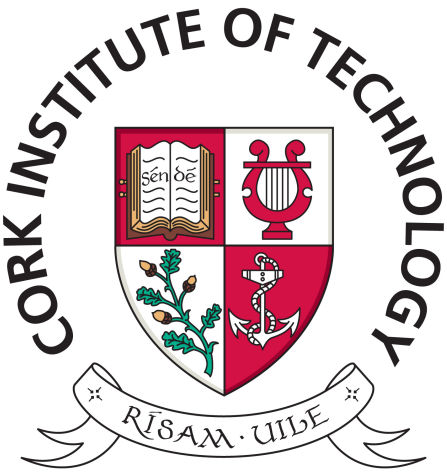

An Intelligent Honeypot

by

Aidan Mitchell

This thesis has been submitted in partial fulfillment for the degree of Bachelor of Science in IT Management

in the
Faculty of Engineering and Science
Department of Computer Science

May 2018

Declaration of Authorship

I, Aidan Mitchell , declare that this thesis titled, 'An Intelligent Honeypot' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for an undergraduate degree at Cork Institute of Technology.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at Cork Institute of Technology or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this project report is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

CORK INSTITUTE OF TECHNOLOGY

Abstract

Faculty of Engineering and Science
Department of Computer Science

Bachelor of Science

by Aidan Mitchell

Honeypots are an essential element in current cyber defensive arsenals. With IT becoming such a large part of the world, malicious attacks such as the Wanna-Cry virus are growing in numbers. Honeypots are a way of trapping these attackers before they even get a chance to get in to your system. This project involves the creation of an intelligent honeypot for network monitoring. The basic premise is for the honeypot to respond positively to attacker vulnerability probes. For example, if port 445 is probed then the honeypot takes on a windows profile and simulates vulnerability for a SAMBA attack. The honeypot could potentially simulate multiple simultaneous personalities/profiles. This will, if successful, confuse the attacker and obfuscate network profiling and thus improve the security of the system. Part of this project will include an investigation into the currently available honeypot software tools to determine the best approach to achieving this final system.

This project would have to include writing a python file to act as a honeypot, this honeypot python file should have the capabilities to take on different profiles and simulations. It will also involve setting up a Linux server as a virtual machine to host the python file. It should also include using network monitoring tools such as Tshark to use on the server to find any irregular attacks on the system which can be used as research for the honeypot file.

The beneficiaries of this project will be most IT Security teams as it will help to make their systems more secure. As stated before Honeypots are an essential element in current cyber defensive arsenals. The success of this project will be measured by whether or not the script works, the number of threats this project prevents and also measured by the amount of knowledge that can be learned about attackers from this Honeypot.

Acknowledgements

My many thanks go out too:

Beweleys for demonstrating their great coffee making skills, which has proved invaluable.

Dr. Sean McSweeney, my project supervisor for my research phase, for helping steer me in the right direction and for being available to help whenever I needed it.

Mr. John O Brien, my project supervisor for the implementation phase, who gave me the drive to help me complete the implementation of this Honeypot.

Dr. Josh Reynolds for providing me with the original idea for this project, which I'm excited to complete.

Dr. Micheal O Sullivan from IBM Cloud who pushed me into choosing a topic I knew little about but was still interested in, therefore gaining more knowledge.

Mr. Christopher Murphy from IBM X-Force for teaching me how too be proud of a good project.

Mr. Peter Halligan from IBM Cloud for pushing me to begin my research for this project as early as possible.

The folks at IBM Security for making me interested in the Security Field of IT during my time as an intern there.

My dog Muttley for always being by my side during long hours of staring at a laptop screen.

And most importantly, my class mates and study buddies, Megan Kearney and Norbert Skurzewski, for lending a helping hand whenever I needed it.

Contents

List of Figures

Abbreviations

IT	**I**nformation **T**echnology
IBM	**I**nternational **B**usiness **M**achines
VM	**V**irtual **M**achine
SSH	**S**ecure **SH**ell
CIT	**C**ork **I**nstitute **T**echnology
ISP	**I**nternet **S**ervice **P**rovider
USB	**U**niversal **S**erial **B**us
ID	**ID**entification
IP	**I**nternet **P**rotocol
CNN	**C**able **N**ews **N**etwork
OS	**O**perating **S**ystem
MIT	**M**assachusetts **I**nstitute **T**echnology
AI	**A**rtificial **I**ntelligence
SMB	**S**erver **M**essage **B**lock
SQL	**S**tructured **Q**uery **L**anguage

Dedicated to Barry B. Benson

Chapter 1

Introduction

1.1 Motivation

The internet is well and truly on the rise, with 3.77 billion global internet users currently [1]. Nearly all of these users have had or will have some form of personal data online, and where there is personal data, there is always somebody trying to gain hold of it. A quick look around any room in a home or business and anybody could pick out a few internet connected devices, many of which don't have security administrators monitoring them. There are many examples of this happening daily, with malicious attacks this year such as the Wanna-cry or the Petya virus[2].

This makes information-security and its arsenal an utmost importance in the IT industry. But while many of these tools do help protect systems in an IT industry, only few of these tools have the ability to entrap an attacker. That is the motivation for this project, to try and develop a Honeypot which can be used by information-security teams that can be used to detect and record an attack while it is happening, while gaining future knowledge on how to prevent such attacks.

But then why stop there at a simple Honeypot system. Their are many kinds of Honeypots already available to information security teams, so how will this particular one be different? After performing some research I noticed that certain ports on a system are related to specific types of attacks, and while there are many different types of Honeypots available, none of these Honeypots can adapt to secure separate ports of a system. So even more motivation came along for this project to not only be just a Honeypot, but it will be an intelligent honeypot with the ability to adapt ports.

1.2 Executive Summary

As stated in the abstract, the basic premise is for the honeypot to respond positively to attacker vulnerability probes. What makes this an intelligent honeypot, will be its ability to adapt to different port probes. Say for instance port 22 is being probed, then the honeypot will take on a Linux profile and appear vulnerable for a ssh brute force attack. While on the other hand if port 445 is probed, the honeypot will take on a windows profile and simulate vulnerability for a SAMBA attack. This process will be able to confuse the attackers while at the same time, profile their attacks, therefore giving the security administrator information necessary to prevent future attacks.

The honeypot itself will be hosted on a virtual Linux server using VMware workstation. This server will then be monitored using a tool such as T-Shark, this is for added awareness for when a malicious attack may happen. Another feature of this project is that it will be written using python scripting. Python has a vast range of packages available so it will be most useful. The python file itself will be monitoring individual ports for malicious attacks, when these ports are probed, it will be using the Scapy package to craft packets to take on profiles while at the same time preventing the attacker from accessing the system.

1.3 Contribution

My largest contribution to this project will be my python script programming skills, these have been gained through two ways. One being work placement, where I helped IBM Security in writing python files for their networking team. The second way these skills have been gained is from my script programming modules where I was taught python extensively. Other contributions to this project are my skills learnt from the Inter-networking module in CIT. This module has taught me skills such as setting up and using virtual machines, as well as using a Linux command line. These will be useful when setting up a Linux server to host the virtual machines. Finally my information-security skills gained from again my work placement in IBM Security, and also my Security Monitoring module in college, which taught me the basics of information-Security for multiple systems, will be very useful as this project is information-security based.

1.4 Structure of This Document

Chapter one of this Document is straight forward, it will introduce the reader and introduce the core elements of the project.

Chapter two will give some background to the project, by explaining to the reader what area of IT the project will focus on and try to give the reader more of an understanding of the project.

Chapter three will then have a look at the problem the Honeypot will be trying to solve. Each functional and non-functional requirement will be listed in this Chapter, while relating them to solving each problem.

Chapter four will cover topics such as risk assessment and architecture, along with a scheduled plan for the implementation phase of the project.

Chapter five will discuss the implementation of the Honeypot and which steps were involved in it, as well as discussing the use of each tool used.

Chapter six will conclude the document, answering any questions asked throughout the document and giving an example of how I wish to better the Honeypot in the future.

Chapter 2

Background

2.1 Thematic Area within Computer Science

1. The core topic of this project is an intelligent honeypot system. This Honeypot will have the ability to take on different profiles to confuse hackers who are trying to perform attacks.

2. The core area this intelligent honeypot falls under is Network security. This is due to the scripts ability to monitor ports and prevent attacks. Another core area this honeypot falls under is System security as it will help prevent future malicious attacks to a system.

3. The main area of computer science, which both core areas fall under is Security and Privacy. This is self explanatory as both areas are Security related.

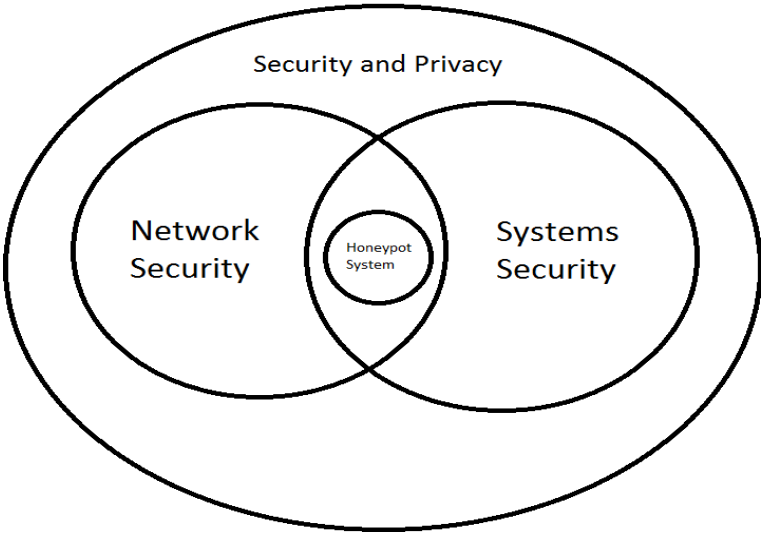

FIGURE 2.1: A picture of the thematic area.

2.2 Project Scope

Computer Security is "The prevention of or protection against access to information by unauthorized recipients, and the unauthorized destruction of or alteration of information." [3] while Computer Privacy "is is the privacy and security level of personal data published" [4]. If you put both of these definitions together Security and Privacy in Computer science, is the protection of unauthorized data from malicious attackers.

Within the area of Security and Privacy, there are many different core areas, although the ones which this specific project fall under are Network Security and System Security. Say a person has a WiFi router at home with a good few devices connected, this is whats known as a network and Network Security is the protection of this network.

A system on the other hand, would be one of the devices themselves which is connected to the network, and system security is the protection of this device.

Finally a Honeypot system, is a computer system, which is set up to try and lure attackers into it, the attackers moves are then monitored from the second they try and gain access, then when they have gained access, they become blacklisted from not only the system, but also the network. This is why the honeypot falls under both system and network security.

2.3 A Review of the Thematic Area

- The top 5 International Conferences and Journals most related to a honeypot system are as follows:
  - Honeypots a New Paradigm to Information Security - R.C. Joshi, Computer Science and Engineering Department, IIT, Roorkee, India
  - A Survey: Recent Advances and Future Trends in Honeypot Research - Matthew L. Bringer, Christopher A. Chelmecki, and Hiroshi Fujinoki, Department of Computer Science, Southern Illinois University, Edwardsville, Edwardsville IL 62025, US
  - Monitoring unauthorized internet accesses through a honeypot system - Mario Marchese, Dipartimento di Ingegneria Navale, Elettrica, Elettronica e delle Telecomunicazioni (DITEN), Universit degli Studi di Genova, Genoa, Italy
  - Distributed Port-Scan Attack in Cloud Environment - Prachi Deshpandel, Aditi Aggarwal, S.C.Sharmal, P.Sateesh Kumarl, Indian Institute of Technology Roorkee, Roorkee-India-247 667

  - When Virtual Is Better Than Real - Peter M. Chen and Brian D. Noble, Department of Electrical Engineering and Computer Science, University of Michigan

- The top 3 most recent books/texts related to a honeypot system are as follows:
  - Hacking the Hacker: Learn from the Experts Who Take Down Hackers - Roger A. Grimes, April 18, 2017
  - A Practical Guide to Honeypots - Eric Peter, April 15, 2008
  - Honeypots: Tracking Hackers - Addison Wesley, September 13, 2002

- The top 5 companies/organizations potentially interested in the Intelligent Honeypot are:
  - IBM Security - Offer a wide range of Security Solutions for both its internal company, and external clients. They are a large company with many different servers and networks worldwide, so a Honeypot would be a perfect fit for this company.
  - eSentire - This company offers more Security Monitoring services too external companies, but a Honeypot would also suit them as it will provide them with a way too gain knowledge on the hackers which they protect other users from.
  - Alien Vault - Much like eSentire, Alien Vault offers Threat Detection and Incident Response services too external companies. A Honeypot would suit this company for their incident response services as again they could use one too gain knowledge on their Hackers.
  - FireEye - This company offers a wide range of Network Security Solutions, which is a perfect fit for a Honeypot as Honeypots protect Networks.
  - Intel Security (McAfee) - Offers many Security Solutions for Businesses and Single Users, offering a Honeypot too businesses would help these businesses may distract hackers from real data.

- The top 6 wiki/forums/blogs/Youtube channels most relating to a Honeypot System are:
  - Honeypots - Computer and Network Security Channel - Youtube
  - Honeypot (Computing) - Wikipedia
  - Honeypots - ethics.csc.ncsu.edu - Blog
  - Honeypots: The Sweet spot in Network Security - ComputerWorld - Blog
  - Discussion Topics - ProjetHoneypot.org - Forum
  - Information Security - Security.Stackexchange.com - Forum

2.4 Current State of the Art

What is a honeypot? The exact definition of a Honeypot is hard to form, however most definitions are similar to each other. The Computer and Network Security Youtube Channel defines a Honeypot as "An artifact used to deceive potential intruders and attackers"[5] While the book, A Practical Guide to Honeypots, defines them as "A server that is configured to detect an intruder by mirroring a real production system. It appears as an ordinary server doing work, but all the data and transactions are phony. Located either in or outside the firewall, the honeypot is used to learn about an intruder's techniques as well as determine vulnerabilities in the real system"[6]. While these are both two very good definitions, my own definition of a Honeypot is a software trap set up to try and lure in and catch attackers on a network. This is done by making a server look as though it is vulnerable, while at the same time making the server look as though it is host to something valuable. Then a monitoring script or software is ran on the server, either monitoring ports or commands entered or both, to help learn what an attacker will try to perform on the system, this data is used prevent future attacks on the network while also blacklisting the previous attackers from entering the network again.

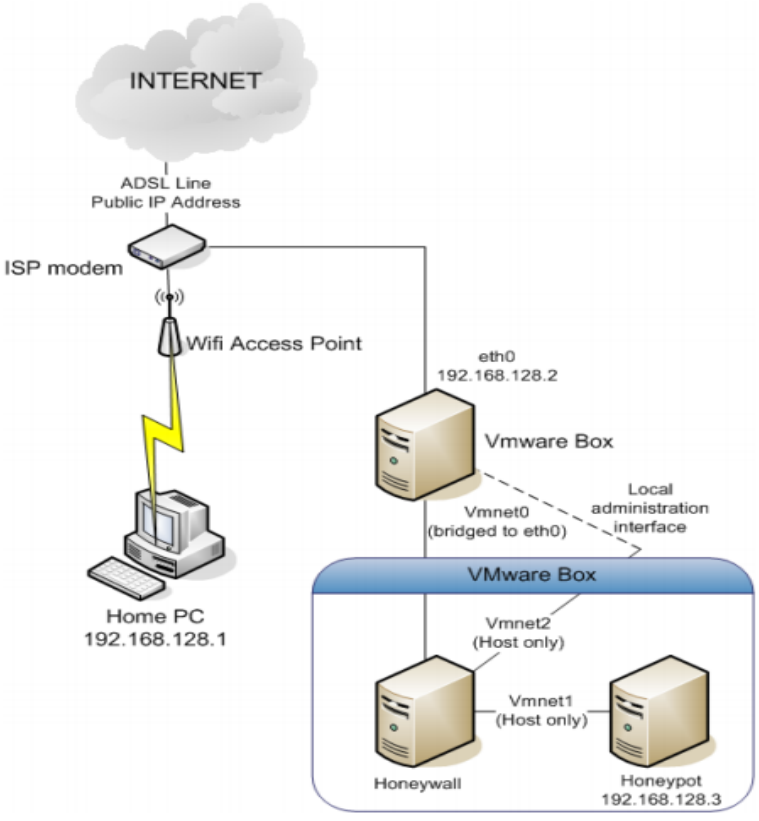

FIGURE 2.2: VMWare Honeypot Demonstration.

The most asked question surrounding Honeypots, covered by many papers such as Recent Advances and Future Trends in Honeypot Research, is the question are Honeypots ethical? Many IT Professionals deem this to be a controversial topic because while Honeypots are completely legal, few find it to be a form of entrapment. According to the blog ethics.csc.ncsu.edu "The argument is that since it is both unethical and illegal to lure someone into stealing an object, why is it legal or ethical to lure an individual into committing a computer crime?"[7]. Its kinda like luring a person into your shop, asking them to steal from you, then arresting them when they do steal. Hackers are lured into very vulnerable looking servers, not knowing that they're being monitored, and are then blocked from networks. Some of these hackers may simply just be white hat hackers looking to provide companies with a list of their vulnerabilities, only to be blocked out. On the other hand, many people find Honeypots to be completely ethical as these attackers should not be attempting to access networks which they aren't authorized to access, which is unethical on their part. They are also deemed ethical as they provide a means of making networks and systems more secure, by preventing attackers from accessing the rest of the network, while monitoring their moves. This proves that Honeypots are completely ethical, as long as the security admins do not go out of their way to make attackers hack their systems.

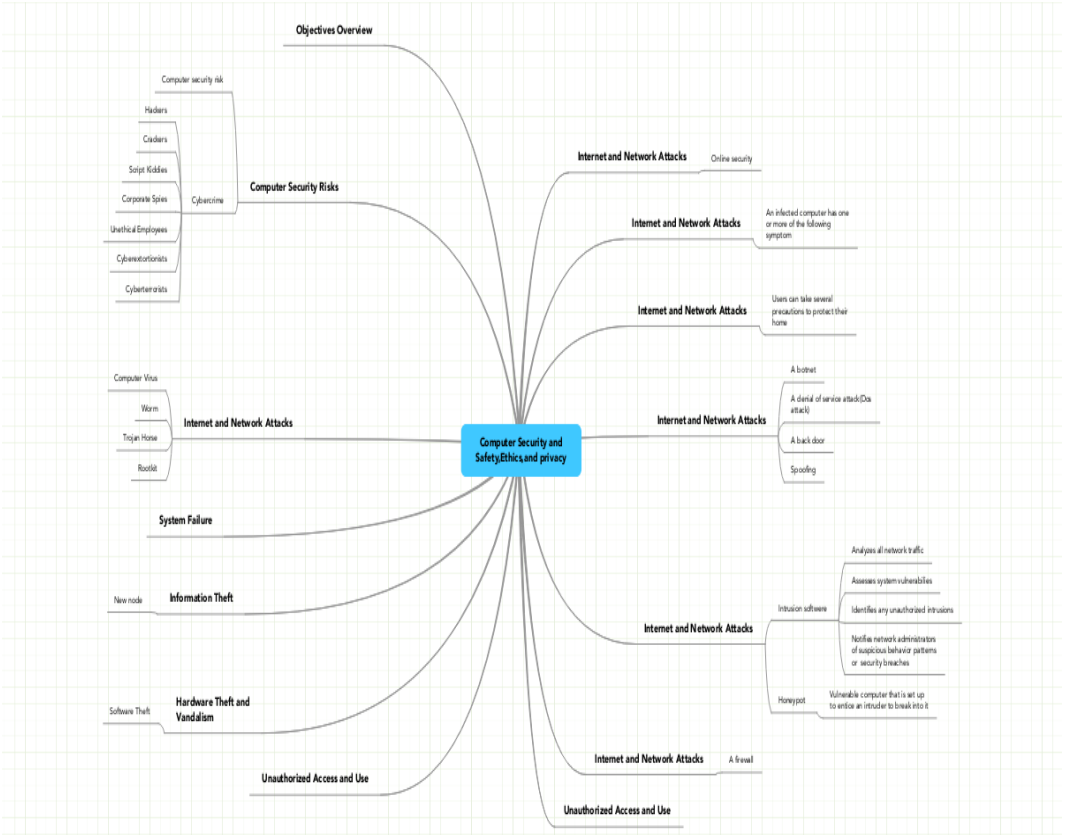

FIGURE 2.3: Ethics in Information Security.

There are many advantages to a honeypot system in the Information Security area of IT. One of the biggest advantages of a Honeypot is that it produces very little data sets with huge value, as apposed to system logs, firewall logs and intrusion detection systems which provide very large sets of data, most of which has little value to Security Administrators. In the book Honeypots: Tracking Hackers it states "Instead of logging gigabytes of data every day, most honeypots collect several megabytes of data per day, if even that much. Any data that is logged is most likely a scan, probe, or attack information of high value"[8]. What makes this small data so valuable is that every part of it is coming from what is more than likely an attacker, and so every part of this data goes towards improving other machines on the system and improving the security. Another advantage of a Honeypot is stated in the book A Practical Guide to Honeypots is that Honeypots produce "Fewer false positives since no legitimate traffic uses honeypot"[6]. This is because a Honeypot should be deployed on a machine with no real purpose other than to be a Honeypot, which means that any attempt to access this machine should be recorded as unauthorized access and so attempts to access should not be marked as a false positive. The same gesture can be used on the opposite side of things, Honeypots can be used to catch false negatives. This again is because most traditional detection technologies such as anti-virus software look for certain patterns for an attack and so might miss when new types of attacks come into the system. But on a Honeypot, as stated above, any attempts to access its system should be marked as unauthorized, meaning even new types of attacks to this system will be caught.

Like most things in this world, everything that comes with an advantage comes with a disadvantage. One of the biggest disadvantages of a Honeypot, is its very limited field of view. Again, according to the book A Practical Guide to Honeypots, "Honeypots only monitor interactions made directly with the honeypot - the honeypot cannot detect attacks against other systems"[6]. What this means is that if an attacker tried to hack into a different system on the network, the Honeypot would in no way know that this attack is occurring, whereas a network firewall may have ways of blocking these attacks. This is backed up by Honeypots: Tracking Hackers stating that "honeypots have a microscope effect on the value of the data you collect, enabling you to focus closely on data of known value. However, like a microscope, the honeypot's very limited field of view can exclude events happening all around it"[8]. This is a good description of its limited field as it shows that while a Honeypot has a very limited field of view on the network, what little data it gathers can be made very useful. Another disadvantage to Honeypots is risk. While a Honeypot does provide some level of Security to a network, any new form of technology, especially ones that expose ports on a network, can be off major risk to an organization. The best practise is to use the Honeypots to work along side other security mechanisms.

As stated above a Honeypot contains small data sets, which have a lot of value to a security administrator, but what exactly is in these data sets and what knowledge can be gained from them? As stated above, any data that is logged by a honeypot is most likely a scan, probe, or attack information of high value. Research Honeypots are a type of Honeypot based purely on gaining knowledge of motives and information about how cyber-attackers work. In the Information Security Forum on Stack Exchange, three very good perceptions of what to do with Honeypot Knowledge are made. The first being watch the behaviour of the data, this will be able to help a security administrator to characterize the different types of person attacking their network[9], and in return they can put measures into place to block these certain types of person from the network. These characteristics of a person can also provide what the attackers may be after, what there goals were and who they might be, again which is all useful in putting security measures into place. If a attacker is very poor at their job, this will bring us to the best piece of knowledge a honeypot can provide, which is who exactly the attacker is, this will most definitely provide more ways to secure a network as well as maybe even bringing about court cases in particular companies.

The particular types of Honeypots often falls under the amount of interaction an attacker will be allowed to have with it, which in return gives three main types of Honeypots, the first being a low interaction Honeypot. The book Hacking the Hacker states "A low interaction honeypot only mimics very simplistic port connections and logs them"[10]. What this is is basically a Honeypot which simulates small vulnerabilities and collects information on them, but at the same time does not present the attacker with a usable system. The second type of Honeypot available is of course a medium interaction honeypot. A medium interaction Honeypot, taken from notes from a Security Monitoring lecture I have taken in Cork Institute of Technology, "imitates a production service in a very controlled environment that allows some interaction from an attacker"[11]. What this means is that it may have a website hosted on it to make it look as though it is a production environment, but everything on this server is controlled and monitored. Which brings us to the final type of honeypot available, a high interaction honeypot. This is very similar to a medium interaction honeypot in that it imitates a production service, but the difference between a medium interaction honeypot and a high interaction honeypot is that it isn't a controlled environment whatsoever, and attackers are given a free for all on the system, until such time that the system is reset again. These high interaction honeypots are mainly used in research environments as it gives many insights into the characteristics of an attacker, but are usually still monitored greatly.

Characteristics	*Low-Interaction Honeypot*	*High-Interaction Honeypot*
Degree of Involvement	Low	High
Real Operating System	No	Yes
Risk	Low	High
Information Gathering	Connections	All
Compromised Wished	No	Yes
Knowledge to Run	Low	High
Knowledge to Develop	Low	Mid-High
Maintenance Time	Low	Very High

FIGURE 2.4: Low Interaction vs High Interaction.

Now we will look at the use cases of each type of Honeypot, i.e. what type of honeypots we will use in different situations. In the report, Honeypots: A New Paradigm in Information Security it states that "Low interaction Honeypots are used when: There is insufficient Hardware to set up a Honeynet and the risk of another type of Honeypot is not acceptable"[12]. What this means is that low interaction honeypots are mainly used in smaller networks which don't have the means to deploy a Honeypot with larger interaction. These smaller networks would usually be Homes or small businesses looking to maybe get a brief insight into who would be trying to attack their systems and protect private information on their devices. A medium interaction Honeypot, according to Honeypots: A New Paradigm in Information Security, are used when "a small amount of risk is acceptable"[12]. These types of Honeypots would be used in larger businesses such as IBM Security or eSentire, too lure in attackers in a controlled fashion, only to prevent them from accessing their networks again. High interaction Honeypots on the other hand, again according to Honeypots A New Paradigm in Information Security, are used when "The purpose is to observe the intruders activities or behaviours"[12]. What I gather from this is that High interaction honeypots are mainly used for research purposes, i.e. in Universities or Colleges to gather information on how a intruder on a system would act and gain a profile on these intruders. This information is used to research new ways of blocking attackers which can be passed onto security companies to be implemented.

Now it is time to look into the area of information security, also often shortened down as info-sec in which a Honeypot would fall under. But what is information security? Information Security according to Wikipedia is "the practice of preventing unauthorized access, use, disclosure, disruption, modification, inspection, recording or destruction of information."[13] Which in general terms means that it is the act of preventing a person from stealing your information. This could cover a wide range of fields, but in IT it is very important as IT is Information Technology itself. It is often confused with cyber-security which is, again according to Wikipedia "the protection of computer systems from the theft and damage to their hardware, software or information"[14]. The confusion between cyber-security and information-security is which one falls under a subset of which? i.e. is cyber-security a subset of information security, or is information security a subset of cyber-security? The answer to this is that they are equals with an overlap. Cyber-security is the security of all things IT related, which includes information stored on these devices. While information security is the security of every type of information, which can also include information stored on IT devices. The overlap being that both areas of security cover the security of information on IT devices.

FIGURE 2.5: Information Security Risks

Companies such as IBM Security or eSentire are largely based around information security, this arises the question, why is information security needed? According to it-governance.co.uk, fifty five million records were leaked in October 2017 alone[15]. Some of these attacks involve Ransomware such as R6DB, a gaming service which provides statistics for Rainbow six, going down after a hacker cloned and wiped the companies database and held it for ransom. [15] Outside of the cyber-world, this would be marked as kidnapping or theft and dealt with from whichever companies authorities, so why would a cyber-crime be dealt with any differently. Another record leaked in October was a data breach which unencrypted Heathrow Airport Files were found on a USB stick dumped on the street. [15] These files involved many security threats such as the exact route the Queen takes when she visits the airport for a trip, timetables of patrols for the security team in Heathrow Airport and finally files disclosing every type of ID needed and which areas these ID's can gain a person access too. These files could have been a huge terrorism threat to the United Kingdom had the USB not been found by a different person who wouldn't have handed it into authorities. This is a perfect example of why information security is needed, a lack of information security in Heathrow Airport provided real terrorism threats to the Airport and impacted the Airport Security team themselves.

How does a Honeypot relate to the Information Security area? As stated above in section 2.1, a honeypot system falls under both Network Security and Systems Security as core areas of IT. Network Security is the security of the network which each system is connected too, while systems security is the security of each individual system, these systems can be smart phones, laptops or any type of device connected to the network. These are very important in Information Security as each of these systems, hold a lot of personal information for there users in today's day and age. This information can range from credit card details to calendars with schedules of users daily routines. Honeypots could also be deployed on Companies networks to help prevent confidential company data from being breached. In the paper, Recent Advances and Future Trends in Honeypot Research, it states "Attackers of this type are after information that they can use, therefore the honeypots must provide data that the attackers will want but do not need to know." [16] The part of the sentence that states that attackers are after information is the important part of this citation, it shows that attackers who meet Honeypot Systems are usually only after the information on these systems, and as stated above, information security is the act of preventing a person from stealing your information, which means a Honeypot would prevent the theft of information.

One particular type of Honeypot which has came too light in 2009, is a Honeynet. While a Honeynet is called a 'type' of Honeypot, it appears too be more of a group of Honeypots networked together. Wikipedia states a Honeynet is "Two or more honeypots on a network form a honey net. Typically, a honey net is used for monitoring a larger and/or more diverse network in which one honeypot may not be sufficient." [17] Which shows too form a Honeynet, a minimum of two Honeypots will need too be deployed on a network. But then that raises the question, why would more than one Honeypot be needed? Typically on larger networks, an attacker would find it strange too only find one odd vulnerable system, but say the attacker found an entire network of vulnerabilities (the Honeynet), they may just assume the network had poor security and carry on. Honeynets have become so popular in recent years that a non profit research organization which focuses specifically on Honeynets from Australia, they call themselves The Honeynet Project [18]. This organization has grown so large in recent years that even Google has been helping them with research, and they set up workshops to promote Honeynets around the world in places such as Dubai, Paris and San Antonio[18].

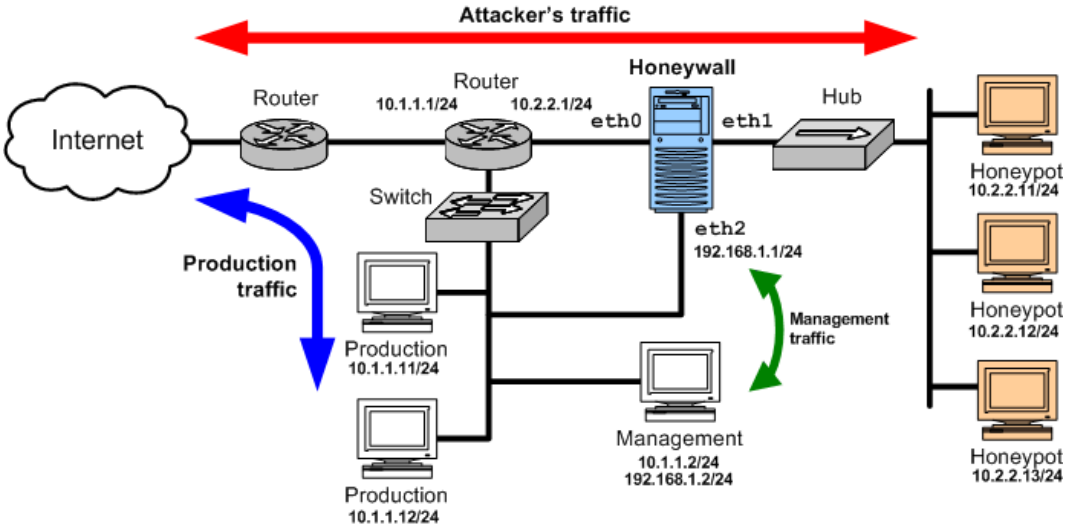

FIGURE 2.6: A HoneyNet.

While Honeypots are great for defending systems, its important too look at what other tools may be on hand for Security Administrators, which will also defend the systems. The first tool helpful tool on hand for an administrator, which every network should already have, is a firewall. "All enterprises make use of firewalls as these systems function to provide enterprises with an adequate level of safety against attacks and damages." - Chang Liu [19]. A firewall is basically a way too control what network traffic can enter or exit a system, and is always used along side Honeypots. This is because a Honeypot would sit outside a firewall on a network, while the rest of the network sits inside the firewall, in order to ensure that an attacker will only see the honeypot when they're

scanning through the network. Another tool used to defend systems on a network from being attack is an intrusion detection system. "A computer-implemented intrusion detection system and method that monitors a computer system in real-time for activity indicative of attempted or actual access by unauthorized persons or computers." - Craig Rowland[20]. What this means is that an intrusion detection system typically does exactly what it says in the name, i.e. it detects intrusions on the network. In other words it can detects malicious activity on the network. While these systems can be used alongside Honeypots, it is more common to use either one or the other. This is because an intrusion detection system would be alerting a security administrator every time an attacker tries to access a Honeypot, while the Honeypot normally records these interactions anyway.

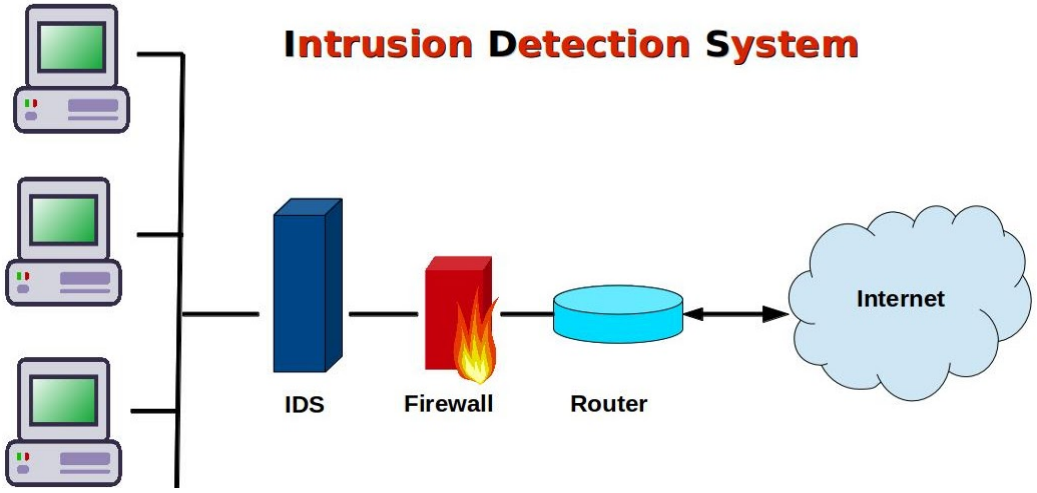

FIGURE 2.7: An Intrusion Detection System.

Packet capture tools such as Tshark or Wireshark are commonly used in large networks too monitor traffic and pick up on malicious activity. But why are these tools used? Firstly Wireshark is defined as being "a packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education."[21]. The reason why Security experts would use Wireshark would be to record, monitor, and analyze packets both coming and going on a network and try to detect malicious activity from these packets. An example of these tools being used as security tools, is in during a teardrop attack, fragmented packets will be sent to the machine being attacked. Wireshark or Tshark are able too record these packets and a Security administrator will be able to know when this attack is happening, as well as what ip address its coming from.

Virtual machine technologies have been on the rise in recent years, here this report will look at the pros and cons of this technology. The largest advantage of using Virtual Machines instead of a physical machine, is the cost of these machines; if a user wanted multiple operating systems to use in a physical environment, they would have too use a separate physical machine for each one, but in a virtual environment, many different operating systems can be deployed too one single machine, making it cheaper too use a virtual environment[22]. The biggest disadvantage of virtual machine technology is the storage space it uses. Many different Operating systems in a virtual environment would generally take up a lot more storage space than a single operating system being used in a physical environment. [22]

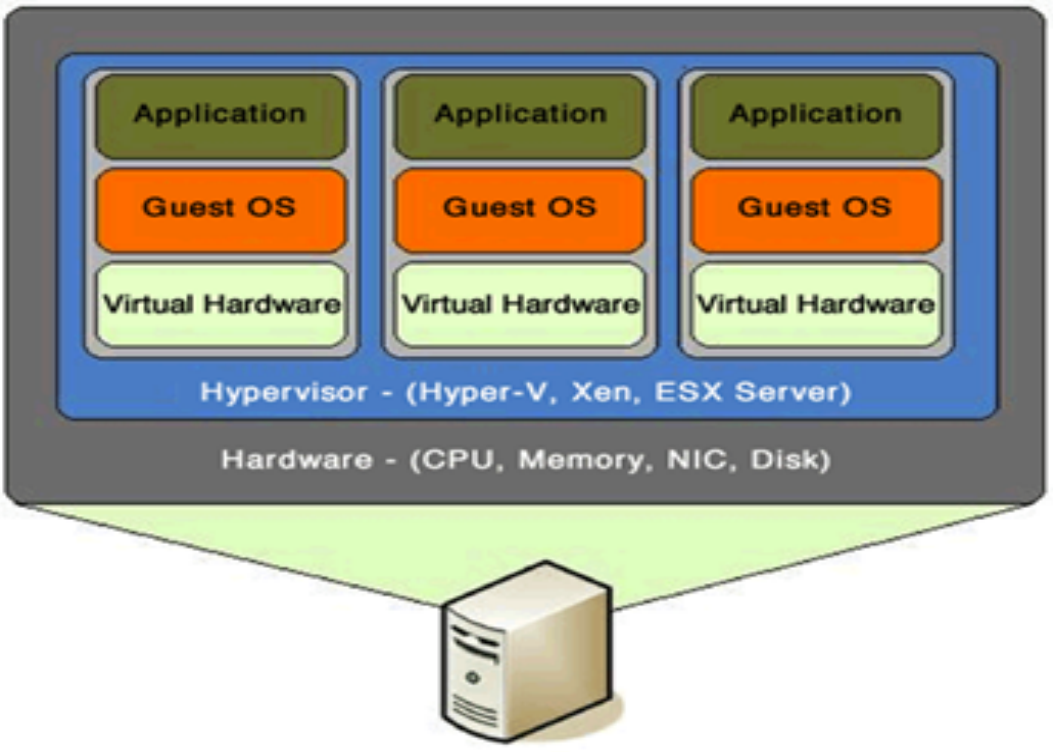

FIGURE 2.8: VM Architecture.

Chapter 3

An Intelligent Honeypot

3.1 Problem Definition

During the past year, cybercrime has hit its all time high. A news article by CNN has called 2017 "the year nothing seemed safe."[23] and many people would agree with this. But it doesn't seem like it would be stopping there, as data breaches each year appear to be growing in size, with 480 million data breaches in 2015[24], then 3.1 billion data breaches in 2016[25], and finally in 2017, the number of data breaches went up by another 21 percent[26]. Many of these 2017 attacks have became quite popular, such as WannaCry[23], which locked down Windows computer systems until a ransom was payed. Or NotPetya[23], which again held Windows OS for a ransom. Or finally Bad Rabbit[23], which infected computers by posing as Adobe Flash player in online advertisements, then scanned networks looking for other users credentials. Theses attacks have became a massive problem as more companies also frequently fail to patch security flaws in a timely manner.

Not only has 2017 been the worst year on record for cyber attacks, MIT Technology Review also believes that we should really worry about these attacks in 2018[27]. It predicts attacks such as more huge data breaches, ransomware in the cloud and even the weaponization of AI.

This will be the problem which I am trying to solve with this project, to try and develop a Honeypot, which will not only help these companies gain an insight into how these attacks are happening, but it will also have the ability too adapt to different attacks as cybercrime grows.

3.2 Objectives

It is very important for objectives or goals too be clearly listed, especially in a self managed project.

FIGURE 3.1: Unlisted Goals.

The objectives I wish too achieve in my Intelligent Honeypot project are as follows:

1. Improve security within an organizations network.
2. Document this project clearly.
3. Provide insight on potential risks.
4. Spoof packets such as SSH or SMB on specific ports.
5. Python will be used too program these features.
6. Host this project on a virtual machine.
7. Use a packet capture tool on the same virtual machine.
8. Demonstrate a fully working project.
9. Handover this project.

3.3 Functional Requirements

Here the functional requirements for this project will be listed. These are functions which the Honeypot System itself will have.

1. Ability to adapt to different ports as they are being scanned.
2. The ability to spoof packets and send them to different ports.
3. To log activity on these specific ports.
4. To use packet capture tools on this system.
5. Use Linux on a virtual machine to host this system.
6. A script which can interact with the Honeypot.

3.4 Non-Functional Requirements

Here the non-functional requirements for the project will be listed, these requirements are ones that specifies criteria that can be used to judge the operation of a system.

1. The script on this system would have too be highly available with no downtime.
2. The system will also have to be highly reliable.
3. The Honeypot will have too have minimum capacity.
4. The script on the system will have too have a way of recovering itself if their is unexpected downtime.
5. The response time of this system to an attacker would have to be near instant.

Chapter 4

Implementation Approach

4.1 Architecture

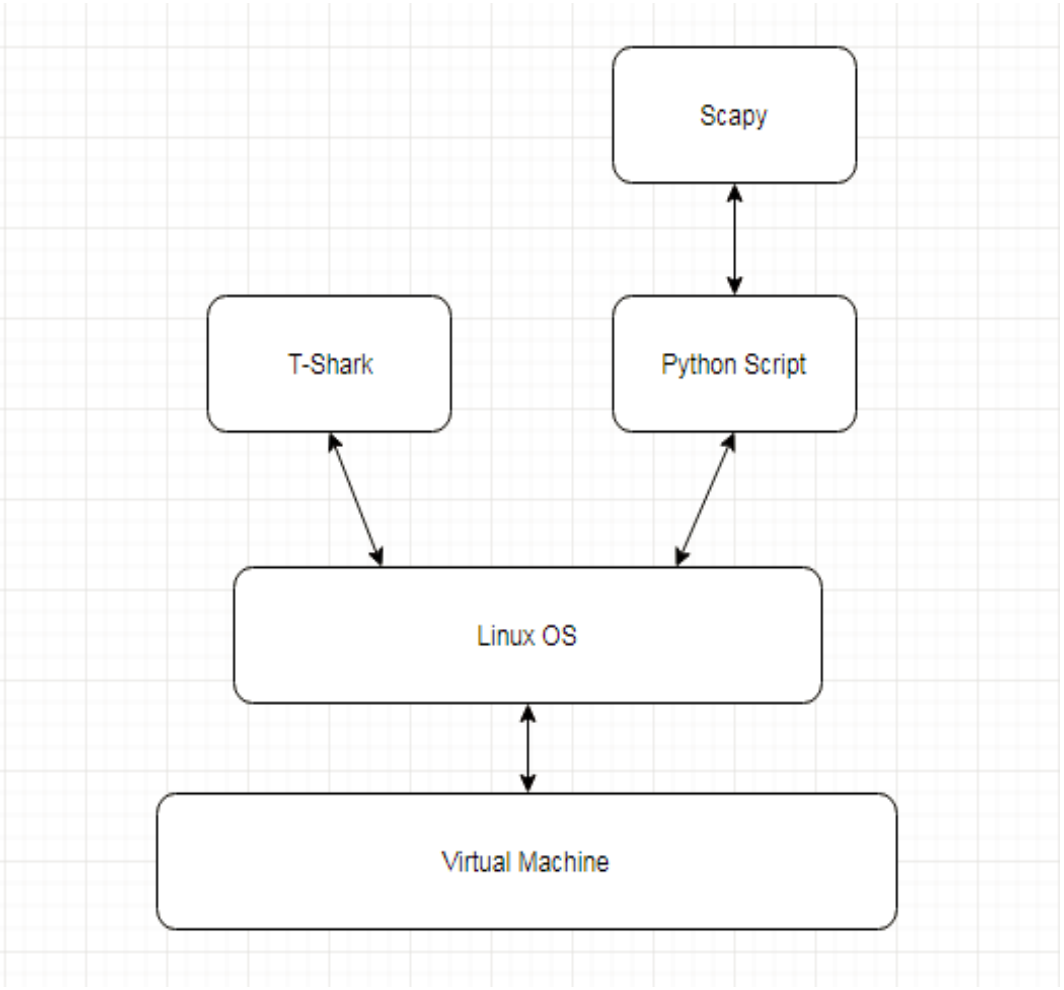

FIGURE 4.1: The technologies involved.

In the above diagram, a view of the technologies involved for the Intelligent Honeypot can be seen. Starting from the top, Scapy is a package which can be imported into Python, which appears too be the most important aspect of the Honeypot. This is because of both its Port monitoring abilities, alongside its abilities too craft packets. The port monitoring abilities will help the Honeypot too monitor specific ports for when attackers are scanning them. While the packet crafting abilities, will be able too make separate ports look like they have separate functions, i.e. making port 22 look as though it has SSH running on it and is a Linux OS, while port 445 has SMB running and looks like a Windows OS.

The next layer in the architecture diagram includes a Python script. Python was chosen originally as it was simple too set up and included a wide range of packages which would help this project along, but after more research, it was discovered that the package Scapy would best suit this Honeypot, and this package was designed specifically for Python. Therefore solidifying python as the choice for scripting on this Honeypot.

Alongside python an instance of T-Shark will be running. T-Shark is a packet capture tool which runs on the command line. The reason for this being is that while Python and Scapy will be monitoring specific ports and adapting too different attacks, T-shark will be monitoring every port in one go. This will help organizations just in case a threat tries too attack a port which the python script is unaware off, and in doing so making the Honeypot a more secure tool.

Underneath each of these tools, a Linux Operating System will lie. This OS will be server based. The reason for the choice of Linux for this project is that it is open-source and highly customizable, making it easy too interchange features of this project if future development were too happen. Another reason why Linux was chosen for the Intelligent Honeypot is that it comes with Python pre-installed. Which will be beneficial for when it comes a time for writing the Honeypot script.

In order too host the Linux Operating system, a virtual machine will be used. The choice for virtual machine software for this Honeypot is VMware Workstation. This software was chosen as VMWare is the most widely used Virtual Machine software in many industries, and I have quite a bit of experience with it already.

4.1.1 Use Case Description

The following section will enumerate each of the functional requirements for this project and describe how each of these requirements will be tackled.

- Ability to adapt to different ports as they are being scanned.

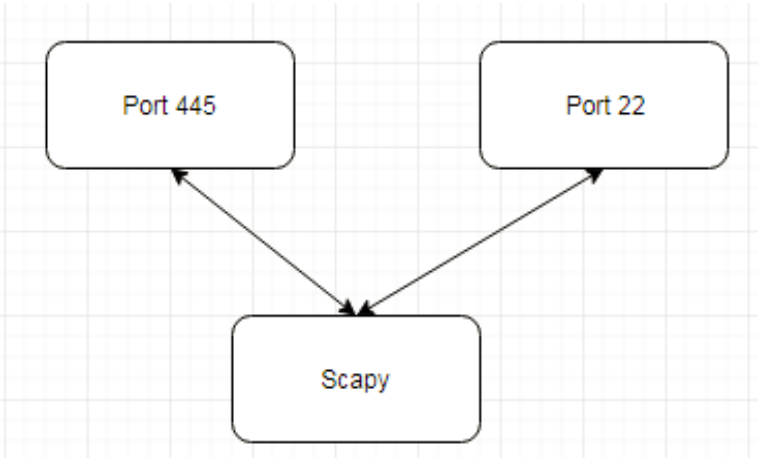

FIGURE 4.2: Scapy Port Scan.

Scapy will be used to scan each port, and adapt which packets will be sent too them as it picks up on traffic coming in, this traffic will more than likely be malicious as Honeypots are not too be touched by anybody other than the user and the attacker.

- The ability to spoof packets and send them to different ports.

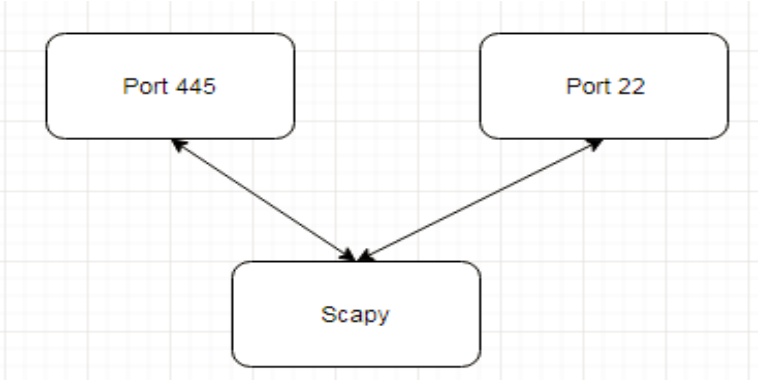

FIGURE 4.3: Scapy Packet Send.

Again using Scapy, packets will be crafted in Python and sent too specific ports. This will confuse the attacker to think that SSH or SMB are active so they can begin their attack.

- To log activity on these specific ports.

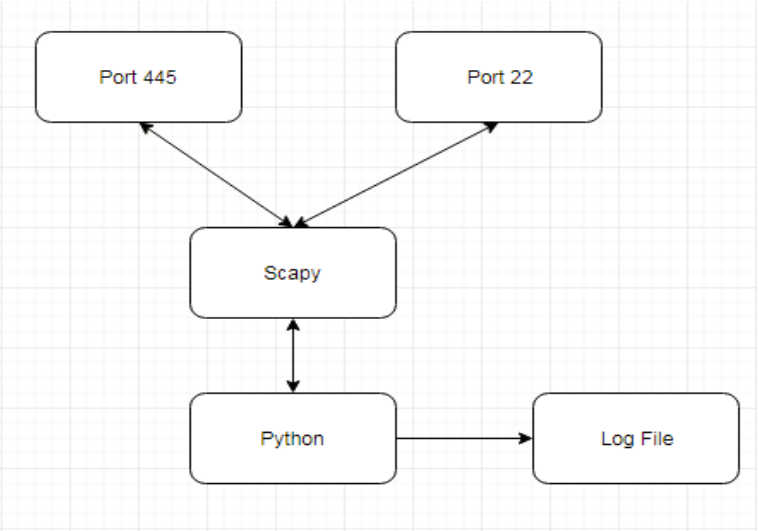

FIGURE 4.4: Logging activity with Scapy.

Python will use Scapy to scan ports for any activity, and record any suspicions in a Log File. The user of this Honeypot, can then enter the system and check the log file to discover who has been trying too access this system.

- To use packet capture tools on this system.

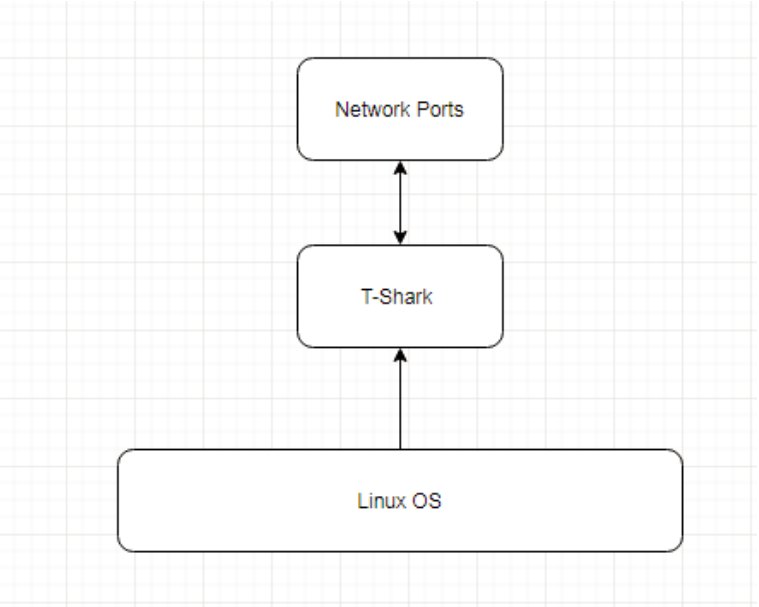

FIGURE 4.5: Logging activity with T-Shark.

T-Shark will be running in a terminal on the Linux OS which will scan every network port. Again, a user here can enter the system too open T-Shark to view any attempt to access the system.

- Use Linux on a virtual machine to host this system.

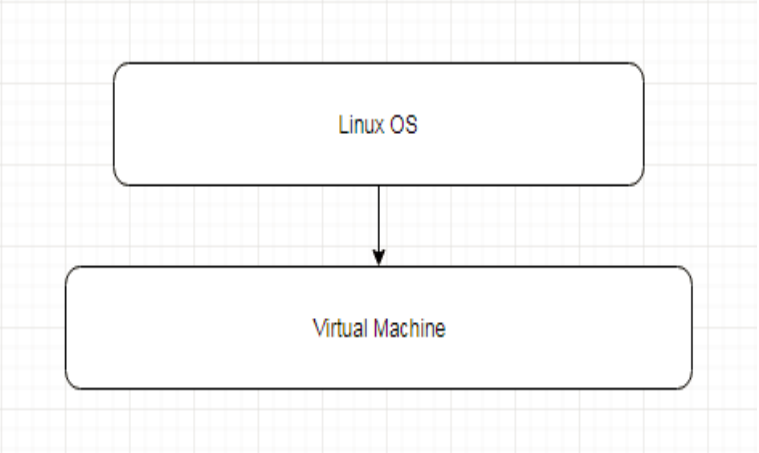

FIGURE 4.6: Linux OS on a Virtual Machine.

VMware will be used to hose a Linux OS. A user should have a knowledge on Linux and of VMware in order to use the Honeypot system.

- Use a Python Script which can interact with the ports.

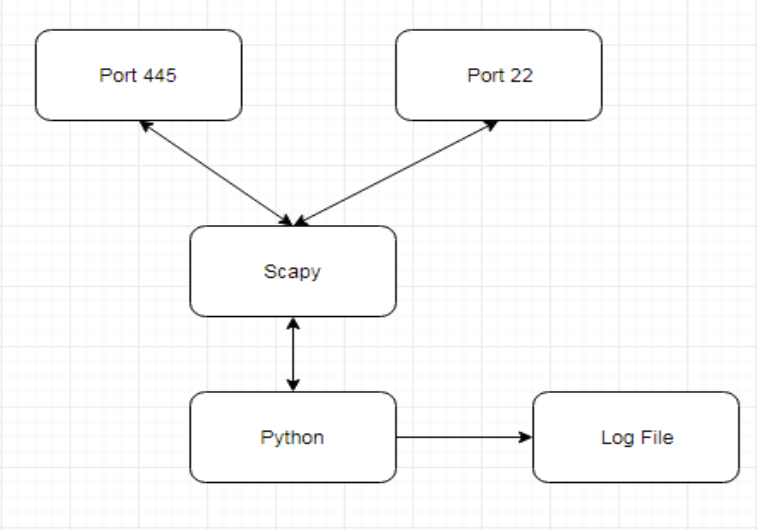

FIGURE 4.7: Python Script Interaction.

A python script will use Scapy to interact with the Ports. A user will be able to start or stop this Python Script, although it is discouraged that it be stopped.

4.2 Risk Assessment

In this section any risk which this project can hit will be identified. These are very important in any project, but I find they are especially important in a security based project, as people may rely on this system too fully work in order to not have attackers on their networks. The initial risk matrix classification for these risks seen below will be used.

TABLE 4.1: Initial risk matrix

Frequency/ Consequence	**1-Rare**	**2-Remote**	**3-Occasional**	**4-Probable**	**5-Frequent**
4-Fatal					
3-Critical					
2-Major					
1-Minor					

Risk of Software Failure - Remote/Critical

While great research has gone too figure out which Software would be the most reliable for this Intelligent Honeypot, there will always be a risk of it failing, these risks could be caused by anything from a simply update being ran on the machine, too crash failures from long run-time of these tools.

Here are 2 ways in which these risks can be avoided:

- **Backup Data:** Any data of importance on this Honeypot System should be backed up at regular intervals, if a software failure occurs after this an option too rollback will be available.
- **Change Management:** Any updates or new features made too this system should go through change management procedures. This will help to make sure that unnecessary updates or changes wont happen.

Risk of Hardware Failure - Occasional/Critical

Development of this project on only one physical machine could cause major implications if a hardware failure were too occur, especially if it was for example hard drive corruption which causes it.

Ways that this risk can be avoided include:

- **Backup Data:** Much like if their is a Software Failure, having backup data for a Hardware Failure will help if a user would like too move the system to a new machine.
- **Backup Hardware:** While this can be expensive for some users, having backup hardware, along with the backup data, helps to ensure that the Honeypot System will be running again in no time.

Risk of Availability - Remote/Fatal

This is a very important risk to tackle, as mentioned before it is specifically important in a security system. If the python script, or t-shark were too stop running for any reason, it would be giving the attacker a chance too enter into a system which they can use at their own free will.

Ways too avoid this particular risk are:

- **Test the system:** Run as many tests as possible on both t-shark and on the python script in order too make sure they work as expected.
- **Run on Start up:** Arranging a way for this project to run on start up will guarantee that even if the system crashes or reboots, the Honeypot script or T-Shark will run as soon as they are available.

Risk of not Meeting Requirements - Rare/Major

While this risk falls into the green area of the initial risk matrix, I find that this risk is important either way. This is because not meeting the requirements will have meant that I didn't research this project great enough and it will all have been for nothing.

Ways too avoid this particular risk are:

- **Plan Time:** Planning my time with accuracy and sticking too the plan will make sure I have the proper amount of time too meet these requirements.
- **Estimate Risks:** Exactly what this section of the research phase was for, estimating any risks will help me eliminate any risks encountered easier, making the project run smoother.

4.3 Methodology

As this is a Security based project, and people may rely heavily on security these days, correct research had too be carried out. To do this the following methodology was used:

- **Life Cycle:** During this research phase of the project a waterfall approach was used. Although I am not a big fan of this approach, it fit perfectly with this report as each phase in this report had too be completed before starting the next, which is what waterfall exactly is.
- **Resources:** The correct resources had too be found in order too backup facts and quotes in my research with references, I found that Cork Institute of Technologies Library was the most helpful source of reports, alongside Google Scholar which had millions of entries.
- **Skill-set:** Over the course of my studies in Cork Institute of Technology, I found that my research skills would be most valuable too Computer Science, this is because Computer Science is such an ever changing place and one would have too know how too keep on top of the knowledge. These exact skills were transferred over too this project as research was a very important step.

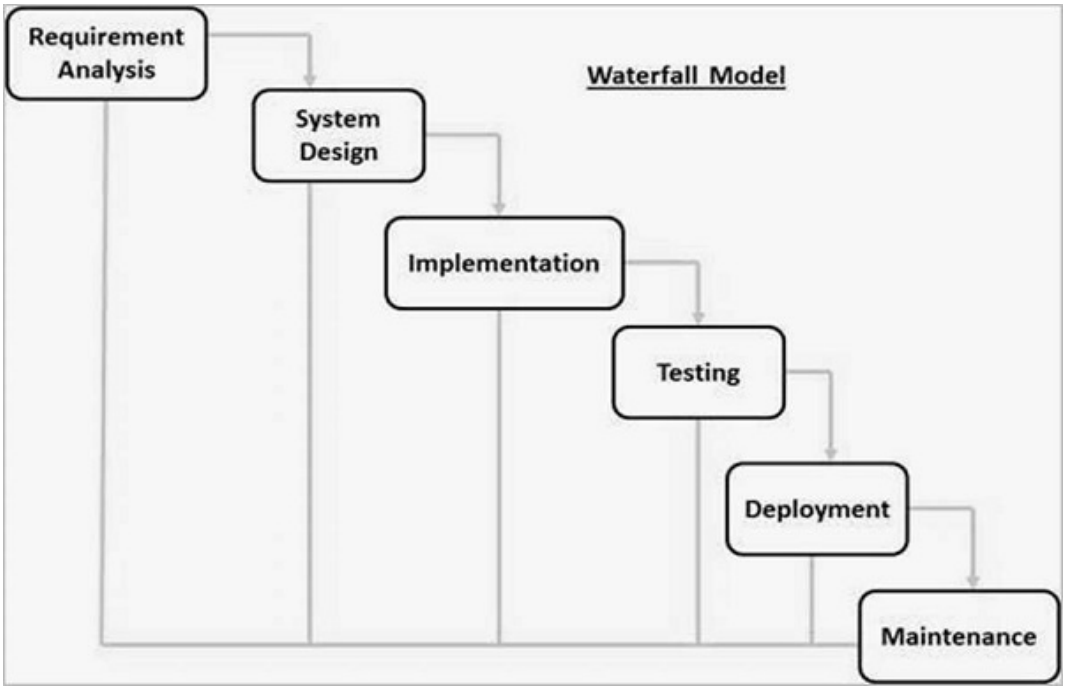

FIGURE 4.8: A Waterfall Life Cycle Approach.

Following the research phase of this project, I find that implementation phase will be just as important, and so the following methodology will be applied too the implementation phase of this Intelligent Honeypot:

- **Life Cycle:** During this research phase of the project a agile life cycle will be used as I feel this is the most effective approach to any form of software development. An agile approach, after going through each phase repeats itself in order too better any code and planning.
- **Resources:** For the implementation of this project I find that the most important resource will be this Research Document, this is due too the fact that the time line for this project, along with requirements will all be found in this document.
- **Skill-set:** A large skill-set from computer science will be applied too this phase of the project. These skills include my Script Programming skills, which will help with the Python script development, my Network Security skills, which will help with port scanning, and finally my Inter-Networking skills, which will help with both virtual machines and Linux usage.

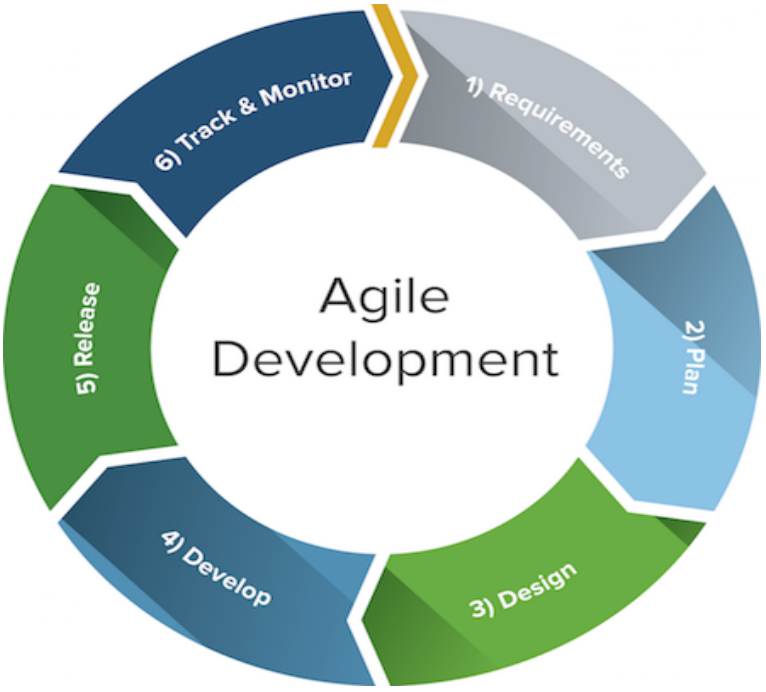

FIGURE 4.9: An Agile Life Cycle Approach.

4.4 Implementation Plan Schedule

During the implementation phase, alongside the Agile life cycle approach, I plan on using the Scrum framework for managing each process. I know this framework is more team based framework and this will be a self-managed project, but I feel like the sprint planning aspects of Scrum will help me too organize the project in the direction I want it too go in. In this section I will break down each sprint and tasks which I will assign too these sprints.

Sprint	Dates	Tasks
1	1st Feb - 15th Feb	1. Download and install VMware Workstation. 2. Download and install CentOS to VMware Workstation. 3. Ensure Python is installed on CentOS. 4. Install T-Shark for Linux on CentOS.
-	16th Feb - 22nd Feb	1. Sprint 1 retrospective. 2. Sprint 2 planning.
2	23rd Feb - 9th March	1. Follow new sprint plan tasks. 2. Install Scapy for Python. 3. Disable any utility running on Port 22. 4. Disable any utility running on Port 445.
-	10th March - 17th March	1. Sprint 2 retrospective. 2. Sprint 3 planning.

Sprint	Dates	Tasks
3	18th March - 1st April	1. Follow new sprint plan. 2. Begin Python script development. 3. Test T-Shark capabilities. 4. Run T-Shark.
-	2nd April - 9th April	1. Sprint 3 retrospective. 2. Sprint 4 planning.
4	10th April - 24th April	1. Follow new sprint plan tasks. 2. Fix any script bugs from previous sprint. 3. Finalize script. 4. Review T-Shark logs.
-	25th April - 2nd May	1. Sprint 4 retrospective. 2. Final Sprint planning.
Final	2nd May - 9th May	1. Follow new sprint plan tasks. 2. Test all features. 3. Fix any errors. 4. Review the project.

4.5 Evaluation

Evaluation is a very important step of each project, it is a plan on how to measure the success of the project. In order too evaluate this Honeypot project, the following evaluation steps will be carried out:

- **Functional Requirements:** Does this Honeypot meet each functional requirement described in chapter 3. This is important as these are the core features of the project.

- **Non-Functional Requirements:** Meeting these requirements are just as important as meeting the functional requirements as they are essentially the reliability of the project.

- **Objectives:** If all objectives enumerated in section 3.2 have been met, I would consider this progress has been a success.

- **Problem Solved:** If I have failed too meet the requirements, but still have found a way too solve the problem, I would still call this project a success, as the initial problem was the reason behind each chapter.

- **Feedback:** A great way too evaluate the success of a project is too present the project too a group of people and ask them how they would perceive its success.

- **Survey:** Another well known way of evaluating the success of a project is too allow users of a project to fill out a survey.

4.6 Prototype

This section is important to help envision how I perceive this project, however, as this Honeypot projects main aspect is python based, it is hard too show a wire-frame of how this will look. Here are some prototypes showing some of the tools of this project in use.

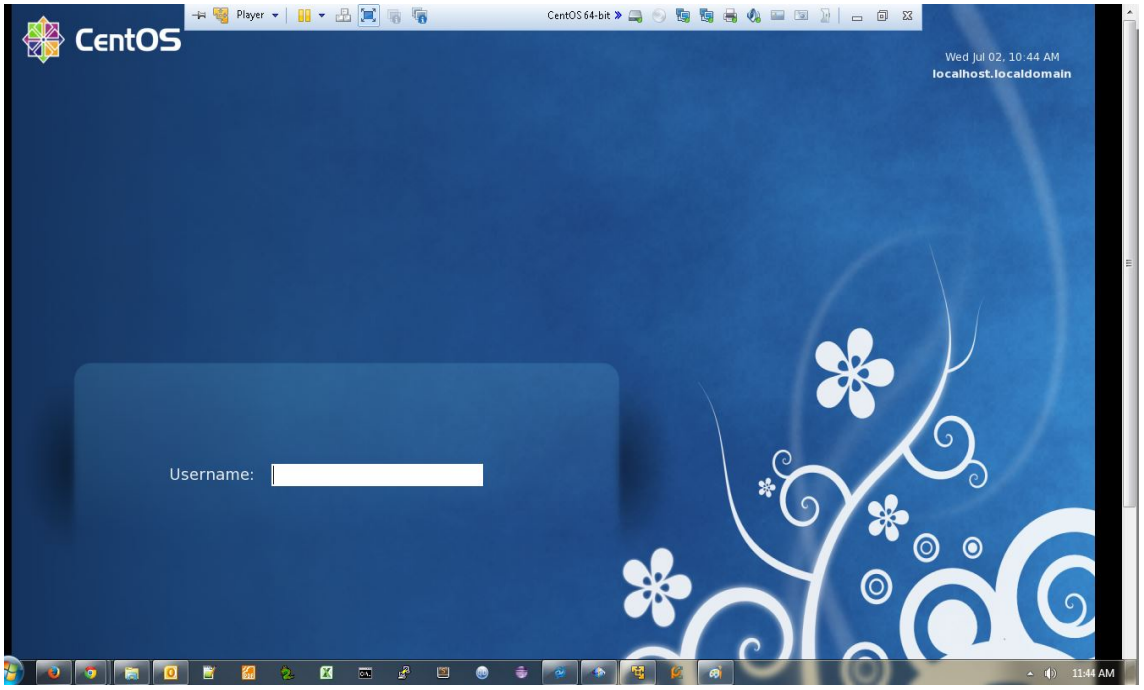

FIGURE 4.10: Running CentOS on a VM.

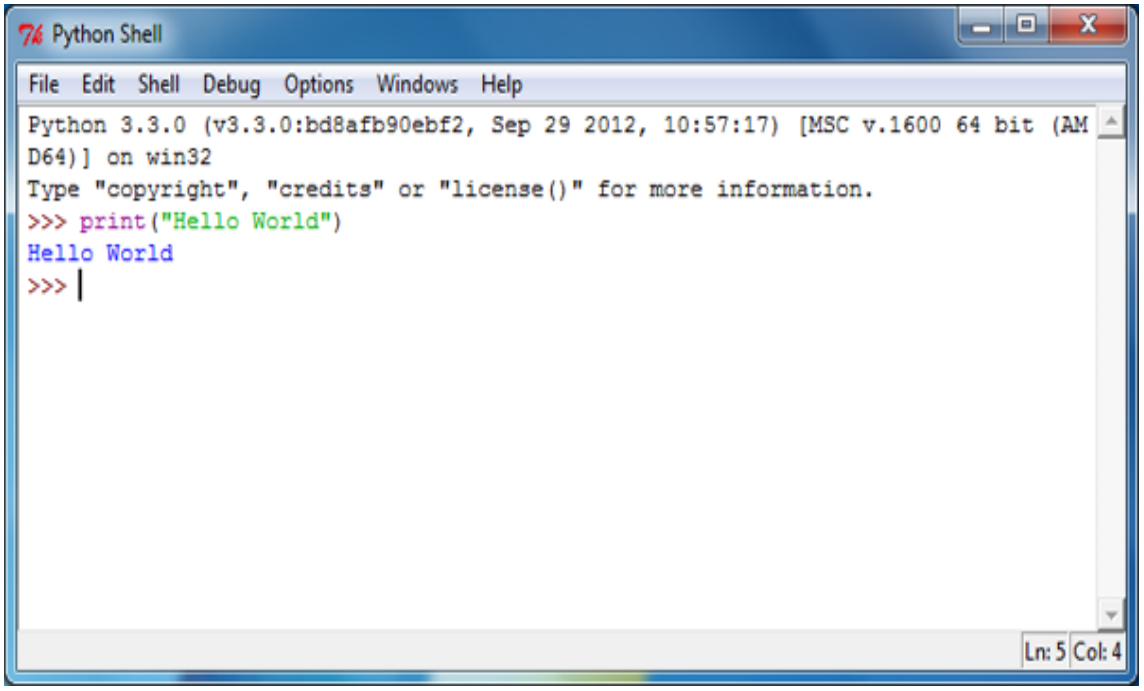

FIGURE 4.11: A Python hello-world Script.

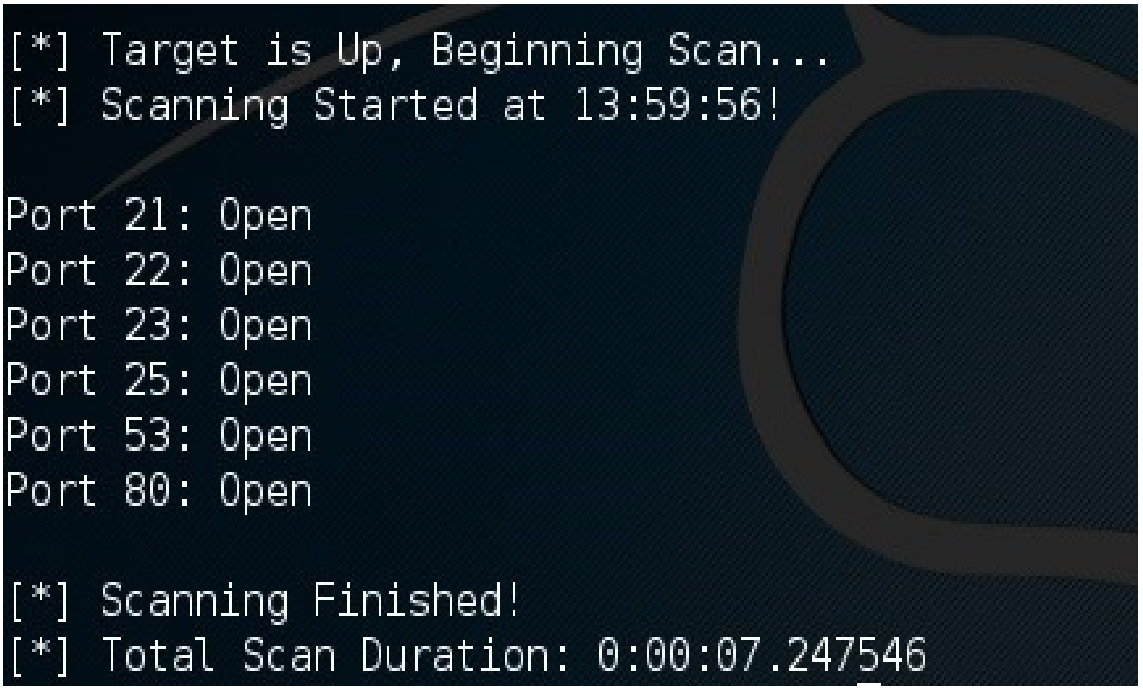

FIGURE 4.12: A Scapy Port Scan.

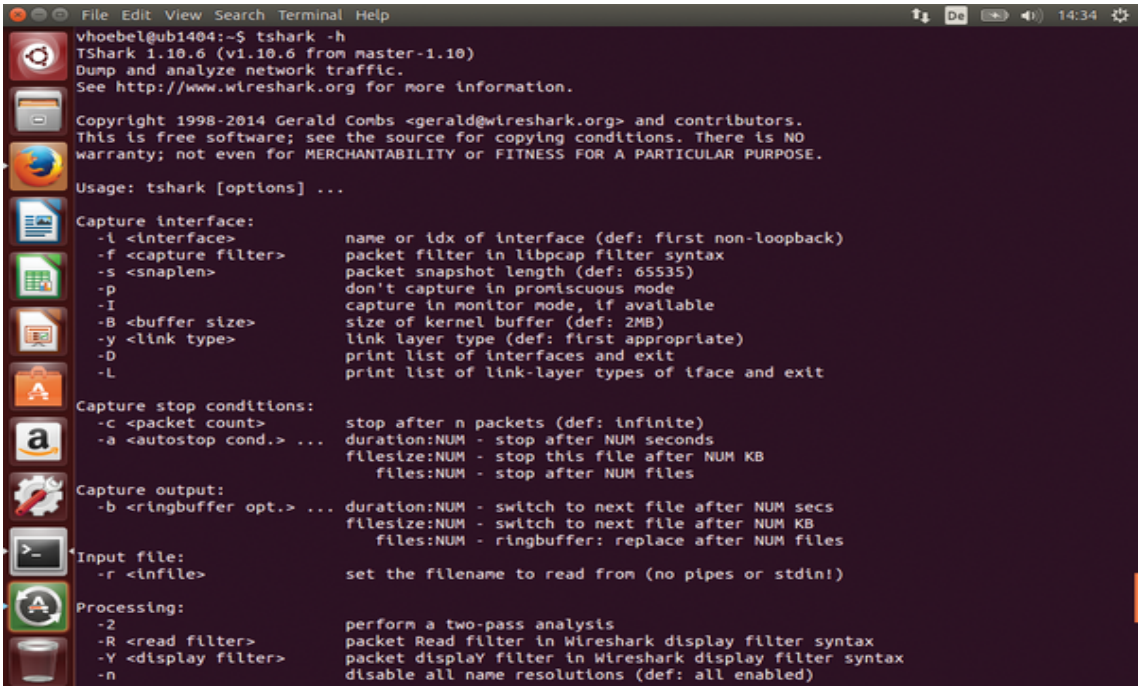

FIGURE 4.13: A Scapy Port Scan.

Chapter 5

Implementation Phase

5.1 Introduction

During the Implementation phase each tool used had a specific purpose. This section will describe the use of each tool, then it will discuss the development of the Honeypot itself, and finally the validation and verification of the project will be discussed.

5.2 Tools Used

The following tools were used for the development of the Honeypot:

1. VMWare: Was used in order to host the CentOS and Kali virtual machines.
2. CentOS: Was chosen to act as the OS to host the Honeypot on.
3. Python: Used to write scripts for the Honeypot.
4. Scapy: This was used when crafting packets the spoof SSH or SMB.
5. T-Shark: Chosen to record activity on ports 22 or 445.
6. Slack: This messaging app was used when creating a chatbot to alert on the Honeypots attackers.
7. Apache: This was used to Host a website on the Honeypot to lure in attackers.
8. Kali: This was used for testing purposes.

5.3 Development

1. Installing VMware Workstation:

- Navigate to https://www.vmware.com/products/workstation-pro.html and purchase VMWare Workstation.
- After purchasing, download the installer provided on the website.
- Run the installer and follow the install wizard to configure it for the machine its running on.
- VMware is now installed.

2. Installing a CentOS Virtual Machine:

- Navigate to https://www.osboxes.org/centos/#centos-7-x-info
- Download the VMware version of CentOS 7-1708.
- When this file is downloaded, extract it using WinRaR or a similar tool.
- Open VMware Workstation, and click file, then open virtual machine.
- Navigate to the folder which was extracted earlier and open the virtual machine inside.
- CentOS is now installed.

3. Installing a Kali Linux VM:

- Navigate to https://www.offensive-security.com/kali-linux-vm-vmware-virtualbox-hyperv-image-download/
- Download the VMware version of Kali 64 Bit.
- When this file is downloaded, extract it using WinRaR or a similar tool.
- Open VMware Workstation, and click file, then open virtual machine.
- Navigate to the folder which was extracted earlier and open the virtual machine inside.
- Kali is now installed.

4. Installing ScaPy on the CentOS VM:

- Open the command line on the CentOS VM.
- Enter the following command to enable the EPEL repo:

  sudo yum install epel-release
- Now that the EPEL repo is enabled, it can be used to install pip using the following command:

  sudo yum -y install python-pip
- Finally to install ScaPy enter the following command:

  sudo pip install scapy

5. Installing T-shark on the CentOS VM:

- To install T-Shark, the installation steps for installing Wireshark must be followed as it is packaged together.
- To install Wireshark, simply enter the following command:

  sudo yum install wireshark
- Wireshark and T-shark are now installed.

6. Disabled services or packets using ports 22 or 445 on the CentOS VM.

- To remove SSH from the Virtual Machine, which uses port 22, enter the following in the command line:

  sudo yum remove openssh-server
- As this is a Linux based machine, microsoft-ds wont be installed on it. This means port 445 isn't already in use.

7. A basic python script has been created.

- From the command line open a vim text editor for creating the python script:

  vim capture.py

- Add the following lines of code inside this file for a basic hello world script:

  #!/usr/local/bin/python2

  print("hello world")

- To close VIM press escape, then type :wq

- To run the python script, in the command line type:

  python capture.py

8. T-shark can be called from a python script.

- Open the python script again in VIM:

  vim capture.py

- Add the following line near the top of the python file, this is importing the OS library which will be used to call tshark on the command line from python:

  import os

- Add the following line near the bottom of the script, this calls tshark on the command line from the python script:

  os.system("tshark -i any -w honey.pcap -q ")

- When this script is ran, it should now start a tshark capture and log the files in honey.pcap.

9. Making the script capable of logging any activities on ports 22 and 445.

- To log the activity on only port 22 and port 445, edit the same line as above to look like the following:

  os.system("tshark \"tcp port 22\" or \"tcp port 445\" -i any -w honey.pcap -q ")

- The script should now log activity only only ports 445 and 22.

10. Manipulating the script to only show every port as open.

- In order to show each port as open using ScaPy, the following libraries are imported into the code near the top:

  from scapy import all

  from scapy.all import *

- Next, in order to craft scapy packets to make the ports appear as open, define a method and add the following lines to the code (remember to tab the code):

  os.system("iptables -A OUTPUT -p tcp -o eth0 –sport 1:65535 –tcp-flags RST RST -j DROP")

  def packet(pkt):

  if pkt[TCP].flags == 2:

  print('SYN packet detected port : ' + str(pkt[TCP].sport) + ' from IP Src : ' + pkt[IP].src)

  send(IP(dst=pkt[IP].src, src=pkt[IP].dst)/TCP(dport=pkt[TCP].sport,

  sport=pkt[TCP].dport,

  ack=pkt[TCP].seq + 1, flags='SA'))

  sniff(iface="eth0", prn=packet, filter="tcp[0xd]&18=2",count=100)

  os.system("iptables -D OUTPUT -p tcp -o eth0 –sport 1:65535 –tcp-flags RST RST -j DROP")

- When this python script is ran, it should show each port as open, however after each port is scanned they will remain closed again, so I added a while loop around the script to keep it running continuously. To do this add this line before the packet crafting section of the script and tab the rest of the section out :

  while True:

11. Manipulating the script to only show ports 445 and 22 as open

- Instead of every port appearing as open, I only want ports 445 and port 22 to appear as open.
- To do this an if else function is needed inside the python script, these can be seen here in green:

```
os.system("iptables -A OUTPUT -p tcp -o eth0 –sport 1:65535 –tcp-flags RST RST -j DROP")
def packet(pkt):
if pkt[TCP].flags == 2:
if(str(pkt[TCP].dport)) == "22":
print('SYN packet detected port : ' + str(pkt[TCP].sport) + ' from IP Src : ' + pkt[IP].src)
send(IP(dst=pkt[IP].src, src=pkt[IP].dst)/TCP(dport=pkt[TCP].sport,
sport=pkt[TCP].dport,
ack=pkt[TCP].seq + 1, flags='SA'))
sniff(iface="eth0", prn=packet, filter="tcp[0xd]&18=2",count=100)
os.system("iptables -D OUTPUT -p tcp -o eth0 –sport 1:65535 –tcp-flags RST RST -j DROP")
if pkt[TCP].flags == 2:
elif(str(pkt[TCP].dport)) == "445":
print('SYN packet detected port : ' + str(pkt[TCP].sport) + ' from IP Src : ' + pkt[IP].src)
send(IP(dst=pkt[IP].src, src=pkt[IP].dst)/TCP(dport=pkt[TCP].sport,
sport=pkt[TCP].dport,
ack=pkt[TCP].seq + 1, flags='SA'))
sniff(iface="eth0", prn=packet, filter="tcp[0xd]&18=2",count=100)
os.system("iptables -D OUTPUT -p tcp -o eth0 –sport 1:65535 –tcp-flags RST RST -j DROP")
```

- Only ports 22 and 445 will appear as open now.

12. Running the script when the OS boots.

- In order to do this Crontab is used.
- Open a Crontab editor in the command line using:

  crontab -e
- Add the following line to the crontab to make the script run on reboot:

  @reboot python /home/osboxes/script/capture.py
- The script will now run on startup for the root user.

13. Setting up a Slack Channel for message alerting.

- Navigate to slack.com in a web browser.
- Click get started.
- Click create a new workspace.
- Log in or register for a Slack account.
- Enter a full name and display name for the user which will appear on the Slack channel.
- Set a password.
- Tell Slack about the Team which will be using the Channel.
- Enter a name for the Slack Channel.
- Finally, hit create workspace and the channel will be created.

14. Configure a Chatbot on Slack for honeypot alerts.

- Log in to the Slack Channel previously set up.
- On the left hand pane, hit the "plus" button next to Apps.
- In the Search bar that appears, search for Webhook.
- Find the App named Incoming WebHook from the list and hit install next to the App.
- Give the WebHook a name which will appear in the Channel, and upload an icon for it.

- Record the WebHook URL given as it will be used later.
- The Chatbot is now configured.

15. A script has been created to send the Honeypot alerts to slack.

- Create a new python script called notifyslack.py using:

  vim notifyslack.py

- At the top of the python script, the following libraries are imported:

  from collections import Counter

  import os

  import slackweb

- Next the Slack WebHook link from earlier is entered in a variable in the script:

  slack =slackweb.Slack(url="ENTER WEBHOOK URL HERE")

- Next towards the end of the script, but above the main method, a method for getting a list of IP's from the pcap file and sorting them by most common should be made:

  def getIPs():

- The first step of this method is too open the pcap file and read each ip source address found in it, into a txt file:

  os.system("tshark -r .config/honey.pcap -T fields -e ip.src → *.config/honey.txt*")

- After the txt file is created, the IP address of the local machine will be read into a variable which will be used later:

  localIP = socket.gethostbyname(socket.gethostname())

- Then a IP array is created for reading in each IP address from the txt file, while ignoring the IP address from the local machine:

  ip_array=[]

  with open('.config/honey.txt') as ip_file:

  for ip in ip_file:

  ip = ip.strip()

  if ip != str(localIP):

  ip_array.append(ip)

- This array is then sorted into another array, which sorts it by most common and records the count of each repeated IP address:

  count_ips=(Counter(ip_array).most_common())

- Next a different method for sending the Slack alerts is created:

  def notification(count_ips):

- Arrays for both the top ip addresses and the number of attacks are then defined:

  def notification(count_ips):

  TopIP=[]

  NumberAttack=[]

  for ip, attacks in count_ips:

  TopIP.append(ip)

  NumberAttack.append(attacks)

- A slack alert will then be generated inside the same method for the top 3 or less attacks:

  if not TopIP:

  attachment = {"color": "#008000",

  "title": "No IP's attacked today"}

  else:

  if len(TopIP) == 1:

  attachment = {"color": "#FFFF00",

  "title": "Top Honeypot Attacker IP:",

  "text" : "1." + str(TopIP[0]) + " - " + str(NumberAttack[0]) + " attacks"}

- Repeat the second part of the if statement, adding on extra parts of the TopIP and NumberAttack arrays for the alert on 2 IP's and alert on 3 IP's.

- Send the slack alerts using the following lines of code:

  attachments.append(attachment)

  slack.notify(attachments=attachments)

- The alert script should now send alerts showing the top 3 or less attackers.

16. Hosted a website on the Honeypot.

- From the terminal, enter the following command to install Apache:

  sudo yum install httpd

- Change directory to the httpd files where the website will be stored:

  cd /var/www/html/

- In a web browser, navigate too https://github.com/dogecoin/dogecoin.com

- Clone the files from this repository into the httpd files and extract them.

- Restart Apache from the command line using:

  service httpd restart

- A website should now be hosted on the VM, which will be used to attract attackers too this Honeypot.

5.4 Validation

The following is a list of requirements set for this Honeypot and how they have been validated:

1. Ability to adapt to different ports as they are being scanned:

This was validated by running NMAP scans from the Kali Machine, onto the Honeypot. If the open ports appeared different in each scan it worked.

2. The ability to spoof packets and send them to different ports:

This was validated by running a NMAP scan, first when the Honeypot script wasn't running, this will show there is no services on the machine. Then the NMAP scan is ran again at the same time that the Honeypot script is, this time it will show that port 22 or port 445 are open, showing that SSH and SMB are being spoofed on the script.

3. To log activity on these specific ports.

With the Honeypot script running, again run an nmap scan of the CentOS machine using Kali. Now open the honey.pcap file and it should be populated with packets captured from the Kali machine.

4. Use Linux on a virtual machine to host this system.

Run a NMAP -O on the CentOS Machine without the script running to verify that it is CentOS Linux.

5. A script which can interact with the Honeypot.

Run both the capture.py script and notifyslack.py scripts to validate that they both work.

6. The script on this system would have too be highly available with no downtime.

Test the cron job by rebooting the CentOS VM, the script should run on startup again.

7. The system will also have to be highly reliable.

Research was done before choosing CentOS as the system to host the virtual machine and it was deemed highly reliable.

8. The Honeypot will have too have minimum capacity.

The honeypot only records activity on ports 22 and 445, instead of recording a lot of wasted data from all of the ports, this can be validated by opening up the capture.py script and looking at the line which begins the packet capture.

9. The response time of this system to an attacker would have to be near instant.

This can be validated with yet another nmap scan on the CentOS machine, while the capture.py script is running. As the nmap scan is running, one can see the script printing out messages reacting to the scan.

5.5 Usage

This section will familiarize a user with how to use this Honeypot.

1. Booting up CentOS:

- Open VMWare Workstation on your machine.
- Double click on the honeypot VM to open it in VMWare.
- Click power on this virtual machine.
- When the login screen appears, login with user osboxes.org and password osboxes.org
- The CentOS Machine should now be booted up.

2. Booting up Kali:

- Open VMWare Workstation on your machine.
- Double click on the Kali VM to open it in VMWare.
- Click power on this virtual machine.
- When the login screen appears, login with user root and password toor
- The Kali Machine should now be booted up.

3. Running the main Honeypot script:

- Right click on the desktop in the CentOS VM.
- Click open terminal from the list of options given.
- When the terminal appears, type "cd script" to change to the directory which holds the scripts.
- Type "ls -la" once inside this directory to see all available files.
- To run the capture.py script type "sudo python capture.py" and type the password osboxes.org
- The script should now be running.

3. Running an nmap scan of the Honeypot:

- Right click on the desktop in the CentOS VM.
- Click open terminal from the list of options given.
- When the terminal appears, "ifconfig" this should give the IP address of the CentOS Machine.
- Switch over to the Kali Machine now and right click on the desktop, then click open terminal.
- Once the terminal is open, type "nmap -sS IPADDRESSHERE" into the terminal, replacing IPADDRESSHERE with the CentOS VM's IP.
- The nmap scan should now run.

4. Running the Slack Alert script:

- Right click on the desktop in the CentOS VM.
- Click open terminal from the list of options given.
- When the terminal appears, type "cd script" to change to the directory which holds the scripts.
- Type "ls -la" once inside this directory to see all available files.
- To run the notifyslack.py script type "sudo python notifyslack.py" and type the password osboxes.org
- The script should now run, and send a message too https://smarthoneypot.slack.com notifying the user of the top attackers.

Chapter 6

Conclusions and Future Work

6.1 Discussion

One of the largest problems encountered during the research phase was the timing, had I more time I would've looked into more features which I could have added to the Honeypot. I also felt like I didn't have much time to review these chapters thoroughly as many other real world distractions got in the way, throwing me of my schedule for completing this investigation. This will greatly influence my next phase of the project as I will be more time cautious when implementing the Honeypot and I will try my best too follow the time line outlined in chapter four.

Other problems in which I encountered during the research phase of this project involved choosing the right methodology in order to conduct my research. Eventually I conquered this problem and found a rhythm for researching. This will influence my implementation phase as I now know that methodology is very important in a project.

During the implementation phase of the project most of the problems I encountered were involving bugs in my code, but these problems were all solved during the test phases of each sprint.

Another issue I found with my implementation phase was again the timing for the schedule. I managed to complete my entire original idea for the Honeypot quite early on in the phase which left me to try and come up with new idea to expand upon it each week, which all should've been done in the research phase.

Overall I found that every part of the research phase was just as important, if not more important that the implementation phase for this project.

6.2 Conclusion

Overall, during the background chapter of this research, I was very happy with its outcome. I found that there was a very large amount of Research Papers and articles already done about Honeypots which was able too help me answer any questions about their background easily.

The problem chapter (An Intelligent Honeypot) I found too be easily completed as well, this is because with the background research on Honeypots already completed, I was able too look at which problems still exist, and how my Honeypot would be able too solve this problem.

Solution approach was one of the harder chapters then. While I already knew about the project and what I wanted from it, I still wasn't sure how I was going too get there. Luckily I figured it out eventually and I was quite happy with my solution on how too implement this Honeypot.

During the implementation phase, I found that the original concept for this Honeypot was easily completed as there was many resources for python and scapy available online. This left me with a lot of spare time to fine tune my code, as well as add new features to the Honeypot. I was happy with this in the sense that I was way ahead of schedule, but I was also a bit disappointed that I hadn't foreseen this coming so that I could plan the extra work more.

6.3 Future Work

During the implementation phase, I was lucky enough too find extra time to add in some future work which wasn't part of my original concept for the Honeypot. This involved two new features, the first being sending Slack alerts to notify users of the Top IP addresses attacking the Honeypot.

The second new feature, not listed in the original concept, which I added was a website which was hosted on the Honeypot. This would be used as a lure to try and attract attackers to this Honeypot in the first place. I chose a cryptocurrency website to host on it, as attacks on cryptocurrencies are becoming more and more common.

If I had more time too research during this phase of the project I would've looked in to a lot more features for my Intelligent Honeypot. One of the features which always stood out too me which I wish I had more time for was too implement a form of AI analytics into this Honeypot,therefore making it more intelligent, while providing more insight into the attackers in a user friendly way too the user of this system.

Another feature I would've like too implement if I was given more time too research this project, would be too monitor for SQL injections on the website which was hosted on the Honeypot. This would've given more of an insight into the attacker and what methods they've used to attack the Honeypot.

Bibliography

[1] "Number of internet users worldwide," 2015 - 2017. [Online]. Available: https://www.statista.com/statistics/273018/number-of-internet-users-worldwide/

[2] Wikipedia, "Wannacry ransomware," 2017. [Online]. Available: https://en.wikipedia.org/wiki/WannaCry_ransomware_attack

[3] E. Wiktionary, "computer-security," 2017. [Online]. Available: http://www.yourdictionary.com/computer-security#wpRVbMA0PvYGv1gm.99

[4] Techopedia, "Internet privacy," 2017. [Online]. Available: https://www.techopedia.com/definition/24954/internet-privacy

[5] Computer and N. Security, "Honeypots," 2015. [Online]. Available: https://www.youtube.com/watch?v=fQqWe8br2Gw

[6] E. Peter, "A practical guide to honeypots," 2008. [Online]. Available: http://www.cse.wustl.edu/~jain/cse571-09/ftp/honey/

[7] "What are the ethical issues concerning honeypots?" 2017. [Online]. Available: https://ethics.csc.ncsu.edu/abuse/hacking/honeypots/study.php

[8] L. Spitzner, "Honeypots: Tracking hackers," 2002. [Online]. Available: http://www.it-docs.net/ddata/792.pdf

[9] S. Exchange, "Information security," 2017. [Online]. Available: https://security.stackexchange.com/

[10] R. A. Grimes, "Hacking the hacker," 2017. [Online]. Available: http://onlinelibrary.wiley.com/book/10.1002/9781119396260

[11] V. Ryan, "Honeypots," 2017. [Online]. Available: https://citbb.blackboard.com/webapps/blackboard/execute/content/file?cmd=view&content_id=_491604_1&course_id=_61889_1&framesetWrapped=true

[12] R. Joshi, "Honeypots: A new paradigm to information security," 2017. [Online]. Available: https://books.google.ie/books?id=c_rRBQAAQBAJ&printsec=frontcover&dq=honeypots+in+information+security&hl=en&sa=X&ved=0ahUKEwjjsMP8wrnYAhVKCMAKHWDGBd4Q6AEIKDAA#v=onepage&q=honeypots%20in%20information%20security&f=false

[13] Wikipedia, "Information security," 2017. [Online]. Available: https://en.wikipedia.org/wiki/Information_security

[14] WikiPedia, "Cyber security," 2017. [Online]. Available: https://en.wikipedia.org/wiki/Computer_security

[15] L. Morgan, "List of data breaches and cyber attacks in october 2017," 2017. [Online]. Available: https://www.itgovernance.co.uk/blog/list-of-data-breaches-and-cyber-attacks-in-october-2017-55-million-records-leaked/

[16] M. L. Bringer, "A survey: Recent advances and future trends in honeypot research," 2012. [Online]. Available: http://www.mecs-press.org/ijcnis/ijcnis-v4-n10/IJCNIS-V4-N10-7.pdf

[17] WikiPedia, "Honeypot computing," 2017. [Online]. Available: https://en.wikipedia.org/wiki/Honeypot_(computing)#Honey_nets

[18] R. Tanara, "The honeynet project," 2017. [Online]. Available: https://www.honeynet.org/

[19] C. Liu, "Investigating network security through firewall utilities," 2017. [Online]. Available: https://www.theseus.fi/bitstream/handle/10024/129290/INVESTIGATING%20NETWORK%20SECURITY%20THROUGH%20FIREWALL%20UTILITIES.pdf?sequence=1&isAllowed=y

[20] C. Rowland, "Intrusion detection system," 2002. [Online]. Available: https://www.google.com/patents/US6405318

[21] Wireshark, "What is a network protocol analyzer?" 2017. [Online]. Available: http://wireshark.com/

[22] L. Harbaugh, "The pros and cons of using virtual desktop infrastructure," 2012. [Online]. Available: https://www.pcworld.com/article/252314/the_pros_and_cons_of_using_virtual_desktop_infrastructure.html

[23] S. Larson, "The hacks that left us exposed in 2017," 2017. [Online]. Available: http://money.cnn.com/2017/12/18/technology/biggest-cyberattacks-of-the-year/index.html

[24] L. Morgan, "List of data breaches and cyber attacks in 2015," 2015. [Online]. Available: https://www.itgovernance.co.uk/blog/list-of-data-breaches-and-cyber-attacks-in-2015-over-275-million-leaked-records/

[25] ——, "List of data breaches and cyber attacks in 2016," 2016. [Online]. Available: https://www.itgovernance.co.uk/blog/list-of-data-breaches-and-cyber-attacks-in-2016-1-6-billion-records-leaked/

[26] A. Smith, "Cyber security statistics 2017: Data breaches and cyber attacks," 2018. [Online]. Available: https://itsecuritycentral.teramind.co/2018/01/03/cyber-security-statistics-2017-data-breaches-and-cyber-attacks/

[27] M. Giles, "Six cyber threats to really worry about in 2018," 2018. [Online]. Available: https://www.technologyreview.com/s/609641/six-cyber-threats-to-really-worry-about-in-2018/

Appendix A

Code Snippets

A python hello world script:

```
print("Hello World")
```

Python using Scapy to check if a TCP port is open:

```
from scapy.all import *
conf.verb = 0
p = IP(dst="github.com")/TCP()
r = sr1(p)
print r.summary()
```

The capture.py script:

```
#!/usr/local/bin/python2

import os

from scapy import all

from scapy.all import *

def packetcraft():

print ("Begining packet crafting:")

while True:

os.system("iptables -A OUTPUT -p tcp -o eth0 –sport 1:65535 –tcp-flags RST RST -j
DROP")

def packet(pkt):

if pkt[TCP].flags == 2:

if(str(pkt[TCP].dport)) == "22":

print('SYN packet detected port : ' + str(pkt[TCP].sport) + ' from IP Src : ' +
pkt[IP].src)

send(IP(dst=pkt[IP].src, src=pkt[IP].dst)/TCP(dport=pkt[TCP].sport, sport=pkt[TCP].dport,
ack=pkt[TCP].seq + 1, flags='SA'))

elif(str(pkt[TCP].dport)) == "445":

print('SYN packet detected port : ' + str(pkt[TCP].sport) + ' from IP Src : ' +
pkt[IP].src)

send(IP(dst=pkt[IP].src, src=pkt[IP].dst)/TCP(dport=pkt[TCP].sport, sport=pkt[TCP].dport,ack=pk
+ 1, flags='SA'))

sniff(iface=conf.iface, prn=packet, filter="tcp[0xd]18=2",count=100)

os.system("iptables -D OUTPUT -p tcp -o eth0 –sport 1:65535 –tcp-flags RST RST -j
DROP")
```

```
def logports():

print ("Starting T-Shark capture on ports 22 and 445:")

os.system("tshark \"tcpport22\"or\"tcpport445\" -iany -w.config/honey.pcap -q")

print ("T-Shark is now running")

def main():

print("Intelligent Honeypot System")

logports()

packetcraft()

if __name__ == "__main__":

try:

main()

except KeyboardInterrupt:

print "Exiting as user request..."
```

The notifyslack.py script:

```
from collections import Counter

import os

import slackweb

import socket

slack = slackweb.Slack(url="ENTER WEBHOOK URL HERE")

def notification(count_ips):

TopIP=[]

NumberAttack=[]

for ip, attacks in count_ips:

TopIP.append(ip)

NumberAttack.append(attacks)

attachments = []

if not TopIP:

attachment = "color": "#008000",

"title": "No IP's attacked today"

else:

if len(TopIP) == 3:

attachment = "color": "#FF0000",

"title": "Top 3 Honeypot Attacker IP's:",

"text" : "1." + str(TopIP[0]) + " - " +

str(NumberAttack[0]) + " attacks" + "2." +

str(TopIP[1]) + " - " + str(NumberAttack[1]) +

" attacks" + "3." + str(TopIP[2]) + " - " +

str(NumberAttack[2]) + " attacks"
```

elif len(TopIP) == 2:

attachment = "color": "#FFA500",

"title": "Top 2 Honeypot Attacker IP's:",

"text" : "1." + str(TopIP[0]) + " - " + str(NumberAttack[0]) +

"attacks" + "2." + str(TopIP[1])

+ " - " + str(NumberAttack[1]) + " attacks"

elif len(TopIP) == 1:

attachment = "color": "#FFFF00",

"title": "Top Honeypot Attacker IP:",

"text" : "1." + str(TopIP[0]) + " - " + str(NumberAttack[0]) + " attacks"

attachments.append(attachment)

slack.notify(attachments=attachments)

def getIPs():

os.system("tshark -r .config/honey.pcap -T fields -e ip.src → [].*config/honey.txt*")

localIP = socket.gethostbyname(socket.gethostname())

ip_array=[]

with open('.config/honey.txt') as ip_file:

for ip in ip_file:

ip = ip.strip()

if ip != str(localIP):

ip_array.append(ip)

count_ips=(Counter(ip_array).most_common())

print count_ips

return count_ips

```
def main():

count_ips=getIPs()

notification(count_ips)

if __name__ == "__main__":

try:

main()

except KeyboardInterrupt:

print "Exiting as user request..."
```