HONEYPOT DRIVEN ACCESS CONTROL RULES: A PROOF OF CONCEPT

Ryan Cortis
Intitute of Information and Communication Technology
MCAST
Paola, Malta
ryan.cortis.b425967@mcast.edu.mt

***Abstract*—A honeypot is a security tool which its value lies on illicit or unauthorised use of the resource in order to collect intelligence on attack techniques and behaviours. While honeypots are mostly an intrusion detection system and don't solve a specific problem, they are a highly flexible tool and have enormous potential for security use in enterprises. This research is based upon their effectiveness as a security solution in enterprises as well as the complexity of implementing such systems.**

***Index Terms*—Honeypot, security, network, hacking**

I. Introduction

In the era of information and technology, network security has become the main issue in every enterprises network. This has lead to enterprises introducing honeypot systems to their network. A security system designed to mimic production systems, with the intention of being compromised and its sole purpose being to detect and log attacks or deflect them from legitimate targets.

A. Hypothesis

It is hypothesised that honeypots are an effective security tool. As they are highly monitored computer systems designed to deceive attackers and are used to learn, how and what types of attacks where attempted on the system. They also waste an attacker's time by having to go around the honeypot instead of going after systems which have real and sensitive data, ultimately deterring attackers and giving system administrators time to act and put in place necessary measurements to counteract any future security breaches.

Honeypots are also tricky to implement because they are designed to mimic a production system as much as possible therefore implementing a honeypot will be very time consuming to both implement and maintain.

B. Research Questions

- Are honeypots an effective security solution for enterprises?
- Are honeypots tricky to implement?

C. Aims

This research aims to determine what level of security a honeypot offers enterprises, furthermore it also intends to gauge the level of complexity needed to properly implement such honeypots

D. Objectives

An objective of this research is determine the level of complexity needed to create a honeypot through the calculation of the total time to implementation, and through an assessment of any issues which pop up during the process. Another objective of this research is to determine the security effectiveness of honeypots. This will be assessed through the application of several tests.

E. Motivation

The motivation for this project is it to implement a honeypot in order to better learn and understand how honeypots are implemented, function and whether they are an effective security measure for enterprises. Thus, providing a bases of knowledge which can be used to create and manage future honeypot systems. Also, if certain problems are found during the implementation and testing stages these will be tackled, providing the knowledge to more easily tackle issues of a similar nature if found in future honeypot implementations.

F. Relevance of Research

The results of the research carried out on honeypots are important as they will help provide insightful information, regarding whether honeypots should be used as a security solution in an enterprise and if undergoing the implementation of a honeypot is worth while for the enterprise. Knowing this, the enterprise will be able to better determine if implementing a honeypot in their network is feasible.

II. Literature Review

Many recent studies have focused on giving a general overview on the different types of honeypot implementations that exist, legal issues with honeypots, their advantages and disadvantages. Very few have delved deep into the implementation process of honeypots and the effectiveness of such systems in enterprises. In this literature review we will be going through these past works. Which will serve as a basis of knowledge onto which the project will be built upon.

First of all, a honeypot is a security tool setup on a computer system which is used to mimic a production system and used to detect attacks and log them or deflect them from a legitimate target. Its value lies in the unauthorised use of the

system and any interaction with the honeypot is considered to be malicious intent. While honeypots do not solve a security problem, they can still be helpful as information captured from the honeypot can be used to help a system administrator enhance the security of their systems and networks.

Various configurations of honeypots exist but these can be classified into two main types: Production and Research.

A. Production Honeypots

Production honeypots are typically low interaction honeypots which have little to no interaction with attackers and mostly serve as an early warning system. These types of honeypots are of little purpose, apart from capturing data. Which in essence makes them a basic event log system that can't be interacted with but are easy to deploy and maintain across different segments of the network. (Tsikerdekis et al. 2018)

Akkaya & Thalgott (2010) performed a study using a low interaction honeypots called Honeyd. Honeyd is a an open source production honeypot that is designed for Unix systems. Due to it being a low interaction honeypot it doesn't have an operating system but new services, opening and listening of ports can be configured. While the hacker will not be able to find any real computer with a real operating system, he will unknowingly be met with a virtualized network stack. Honeyd would essentially be capturing the TCP traffic the hacker is generating.

In this study, Honeyd was configured to simulate operating systems and services. This was done by configuring a file called Honeyd.conf. In this file the virtual network is created as well as templates on which to assign IP addresses. A template is a virtual machine where open ports, operating system, up time and more, are configured. Each port can be configured to be opened using a script in order to simulate the service. Once the template has been setup, several IP addressed are bound in order to create a full network which looks real to attackers.

Whenever a hacker tries to scan a range of IP's, Honeyd will automatically reply for the IP on which a template is bound. For an IP which isn't bound to a temple, no reply is sent. Thus fake messages are sent to the hacker in attempt to log, fool and waste their time.

In their study it was concluded that low interaction honeypots such as Honeyd can be easily hacked and hackers quickly realise that its a honeypot and stopped attempting to compromise it. But due to Honeyd being a daemon and just simulating operating system's services, hackers were unable to seize other systems using Honeyd, making Honeyd a safe choice from a pure forensics standpoint as Honeyd was sufficient at logging hackers actions.(Akkaya & Thalgott 2010)

B. Research Honeypots

Research honeypots are deployed by security researchers and their goal is to learn tactics, tools and techniques of the hackers exploiting the network and its computer systems. This type of honeypot is a step ahead of production honeypots and give complete freedom to the attacker and is designed to detect and log information without letting the attacker figure out that it's a honeypot. Researchers then share the findings with network and system administrators, anti-virus vendors and system programmers in order for them to patch or update their systems against any security risks. This is effective at acting as an early warning system for 0-day exploits.

In his study Lakhani (2003) talks about a commercial high interaction honeypot named Symantec Decoy Server (Formerly known as ManTrap), providing awareness with regards to the positive capabilities of research honeypots as well as the drawbacks a research honeypot might bring.

This honeypot is designed to act as a decoy system that diverts the attention of attackers to lesser value machines. While doing this the honeypot is able to provide stealth monitoring and thus live attack analysis, zero day recognition of unknown exploits and attacks, reduce false positives and many other positive features.
But such high interaction honeypot systems come with their drawback as Lakhani (2003) goes onto mention that a highly skilled expert is required to deploy and maintain these kind of honeypots. He also goes onto mention that, even tough extra care is taken to make sure that honeypots cant be compromised and pose a threat to production servers. The possibility of the honeypot getting compromised and the attacker gaining access to production servers remains. Therefore a thorough risk assessment has to be undertaken before deploying such honeypots.

C. Things to consider

After reviewing past literature, some points need to be considered before implementing and experimenting on honeypots. First of all, honeypots should be configured in an isolated environment. Where they are protected and not easily accessible, away from production systems as to not risk the attacker gaining access to the production system and compromising sensitive data. Further more, everything that is happening on the honeypot should be logged and stored in a safe environment as it is good to have evidence of any kind of problems that might occur on the honeypot. Such measures can ensure the safety of an enterprises sensitive data, as well as better honeypot logging practices.

III. Research Methodology

In this chapter, the practical work will be presented. First a low interaction honeypot will be implemented and tested

on. After, A medium level honeypot will be implemented and tested on. A high interaction honeypot wont be configured due to the fact that such a system is too complex and time consuming to be implemented in the given time frame of this project.

A. Low level interaction honeypot: Pentbox

Low interaction honeypots are honeypots which emulate services of real operating systems. First, using vmware a virtual machine was installed on a Kali Linux based distribution. On this virtual machine Pentbox was deployed. Pentbox is an open source security suite that has security and stability oriented tools for network and systems. One of these tools is a honeypot. This honeypot is a good starting point as it meets the requirements of being a low interaction honeypot. The Pentbox honeypot is configurable so anyone can create their own service and decide which ports to open and listen on. Due to its limited functionality, Pentbox can only be used to imitate a service running on an end device in order to capture traffic that an intruder is generating.

In order to create a service, the honeypot tool needs to be run from Pentbox. Upon running the honeypot, the user is promoted with 2 options on which to run the honeypot on. ”Fast Auto Configuration” which automatically creates a web server honeypot running on port 80 or ”Manual Configuration” which has more options and can be configured to run on different ports. Option 2 was selected and configured to run on port 23 (telnet) in order to simulate a telnet server service that an enterprise might be running. This was configured so that every time an attacker attempts to connect to the telnet service, a message about the intrusion is logged in a text file and a custom error message is sent to the attacker.

The Pentbox honeypot was also configured on an ftp port 21 in the same way the telnet service was configured. This was done in order to simulate an ftp service which might be used at an enterprise. Port 21 was selected, a directory to which to save intrusion logs was selected and also a custom message to send to the attacker.

The last test that was done on the Pentbox honeypot was to simulate a web server service. This was done in two ways. First, by using the fast auto configuration and secondly, by using the manual configuration. When using the fast auto configuration option, the assigning of port 80, log directory and error message was automatically configured. Using the manual method, the same results where had but the configuration settings would need to be entered manually.

B. Medium level interaction honeypots: HoneyBot

HoneyBot was used as a medium level interaction honeypot. This honeypot creates a safe environment which is used to capture and interact with unsolicited traffic on a network. It is ideal for network security research or as part of an early warning IDS. Using the installation files, HoneyBot was installed on a Windows Server 2016 running on a virtual machine. To an attacker this virtual machine would act as a front facing service which they would attempt to exploit. The honeypot would then be listening in on multiple ports simultaneously. Any attempts at accessing services on the honeypot would be logged and also indicate that a potential attack is happening.

While HoneyBot being a medium level interaction honeypot offers more functionality then a low interaction honeypot such as Pentbox, it doesn’t necessarily mean that the configuration process is harder. Due to HoneyBot being a windows based honeypot, it is able to be setup and configured with the use of GUI. From this GUI you are able to add or delete services and a total of 1357 services come pre configured with the honeypot. This honeypot is also able to provide sound alerts when an intrusion is detected and also send a daily summary of intrusions to a specified email. Logs generated can be viewed either from the Honeypots GUI or exported as CSV.

For this scenario all services/ports on the honeypot where enabled and running but the testing was focused on ports 21, 23 and 80 as these are common ports which are likely to be used by enterprises in internet front facing services which are susceptible to attacks. These tests where conducted using the same Kali Linux virtual machine which Pentbox was run on as as well as the host OS.

The first test that was conducted was using port 21 (ftp). A bash terminal on kali was opened where the command *ftp ”Honeypot IP Address”* was entered. The honeypot then asked the user to enter a username and password. After these are entered an error is displayed to the attacker and all the information related to the attack is logged.

The second test conducted was using port 23 (Telnet). A command line terminal was opened in the host OS where the command *telnet o ”Honeypot IP Address”* was entered. Upon entering the command the attacker would be met with an error saying that they where unable to connect to the host. This attempt would then be automatically logged into the honeypot.

The last test conducted was using port 80 (http). This was done by entering the honeypots IP address in a web browser. Upon entering the web page, an attacker would be met with an ”Under Construction” page which leads to nowhere. This attempted would also be automatically logged into the honeypot.

From these tests, detailed logs where generated which included the data and time of the attack, attackers source IP, port accessed, protocol used as well as any information entered such as username, passwords or strings of text.

IV. Evaluation

In this chapter, the results of tests produced in the methodology section with regards to low and medium interaction honeypots will be discussed.

A. Evaluation: Pentbox

The process of installing Pentbox was rather easy, this was done through a git clone command and then the installation file was executed through bash. Seeing that the system was easy to setup, it wont be hard for an enterprise to implement and deploy it into their network. Services for the honeypot to listen on where easy to configure and logs where detailed and gave a good understanding of where the attack was originating from, what service the attacker is targeting and when the attacker attempted the attack.

While being easy to install and configure, pentbox also had its drawbacks. One of these drawbacks was that the honeypot is only able to listen to one service at a time. This means that every instance of the honeypot has to solely focus on one service and the enterprise would have to be running multiple instances of the honeypot in order to cater to multiple services.

The largest drawback of the honeypot is that it doesn't have much configurability of what happens once the attacker attempts the attack on a service as only a simple error message or page is displayed to the attacker. Therefore, it doesn't take much for an intruder to realise that the service running is fake, as soon as the intruder tries to actually connect on a port. For example when an ftp port is opened and the intruder attempts to connect to it, the server should send back a reply but in this case nothing happens. As a result the intruder quickly realises that something is wrong and aborts the attack.

B. Evaluation: HoneyBot

Honeybot, being a windows based honeypot was rather easy to install. This was done through a simple .exe installation file which automatically installed the Honeypot. Configuration of the honeypot was also rather easy as it was done through GUI which enabled options to be simply ticked or unticked in order to activate or deactivate them. Therefore, installing and configuring this honeypot to be used in an enterprises network should be rather easy.

The honeypot allowed for multiple services to be simultaneously running. Services where easy to add, delete or edit with the click of a button and any incoming attacks to these services where automatically logged. The logs where easy to understand and gave detailed information such as the data and time of the attack, source IP address of the attacker, Protocol used and most importantly any strings of text used during the attack. Such as usernames and passwords. These would be beneficial, as enterprises would be able to keep track and analyse username and password patterns the attackers might use, helping better their existing security measures.

While being better then Pentbox due to being able to listen in on multiple services, it also has similar drawbacks. One of the main drawbacks of HoneyBot is configurability. The honeypot is somewhat limited with regards to this, it does allow the administrator to add custom HTML files to the port 80 service but, all other protocols do not have this configurability. For example, no fake files could be added to the FTP service. Therefore an attacker would quickly realises that it is a honeypot due to the other services not having much depth, especially if the attacker was to port scan the honeypot device. As port scanning would show all the ports that the honeypot is listening in on, making it obvious. Therefore the honeypot is great at detection but not very good at tricking attackers.

V. Conclusion

After this research it can be determined that honeypots are a worthy candidate to be used for security in enterprise networks. They are flexible, cost effective and versatile.

From the methodology section it can be seen that both honeypots where easy to implement, but this isn't the de facto standard for all honeypot implementations. As for this research the honeypots only needed to cater to a very small network infrastructure. Depending on the size of an enterprises network it could prove to be rather complected and time consuming to implement a honeypot. The bigger the network, the more time it will take to make sure that the honeypot is configured correctly thus making sure that it doesn't prove to be a threat to the enterprises security due to improper configuration. The honeypot also needs to be configured in a way that it caters to the enterprises infrastructure by making it as realistic as possible, helping to deceive attackers. It also needs to be made sure that the honeypots logged data is relevant and useful as these logs can be used by network administrators to better the enterprises security. While the implementation of a honeypot is time consuming and not straight foreword, it is an effective security measure. Therefore, It is up to each individual enterprise to determine for themselves if a honeypot is cost effective but it is safe to say that a honeypot is an effective security tool and has its place in enterprises.

While this research has successfully answered its research questions, there are multiple ways this research can be improved upon in the future. This research only tackles low and medium interaction honeypots, thus it leaves the possibility for the implementation and research of a high interaction honeypot. Further improvements to this research can be made by making network more robust in order to make the scenario as realistic as possible. In future research a honeypot could also be created from the ground up. Unlike in this research, where honeypots found on the internet

where used. Developing a honeypot from the ground up could potentially give more accurate information to the ”Are honeypots tricky to implement?” research question.

References

Akkaya, D. & Thalgott, F. (2010), Honeypots in network security.
URL: *http://urn.kb.se/resolve?urn=urn:nbn:se:lnu:diva-6600*

Lakhani, A. D. (2003), Deception techniques using honeypots.
URL: *https://www.isg.rhul.ac.uk/ pnai166/thesis.pdf*

Tsikerdekis, M., Zeadally, S., Schlesener, A. & Sklavos, N. (2018), Approaches for preventing honeypot detection and compromise.
URL: *https://www.researchgate.net/publication/328430317*