A Network Intrusion Testbed through Honeypots

Luisse Margarette A. Macasaet and Joseph Anthony C. Hermocilla

***Abstract*— The field of honeypots is fast evolving and researchers are trying to find more innovations for this technology due to its behavioral analysis capabilities of network intrusions which complements the traditional signature-based detection methods. This paper presents the effectiveness of Honeyd, a low-interaction honeypot, when used as a deceptive tool to lure attackers into thinking that they have found a vulnerable segment in the network and their actions are far from being monitored.**

I. Introduction

Malicious hacking has been a problem since the time when wardialing and phone phreaking were all the hype. Back then, very little was known about these intruders, let alone tools to detect and prevent their attacks. But at this present age of script kiddies and blackhats, we heavily rely on access control tools and other security policies to prevent unwanted access to our private data [1].

A. Background of the Study

Network intrusion detection is concerned with the detection of attacks made against a network that are meant to compromise and exploit the confidentiality, integrity and availability of a resource [2]. The main concern of network intrusion detection, however, is to identify malicious network activities and differentiate them from normal network activities.

Throughout the years, many technologies and tools have been used to create and test systems that perform automated intrusion detection, or simply intrusion detection systems (IDS). One of these is the honeypot. A honeypot is a trap which usually consists of data, a computer or a network site. It appears to be a normal part of a network or more often than not, an important part of a network that contains data valuable to hackers; but in reality, it is actually isolated and monitored for malicious activity [3]. More precisely, a honeypot is an information system whose value lies in unauthorized or illicit use of that resource [4].

B. Statement of the Problem

Since hackers and virus writers have come up with better ways to evade anti-virus technology throughout the years, the use of signature-based anti-virus software is proving to be less effective in putting a stop to malicious codes running in our computers. There is a need to find a way to analyze malicious activity without having to rely on the traditional signature-based anti-virus tools but instead, complement what these tools can already do.

Presented to the Faculty of the Institute of Computer Science, University of the Philippines Los Baños in partial fulfillment of the requirements for the Degree of Bachelor of Science in Computer Science

C. Significance of the Study

It is necessary to use a honeypot instead of a firewall because the detection algorithm that Network Intrusion Detection Systems (NIDS) use is based on how signature-based anti-virus tools detect malicious activities. They both rely on a database of attacks that have already been detected and recorded. This leaves NIDS unaware of newly-developed compromises that are unknown to it at the time of the attack. A honeypot, however, can detect vulnerabilities that are not yet identified.

There is a need to create a testbed that will totally involve a network under study and at the same time, prevent intrusions from exposing and exploiting the vulnerabilities of the said network. Since honeypots are deceptive systems, it will be very useful in hiding the real value of the data that pass over the network.

This study will make use of a honeypot in the creation of a testbed that will help in testing and identifying vulnerabilities of a network. It may also help in the analysis of attacks that are both known and unknown to the public.

This will be different from previous of testbeds of the same nature in a way that the honeypot will not simply be a part of a network that is connected to the internet and will wait for attacks or malicious activity but it will also be attacked deliberately by the tester.

D. Objectives of the Study

The main aim of this study is to implement a network intrusion testbed that uses honeypots. Specifically, this study aims to:

1. Create an interface that will make the configuration of a honeypot easier;
2. Use a honeypot configuration that will be tested over a small network; and
3. Analyze the activity that will be logged by the honeypot

II. Review of Related Literature

In 2001, when Code Red [5] was detected on the internet, Liston had the idea of a sticky honeypot [6], [7]. Thus, the LaBrea Tarpit was born. The LaBrea Tarpit uses unused IP addresses and creates virtual servers on them. These virtual servers respond to connection attempts that are made by attackers. This action delays the attackers until they get stuck for a period of time. This is why it is called a tar pit.

Network-based detection refers to methods used to help detect malicious entities by studying network traffic [8]. Szor [9] proposed to update and maintain a list of hosts or network segments that are allowed to access the resources of a network. Through this, packets from intruders are simply not allowed

to enter the network. Although this is an effective method, address spoofing may be used to imitate or use the address of a host that has access to the network.

Since honeypots have become a specific scientific research field only not too long ago, there have not been a lot of published papers on the use of it. However, in 2008, Jostein Jensen of SINTEF ICT, made a study of a new approach – that is the use of honeypots in detection of malicious software functionality.

The study made use of five computers connected to a network. One computer served as a firewall. Another computer was where the honeypot was installed. This computer logged all the traffic that was able to go through the firewall. Another computer served as a packet sniffer and another one acted as a remote vulnerability scanner. The last one was where the malicious software was installed.

Jensen tested different kinds of malicious software by downloading and installing them into the last computer. The malicious activities from these programs were detected and were marked as malicious. Almost every malicious activity was detected except those that came from rootkits, a kind of software which is a collection of programs that enables a continued administrator-level access to a computer by corrupting operating system (OS) functionalities and/or other applications while hiding its presence.

An earlier study by Jonathan Werrett [10] back in 2003 implemented and tested a network intrusion honeypot. His goal was to test and explore different configurations of a honeypot to see how the honeypot would behave when deployed over a large network, particularly on the network of University of Western Australias Department of Computer Science.

The project was tested on a small local area network (LAN). Honeyd, a honeypot software was used. A series of attacks were launched against the network to see if Honeyd and Snort, a packet logger, would correctly identify the kinds of attacks. Even though this study is quite old, it shows the most basic functionality of a honeypot when used as an intrusion detection system. It is perfect for testing a honeypot configuration before deploying it over a larger network.

Some other researchers found Honeyd and the concept of honeypots useful to the study of network intrusions. For the implementation of their testbed, J. Awad and D. Andreas [11], in 2005, used the combination of Honeyd, a low-interaction honeypot and Honeynet(of the Honeynet Project [12]), a high-interaction honeypot. They wanted to see how effective the combination was of the two, not only in intrusion detection but also in intrusion prevention.

In 1999, Roesch [13] created SNORT, a ”simple, lightweight, and open-source” NIDS for small networks. It performs packet decoding and serves as a detection engine. Once a malicious activity is detected, it alerts the network's subsystems. SNORT is considered one of the best intrusion detection systems of today.

III. Theoretical Framework

The project will make use of a honeypot mainly because it collects data on attacks launched against the network. It is also capable of recording the hacker's activity but at the same time, it conceals the presence of a firewall [1]. It also has the capability to provide a good estimate of the generated traffic of the bots of a botnet [14], thus making it more effective than an ordinary NIDS.

Honeypots may either be used for production or research. For production purposes, it will help in the protection of the system it is connected to through detection, prevention and response to attacks. On the other hand, if it is used for research purposes, it will be dedicated to information collection. Since honeypots are capable of deception techniques [15], they are very effective for these two purposes.

Honeypots have two kinds: low-interaction and high-interaction. Low-interaction honeypots only simulate parts and services of a system while high-interaction honeypots have real systems that the attacker can even interact with [4]. For this study, a low-interaction detective honeypot will be used. Detective honeypots are used to detect unauthorized activity without having to worry about false alerts. [11] The specific honeypot that will be used is N. Provos's Honeyd.

IV. Methodology

Based on the previous studies and works on intrusion detection, this study created a testbed that uses a low-interaction honeypot, specifically Honeyd. Honeyd's functionalities were integrated with those of *Arpd's* (Address Resolution Protocol (ARP) response spoofer, among many others), *Wireshark's* (packet sniffer), and *Nmap's* (network vulnerability scanner) and were applied to an emulated network topology by Honeyd. On top of these, a graphical user interface was made to make it easier for the user to configure and test different network configurations. A log report functionality was also added to the application.

A. Hardware Specifications

The specifications of the actual hardware used are the ff:

1. Attacker
Intel Core i5-2410M
4 GB RAM
Network Interface Card

2. Honeypot
Intel Pentium Dual Core
1 GB RAM
Network Interface Card

B. Software Specifications

The specifications of the actual software used are the ff:

1. Attacker
Backtrack 5 OS
Nmap
NESSUS

2. Honeypot

Ubuntu 10.10
Honeyd 1.5c
ARPD
Wireshark

C. Tools

a) Honeyd: Honeyd is a low-interaction honeypot that allows the emulation of different network topologies and the monitoring of unused IP addresses. It logs and detects unauthorized User Datagram Protocol (UDP), Transmission Control Protocol (TCP) and some Common Management Information Protocol(CMIP) activities. One advantage of Honeyd over other low-interaction honeypots is it can emulate operating systems(OSs) at the kernel level. It can even emulate vulnerabilities. Two more interesting features include specifying the latency of incoming and outgoing packets from a network and specifying packet loss percentage.

b) Arpd: Arpd is a tool that spoofs ARP responses [16]. It listens to ARP requests and causes Honeyd to respond to pings to unallocated IP addresses [11].

c) Wireshark: Wireshark is a network protocol analyzer. It offers the capability of capturing, browsing, and analyzing the traffic running on a computer network [17].

d) Nmap: Nmap is a tool that analyzes IP packets to determine what hosts are available on the network, what services these hosts offer, what operating systems they are running, which firewalls are enabled, and many other characteristics [18].

e) NESSUS: NESSUS is a product of Tenable, one of the world's top network security companies. It is a network vulnerability scanner [19]. Its goal is to detect potential vulnerabilities on systems. Examples of such vulnerabilities are security holes that allow crackers to have access to sensitive data, denials of service against the TCP/IP stack, access to system accounts, and others of the like.

D. Procedure

a) Network Set-up: There were two possible set-ups for the network. One was to have it online in which the attacker could launch attacks through the internet. The second one was to have it remain offline and where it could be attacked locally. In the online setup, at least one computer was required. It did not necessarily need to be attacked because unwanted connections may already be detected from the internet. On the offline setup, however, the basic requirements of the network were at least two computers. One computer was the attacker and the other was just an ordinary host.

The offline setup is shown in Fig. 1 and the online setup in Fig. 2.

b) Honeypot: Arpd was used as a redirection tool. All attacks that are originally directed to other segments or hosts of the network were redirected to the honeypot computer. This was done to make sure that all activities would pass through the honeypot and were being logged.

The command used was "farpd 192.160.0.0/12"

Honeyd was installed on a computer and computers to be emulated were created. For this study, an emulated network

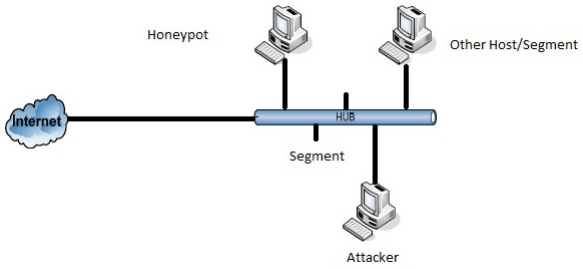

Fig. 1. The Attacker is Part of the Network

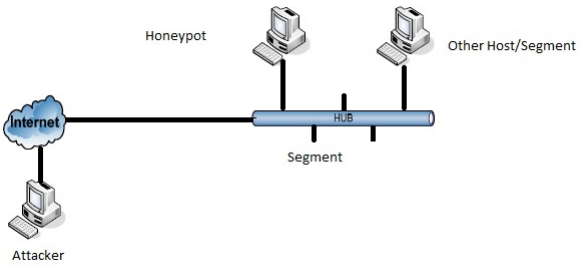

Fig. 2. The Attacker is Outside the Network

of six computers connected through a Cisco 7206 (running IOS 11.1(24)) router was created. Two of the computers had Windows XP Professional SP1 as their OS, another two had Windows 2000 Professional and the last two had NetBSD 1.6. This is shown in Fig. 3. The IP addresses of these computers were automatically bound through Arpd, letting them use the unused IP addresses in the network. Through Honeyd, the open and blocked ports were specified. The contents of the configuration file were a default personality whose ports through all protocols were all open, the Cisco router, and the six hosts with random selected ports opened. The first 6 IP addresses of the network were also bound to the six emulated hosts. One Windows 2000 host was equipped with an emulation of the Kuang2 virus, a virus that infects the executables of a Windows host and allows the set-up of a server that allows remote control of a computer [20]. One Windows XP host was also equipped with an emulation of the Mydoom worm, a worm that sends emails through infected computers and infected hosts targeted a specific URL to flood with traffic [21].

Wireshark served as the packet sniffer. It was used to see the traffic on the network and to confirm that Honeyd logs the traffic.

c) Attacker: The attacker made use of ping requests, Nmap scans, NESSUS scans, and the built-in penetration testing applications of the Backtrack 5 OS. The attacker made use of *Nmap* to determine what OS the the host was running and *NESSUS* to check the vulnerabilities of the host desired to be attacked. The honeypot must be able to detect and log the intrusions. A flowchart is presented in fig. 4.

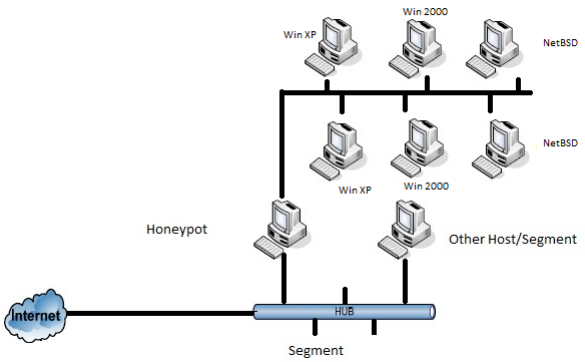

Fig. 3. The Virtual Network of the Honeypot Connected to the Physical Network

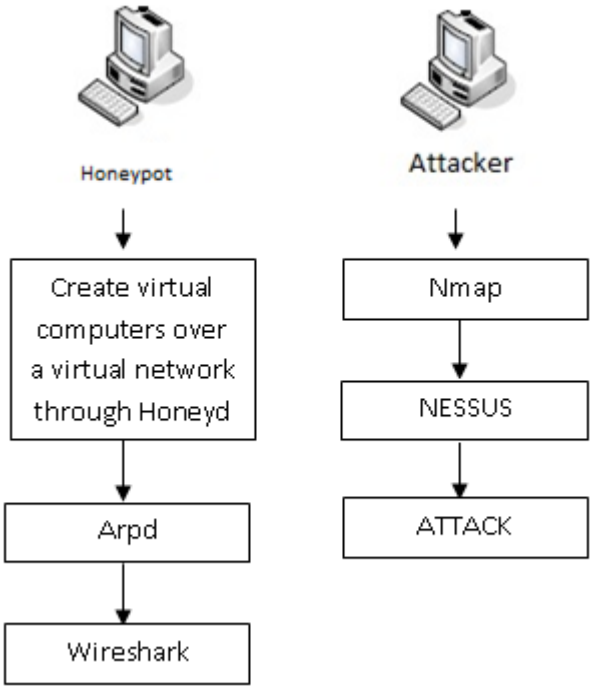

Fig. 4. Flowchart for both Attacker and Honeypot

V. Results and Discussion

A. Graphical User Interface

Honeyd. A GUI was made for Honeyd, which is originally only accessible through a Command Line Interface (CLI). The GUI made it possible for a user to graphically configure the network to be emulated. Features of the Honeyd GUI are shown through the following figures:

This main window allows the user to choose what task to do.

When the user selects ”Start” from the main window, this form will appear. It allows the user to start the Honeyd daemon and specify the options in which it would run. For the options used in the study, the equivalent command is ”honeyd -d -f /root/Desktop/honeypot/honeyd.conf -l /root/Desktop/honeypot/honeyd.log -p /root/Desktop/honeypot/nmap.prints -i eth0 192.160.0.0/12”.

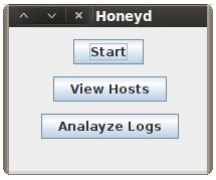

Fig. 5. Main Window

This configuration tells the daemon that it is not daemonized, the configuration file to be used is at /root/Desktop/honeypot/honeyd.conf, the log file to be used is at /root/Desktop/honeypot/honeyd.log, to use the contents of /root/Desktop/honeypot/nmap.prints to help respond to OS fingerprinting requests through the command *nmap* and to use the interface eth0 and to bind the daemon to the network 192.160.0.0 and its hosts.

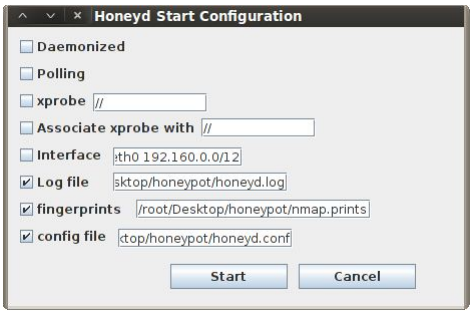

Fig. 6. Start Honeyd Daemon Window

When the user selects ”View Hosts” from the main window, a list of the hosts will appear. Shown here are the hosts already created for the network.

Fig. 7. View Host Properties Window

The look of the view properties window is shown in fig. 8.

Fig. 8. View Hosts Window

When the user selects ”Add Host” from the View Hosts Window, a user will be able to add hosts and specify the

characteristics of the host(s) to be added. The look of the add a host window is shown in fig. 9.

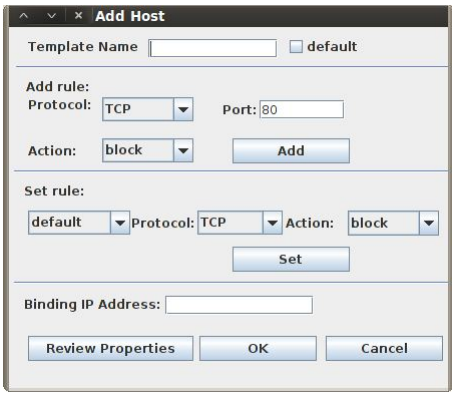

Fig. 9. Add Host Window

To delete a host, one can simply select the host and then click on delete. The design of the user interface is inspired by HoneydGUI [22], an open-source GUI for the Honeyd 1.3.

B. Testing of honeypot configuration

Ping response All the six hosts replied to ping requests. The remaining unused IP addresses of the network also responded to ping requests as if they really do exist.

Nmap The Nmap of the two NetBSD hosts returned a result of Linux 1.6. The Windows hosts, however, returned results of uncertain operating systems. However, the presented list of possible operating systems that the hosts might be running are all Windows operating systems.

NESSUS NESSUS was able to see the vulnerabilities of the hosts, especially of the two Windows hosts which were infected by the Kuang2 Virus and the Mydoom worm. These were classified under ”backdoor.” The open ports were also detected by NESSUS.

C. Activity Analysis

Logging and Wireshark All the incoming and outgoing packets were logged.

Fig. 10 shows the logging that was done through Wireshark. During this time, an Nmap query was being done to the IP address 192.168.0.4 which supposedly belongs to a Windows 2000 machine.

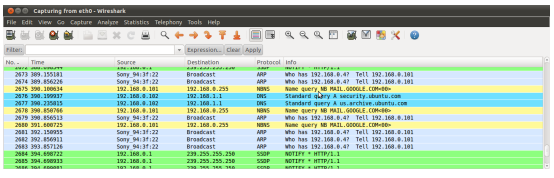

Fig. 10. Wireshark Logs

Fig. 11 shows the logging that was done in the honeyd.log file.

Log Analysis The log analysis feature gave the correct statistics for the incoming and outgoing packets that were logged.

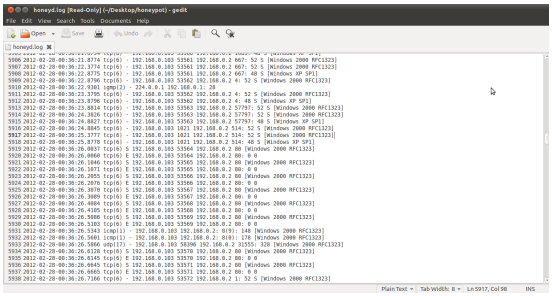

Fig. 11. Honeyd log file

Fig. 12 shows how to see reports on the logs that the daemon was saved.

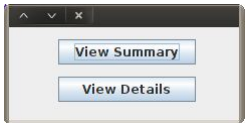

Fig. 12. Analyze Logs Window

The summary of logs will simply show the top IP addresses that initiated a connection, and the number of connections initiated. The detailed list, on the other hand, shows all ip addresses that initiated a connection, the resource accessed and how many times it was accessed. Fig. 13 shows an example of a log summary.

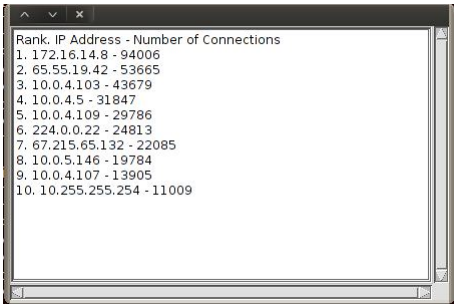

Fig. 13. Analyze Logs Window

VI. Conclusion

The GUI was able to aid the user in using the Honeyd daemon. Aside from creating a GUI for its basic features, it also added the capability to modify the properties of a host while on the process of being added. The configuration was able to log all non-safe activities in the network and this is a big start for behavioral analysis of network activities. Based on the accuracy shown by Honeyd in the experiments done, these deception systems are very powerful when it comes to intrusion detection and anonymous data collection of a hacker’s movements. Honeyd was successful in aiding the detection of intrusions. It did not only deceive the intruder into believing that there really were existing hosts but it also logged the intruder’s movements inside the network.

REFERENCES

[1] N. Rowe, H. Goh, S. Lim, and B. Duong, "Experiments with a testbed for automated defensive deception planning for cyber-attacks," *2nd International Conference in I-Warfare and Security*, March 2007.

[2] Intrusion detection. [Online]. Available: http://en.wikipedia.org/wiki/Intrusion_Detection/

[3] Honeypot. [Online]. Available: http://en.wikipedia.org/wiki/Honeypot_(computing)

[4] N. Provos and T. Holz, *Virtual Honeypots: From Botnet Tracking to Intrusion Detection*. Reading, MA: Addison-Wesley, 2007.

[5] Code red (computer worm). [Online]. Available: http://en.wikipedia.org/wiki/Honeypot_(computing)

[6] H. Goh, "Intrusion deception in defense of computer systems," Master's thesis, Naval Postgraduate School, Monterey CA, March 2007.

[7] L. Haig, "Labrea a new approach to securing our networks," 2003.

[8] J. Jensen, "A novel testbed for detection of malicious software functionality," *The Third International Conference on Availability, Reliability and Security*.

[9] P. Szor, *The Art of Computer Virus Research and Defence*. Addison-Wesley, 2005.

[10] J. Werrett, "Implementing and testing an intrusion detection honeypot," Master's thesis, University of Western Australia, Perth, Australia, June 2003.

[11] A. Johnny and D. Andreas, "Implementation of a high interaction honeypot testbed for educational and research purposes," Master's thesis, Athens Information Technology, Athens, Greece, 2005.

[12] The honeynet project. [Online]. Available: http://www.honeynet.org/

[13] M. Roesch, "Snort - lightweight intrusion detection for networks," *13th USENIX Conference on System Administration*, 1999.

[14] A. Aviv and A. Haeberlen, "Challenges in experimenting with botnet detection systems," *4th Workshop on Cyber Security Experimentation and Test*, August.

[15] K. Charles, "Decoy systems:a new player in network security and computer incident response," *International Journal of Digital Evidence*, vol. 2, 2004.

[16] Honeyd tools. [Online]. Available: http://www.honeyd.org/tools.php/

[17] Wireshark. about. [Online]. Available: http://www.wireshark.org/about.html/

[18] Nmap - free security scanner for network exploration and security audits. [Online]. Available: http://www.nmap.org/

[19] Tenable nessus — tennable network security. [Online]. Available: http://www.tenable.com/products/nessus/

[20] Backdoor trojan and virus: Kuang2 the virus. [Online]. Available: http://www.securityspace.com/smysecure/catid.html?id=10132

[21] Mydoom. [Online]. Available: http://en.wikipedia.org/wiki/Mydoom

[22] Honeydgui project. [Online]. Available: http://download.openpkg.org/components/cache/honeyd/honeydGUI.tar.gz