use a headless system.
 # [Valid argument: yes | no]
HwHEADLESS=no


 # This Honeywall's public IP address(es)
 # [Valid argument: IP address | space delimited IP addresses]
HwHPOT_PUBLIC_IP=192.168.137.3 192.168.137.4
```

Appendix B:

Sample of Snort alert

```
[**] [1:466:5] ICMP L3retriever Ping [**]
 [Classification: Attempted Information Leak] [Priority: 2]
11/11-21:17:09.097950 192.168.137.3 -> 192.168.137.1
 ICMP TTL:32 TOS:0x0 ID:1724 IpLen:20 DgmLen:60
 Type:8  Code:0  ID:512   Seq:14080  ECHO
 [Xref => http://www.whitehats.com/info/IDS311]

 [**] [1:466:5] ICMP L3retriever Ping [**]
 [Classification: Attempted Information Leak] [Priority: 2]
11/11-21:29:34.761638 192.168.137.3 -> 192.168.137.1
 ICMP TTL:32 TOS:0x0 ID:1744 IpLen:20 DgmLen:60
```

```
Type:8  Code:0  ID:512   Seq:14336  ECHO
[Xref => http://www.whitehats.com/info/IDS311]

[**] [1:538:15] NETBIOS SMB IPC$ unicode share access [**]
[Classification: Generic Protocol Command Decode] [Priority: 3]
11/11-21:44:53.073881 192.168.137.1:13659 -> 192.168.137.3:139
TCP TTL:128 TOS:0x0 ID:14831 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0x826C07BC  Ack: 0x705FFD8E  Win: 0x802  TcpLen: 20

[**] [1:538:15] NETBIOS SMB IPC$ unicode share access [**]
[Classification: Generic Protocol Command Decode] [Priority: 3]
11/11-21:56:54.214850 192.168.137.1:13668 -> 192.168.137.3:139
TCP TTL:128 TOS:0x0 ID:14857 IpLen:20 DgmLen:140 DF
***AP*** Seq: 0xC46DBF3F  Ack: 0x95242184  Win: 0x802  TcpLen: 20

[**] [1:538:15] NETBIOS SMB IPC$ unicode share access [**]
[Classification: Generic Protocol Command Decode] [Priority: 3]
11/11-22:08:55.305331 192.168.137.1:13672 -> 192.168.137.3:139
TCP TTL: 128 TOS: 0x0 ID: 14873 IpLen: 20 DgmLen: 140 DF
***AP*** Seq: 0xA6EB2823 Ack: 0xD1346A22 Win: 0x802 TcpLen: 20

[**] [1:538:15] NETBIOS SMB IPC$ Unicode share access [**]
[Classification: Generic Protocol Command Decode] [Priority: 3]
11/11-22:20:56.399252 192.168.137.1:13688 -> 192.168.137.3:139
TCP TTL: 128 TOS: 0x0 ID: 14888 IpLen: 20 DgmLen: 140 DF
***AP*** Seq: 0xA09A5A70 Ack: 0x53FBE24B Win: 0x802 TcpLen: 20

[**] [1:538:15] NETBIOS SMB IPC$ Unicode share access [**]
[Classification: Generic Protocol Command Decode] [Priority: 3]
11/11-22:31:28.115089 192.168.137.1:13714 -> 192.168.137.3:139
TCP TTL: 128 TOS: 0x0 ID: 14937 IpLen: 20 DgmLen: 140 DF
***AP*** Seq: 0xD372AB51 Ack: 0xFE0AC07B Win: 0x802 TcpLen: 20.
```