DEPLOYMENT OF AN HONEYPOT SYSTEM FOR CYBER-ATTACK ANALYSIS AND DETECTION

ALOBA SUAD OLAPEJU
21/SCI01/075

&

NTEWO TOYOBONG SAMUEL
20/SCI01/081

A PROJECT REPORT SUBMITTED TO COMPUTER SCIENCE PROGRAMME, DEPARTMENT OF MATHEMATICAL AND PHYSICAL SCIENCES, COLLEGE OF SCIENCES, AFE BABALOLA UNIVERSITY, ADO-EKITI, NIGERIA.

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF BACHELOR OF SCIENCE(B.Sc.) DEGREE IN COMPUTER SCIENCE

MAY 2024

DECLARATION

Aloba Suad Olapeju and Ntewo Toyobong Samuel hereby declare that this project was conducted by us and is a correct record of our research work. It has not been presented in any previous application for any degree of this or any other University. All citations and sources of information are clearly acknowledged by means of references.

Signature…….……..... Date……
Aloba Suad Olapeju

Signature…….……..... Date……
Ntewo Toyobong Samuel

CERTIFICATION

This is to certify that Ntewo Toyobong Samuel and Aloba Suad Olapeju with matriculation number 20/SCI01/081 and 21/SCI01/075 in the Department of Mathematical and Physical Sciences, Afe Babalola University, Ado-Ekiti (ABUAD), and that it was carried out under our supervision for the award of the Computer Science degree program.

……. …….

Mr. Oluwafemi.A. Sanya Date

Supervisor

……. …….

Dr Fumilayo.H Oyelami Date

Head of Department

DEDICATION

This project is dedicated to Almighty God for His unwavering guidance and strength, all throughout this project work and our stay in this university. And to our parents, supervisor, siblings, and friends for their unwavering support and boundless love and prayers.

ACKNOWLEDGEMENT

All honor and gratitude go to God, for always being our guide, stronghold and for His endless favor, mercy and grace upon me during the course of our study in Afe Babalola University.

With heartfelt gratitude, we acknowledge our supervisor and lecturer, Mr. Oluwafemi.A. Sanya who always helped and guided us. For his great support, words of encouragement and contribution to this study till the completion of the project, We are very thankful.

I acknowledge the Provost of the program, Prof P.Okiki, for his beneficial support in honor of his contributions to our academics throughout our time in Afe Babalola University. We also appreciate the Head of Department of Mathematical and Physical Sciences, Dr Fumilayo.H Oyelami, for her academic guidance and moral support during my stay in Afe Babalola University.

We appreciate our level advisor, Dr. Stephen Oyebami, for his support, always lending a helping hand throughout our stay in Afe Babalola University. We acknowledge all our lecturers who strived to ensure we get the best education and for their endless motivation. We take this opportunity to express our heartfelt thanks to our parents, siblings and friends for their immense support and constant encouragement.

May God bless you. Amen.

TABLE OF CONTENT

LIST OF TABLES AND FIGURES

ABSTRACT

With the exponential growth of computer and internet technologies, network security has become an increasingly pressing concern. Despite existing measures such as Intrusion Detection Systems (IDS) and firewalls, cyber attackers continue to exploit vulnerabilities, posing significant threats to organizational data and infrastructure. Depending solely on conventional defenses such as IDSs and firewalls is insufficient for identifying emerging attack patterns capable of infiltrating network resources *(Marcin et al; 2016)*. The aim of this project is to deploy a honeypot in a controlled environment, leveraging cloud-based servers to gather logs of attacker's activities and give valuable insights on the logs that were gathered to help improve security posture. To implement this, Linode cloud platform was used and the Cowrie honeypot was deployed on this server. The log was analyzed and visualized using various data analysis models such as matploitlib. The result included the successful deployment of the cowrie honeypot on the server and the analysis of the logs gathered. Some Ip addresses were noticed to have made countless attempts, usernames like root, nproc and root were used the most to try to access the systems. Username and password combinations like 'nproc/nproc' were seen to be paired a total of 177 out of 512 attacks recorded, 'root/password' were used together 10 times. While the number of successful attacks may have been lower compared to unsuccessful ones, the information gathered remains valuable as it can aid in anticipating future attacks. Through this research, we aim to contribute to the advancement of network security practices and allow individuals and organizations to proactively defend against evolving cyber threats.

Keywords: Honeypots, cyberattacks, cloud server, security

CHAPTER ONE

INTRODUCTION

Background to study

One of the most crucial aspects of information technology (IT) is computer system security. This field is developing quickly because everyone wants to protect their information and no one wants to give it to an attacker through hacking or compromised data information to the Attacker by the intrusion and compromised data *(Selvaraj et al; 2015).* The internet has connected us globally, but it has also exposed vulnerabilities like Heartbleed, ShellShock, and Poodle, raising concerns about system security. Heartbleed showed that attackers can exploit vulnerabilities faster than patches can be deployed. Relying solely on traditional defenses like IDSs and firewalls is inadequate for detecting new attack patterns that can access network resources *(Marcin et al; 2016).*

While honeypots have a rich history dating back to Clifford Stoll's pioneering efforts in the 1980s, their significance in modern cybersecurity remains undiminished. This honeypot already used multiple virtualized systems hosted on a single hardware component. Since then, the development of sophisticated honeypots and honeynets has continued (Somwanshi et al; 2016). One of the first documented instances of using a honeypot occurred in 1986 and involved a widely publicized, high-profile incident. A system administrator for US Berkeley named Clifford Stoll found that someone was logging in with superuser access, as they detail in their book The Cuckoo's Egg. In an attempt to apprehend this assailant, Stoll deployed two defenses in a honeypot configuration. This involved waiting for the unidentified attacker to dial in after connecting borrowed terminals

to each of the system's incoming phone lines. They were able to find exactly what the attacker was searching for thanks to this strategy, and he even succeeded in luring them into a completely made-up Star Wars missile defense system department. This resulted in the first successful honeypot facilitation and the arrest of a German KGB employee *(Kelly et al; 2021).*

Honeypots are systems designed to divert attackers' attention from important systems while collecting data about their malevolent behavior. They are often equipped with event recorders and monitors, allowing for the examination of emerging attack trends. The construction, installation, and usage of honeypots determine their usefulness and potential issues (Titarmare et al; 2019). According to Sahu and Richhariya (2012), honeypots are virtual computers configured to mimic real machines, act or appear to be running complete services and applications, and have open ports similar to those on a regular system or server on a network.

Recently, organizations have developed a unique interest in the implementation of security and protecting information on the Internet. Honeypots are a prominent way for security professionals to take advantage of new techniques used by attackers. A honeypot is an information system resource that gains value through illegal or unauthorized use. Honeypot interactions are assumed to be malicious due to the attacker's perception of the system as vulnerable and easily attacked *(Ahmad et al; 2023).*

Honeypots are widely used globally as a security measure. Honeypots can be categorized as research or production. Honeypots come in various shapes, sizes, and types. This concept's beauty lies in its versatility in honeypot implementation. Research honeypots are

deployed in the open to analyze ongoing attacks, while production honeypots are deployed within an organization perimeter *(Praful et al; 2020).* The honeypot's purpose is to become the target of an attacker's malicious interactions and various threats. This sets honeypots apart from other security tools like firewalls and intrusion detection systems, where being the target of an attacker is an extremely undesirable circumstance. There should be no reason for legitimate users to interact with or enter the honeypot environment *(Kemppainen et al; 2018).*

"IDS keeps an eye out for attackers on the network. But when IDS is used with honeypots, it gets even better at catching threats. Honeypots add an extra layer of security, making the whole system stronger. "IDS, or intrusion detection systems, can be enhanced by honeypots, which simplify detection processes for unauthorized probes, scans, and attacks. These tools complement IDS, providing a targeted, nuanced approach to threat detection, enhancing overall security posture. IDS is a digital device that monitors network traffic for policy breaches, issues, and malicious activity, proactively reporting any harmful activity to the administration *(Charvi et al; 2021).* An intrusion detection system, or IDS, examines network traffic to find exploits and vulnerabilities. It can also log events, provide alerts, and notify administrators of potential assaults via email.

In contrast, an intrusion prevention system uses its database's knowledge of attack behaviors to try and stop known intrusion signs and specific undiscovered attacks. However, an IDS generates thousands of intrusion alerts daily, and some may include false positives. This usually makes detection of the actual threats and protection of assets complex for the IDS. Thus, the intervention of humans is necessary to look into the assaults that an IDS has identified and reported (*Kaur et al; 2014).* Honeypots streamline the

detection process by operating without any legitimate production activity. This makes any connection to a honeypot inherently suspicious, allowing it to identify unauthorized probes, scans, or attacks with minimal false positives and negatives. On the other hand, intrusion detection systems are specifically designed to identify attacks *(Marcin et al; 2016)*. To detect the black hats. Society must keep up with hacker innovations. In recent times, two types of security scene activities have been observed: black hats and white hats. Black hats destroy the network, while white hats protect it and to research hackers in social networks and how they communicate with one another. It is necessary to provide a real operating system to the a/ttacker in order for the attacker to gain root privileges on the system and identify information about the attack *(Janardhan et al; 2016).*

The main purpose of this project is to deploy a honeypot in a controlled environment to collect footprints and cyber-attacks carried out by attackers. This honeypot will serve as a decoy system, attracting malicious actors and recording their activities while protecting legitimate assets. The captured data will be saved in a detailed log, providing administrators with valuable insights into the various cyber-attacks employed by adversaries. This analysis will help administrators understand the tactics, techniques, and procedures used by attackers, thereby increasing their knowledge and preparedness for future cyber threats.

Problem Statement

With the rapid progress of computer and internet technologies, network security has become a pressing concern. The frequency of cyber-attacks is increasing daily. Attackers employ various tools such as Nmap, SubSeven, or LoftCrack to identify and exploit vulnerabilities in a company's firewall. While there are existing methods like IDS and Firewalls to detect suspicious activities, they have limitations such as producing a high rate

of false alarms and generating alerts lacking sufficient detail for thorough analysis (*Akshay; 2016).* Also, Intrusion Detection Systems (IDS's) and firewalls do not usually show the footprints and patterns of attackers while honeypots and honeynets do not directly address security issues; instead, they give system administrators information to enhance network security as a whole. The primary goal of the honeypot is to identify the attacker, grant them access to the network, record the intrusion's details (IP address, TCP address, etc.), and redirect their traffic to the fake database rather than the real one *(Kemppainen et al; 2018).* Implementing a Honeypot will give administrators a way of obtaining the attackers information and footprints which will be stored in a log to enable them to analyze these cyber-attacks

Aim and Objectives of the Study

This project aims to design and implement a honeypot system for cyber-attack detection. The objectives of the project are:

i. To Implement a honeypot system in a controlled environment (cloud server) to gather logs.
ii. To analyze the data on the attacker's activities and footprints stored in the log from (i).
iii. Give valuable insights into the behaviors and methodologies of attackers from the analyzed log in (ii) to enable more effective and efficient security intelligence.

Justification Of The Study

The goal of the study is to address a critical need in modern cybersecurity. The increasing frequency and sophistication of cyber-attacks have made regular defense mechanisms such as firewalls and intrusion detection systems (IDS) inadequate. The need for creative approaches to strengthen security postures is made clear by the appearance of new attack vectors and the shortcomings of conventional methods in offering thorough insights into attacker behaviors.

Focusing on the implementation of honeypot systems, the research offers a proactive and sophisticated approach to identifying and evaluating cyberattacks. Honeypots provide important insights into malicious activities while lowering the risk to real assets. They are made to mimic real systems and entice attackers. This proactive strategy emphasizes the value of preventive defenses and fits in well with the way cyber threats are changing.

By employing honeypots, administrators can gain access to extensive datasets that offer significant insights into the tactics employed by attackers, enhancing their ability to predict and effectively address cyber threats.

Scope Of Study

The current study or project focuses on deploying a honeypot system to detect cyberattacks. The important part of this system is that the framework to be deployed can detect various attacks. It is important to highlight that the system is designed in accordance with specifications and guidelines that allow it to be both appropriate and adaptable enough to safeguard users' computer systems or various organization’s computer networks. The target's vicinity is also used to acquire intelligence on the attacker.

Brief Methodology

Cloud Platform Selection

In selecting the appropriate cloud platform for hosting the honeypot system, a comprehensive evaluation is imperative. Factors such as scalability, availability, performance, and compliance requirements are carefully considered. This assessment extends to factors like the geographical locations of data centers, network latency, and compliance with data sovereignty regulations. Additionally, the platform's support for virtual machines, containers, and networking capabilities necessary for hosting and configuring the honeypot system is thoroughly examined.

Honeypot Deployment

Installation and configuration of the honeypot software on the cloud-hosted VMs or containers. Customize the configuration to emulate various services and protocols to attract potential attackers. The desired honeypot software is installed on cloud-hosted VMs or containers. Emulate various services and protocols to attract potential attackers. Implement security best practices like network segmentation, access controls, and regular software updates to prevent compromise of the honeypot environment.

Monitoring and Logging

Behavioral Analysis:

Analyze the data to identify recurring patterns, tactics, techniques, and procedures (TTPs) used by attackers. This may involve categorizing attacks, identifying attack trends, and profiling attacker behavior. This process involves going through large amounts of log data, identifying common attack vectors, and understanding the motivations and strategies used by various threat actors.

Insight Generation

Generating actionable insights from honeypot data is critical for improving overall security posture and response effectiveness. Synthesizing behavioral analysis findings into intelligence reports allows stakeholders to gain a clear understanding of the observed attacker behaviors, tactics, and techniques. These reports focus on key findings, trends, and recommendations for strengthening defenses and improving incident response capabilities.

Expected Contribution To Knowledge

The research is expected to make an essential contribution to cybersecurity in multiple areas. Initially, it seeks to improve comprehension of attacker behaviors by analyzing data gathered from honeypot interactions. By revealing recurrent motifs, strategies, and tactics used by attackers, the study will offer deeper insights into cyber threats, driving the creation of more effective defense strategies. Secondly, the study will emphasize the use of honeypot systems as proactive defense mechanisms. By attracting and interacting with potential attackers, these systems lessen the possibility of successful breaches against legitimate assets, highlighting the significance of preemptive security measures to mitigate cyber risks. Through the conversion of raw data into strategic insights that fortify defenses and increase resilience against cyber threats, this process will enable stakeholders to make well-informed decisions regarding security posture and incident response.

CHAPTER TWO

LITERATURE REVIEW

Honeypot

A honeypot is an information system resource that gains value through illegal or unauthorized use. Honeypot interactions are assumed to be malicious due to the attacker's perception of the system as vulnerable and easily attacked *(Ahmad et al 2023).*

By the way, the first resource was a book by Clifford Stroll titled The Cuckoo's Egg (Cliff Stoll, 1989). The second is the white "An Evening with Barford in which a cracker is lured, Endured, and studied (Bill Cheswick, 1992)" by the security icon Bill Cheswick. This does not mean that honeypots were not invented until 1990; they were undoubtedly developed and used by various organizations well before then. Much research and development has occurred within the military, government, and organizations, but very little of its public knowledge before 1990.

The concept of honeypots was first described by Clifford Stoll in his book. This honeypot already used multiple virtualized systems hosted on a single hardware component. Since then, the development of sophisticated honeypots and honeynets has continued (Somwanshi et al ;2016).

Honeypots can be viewed in terms of the information they provide to intruders. These parameters may include the value and level of the information, the sources that intruders can access, and the degree to which intruders can identify the honeypots. Honeypots can be divided into three groups based on how interactive they are: low, medium, and high interaction *(Baykara et al; 2018).*

Low-interaction honeypots

Low-interaction honeypots only mimic a limited range of services, such as SSH or FTP; they deny the attacker any access to the operating system. Low-interaction honeypots generates the bare minimum of responses to facilitate protocol handshakes. Since there is little data collection, low-interaction honeypots are primarily used to evaluate statistics. Still, they are adequate to identify spikes in the volume of requests, such as those generated by autonomous worms. In general, low-interaction honeypots are production honeypots ((*Nawrocki et al; 2016*).

Medium interaction Honeypot

A moderately involved honeypot offers more interaction options but lacks an operating system. Advanced fake daemons increase vulnerability, making compromises improbable. Higher interaction levels allow for sophisticated attacks through logging, making compromises more likely.

High-interaction honeypots

The most advanced honeypots are those with high levels of interaction. Since they give the attacker access to an unrestricted real operating system environment, they are the most difficult to implement, deploy, and maintain. In addition, a vast array of services is set up. High-interaction honeypots gathers as much data as it can, including full attack logs, data access, file tree traversals, executed byte codes, and more. Owing to its great complexity, networks forensics specialists typically perform high-interaction honeypots log analysis; it is not usually done automatically. High-interaction honeypots frequently serve as research spies.

Existing Honeypots

There are various honeypots for different purposes to detect threats. They all operate differently.

Cowrie

Cowrie is frequently used to simulate SSH services and is actively updated. It's still a useful tool for logging and debugging SSH-based assaults. Cowrie's strength is its ability to capture conversations with the SSH service that is being imitated, which offers comprehensive insights into the actions of the attacker. Cowrie's surrounding population is lively, which adds to its continued efficacy as an SSH honeypot.

Its abilities go beyond mimicking SSH. Its ability to imitate other operating systems and services makes it a flexible tool for researching a wide range of attack methods. By exposing trends in attacker behaviors, the thorough logging helps with threat intelligence as well as post-incident analysis. Cowrie's active community involvement guarantees that it stays at the forefront, adjusting to the changing tactics used by bad actors.

HoneyD

HONEYD is maintained and developed by Niels Provos. In 2019, its ability to be customized to behave like other operating systems and services adds to its usefulness as a honeypot. The involvement of the community is essential to Honeyd's continued relevance. Honeyd is a honeypot for Linux/Unix developed by `security researcher Niels Provos. Honeyd was ground-breaking such that it could create multiple virtual hosts on the network (as opposed to just using a single physical host). The honeypot may mimic different

services and operating systems, each of which reacts differently to different messages. Honeyd can trick even sophisticated network analysis tools like **nap** because it emulates operating systems at the TCP/IP stack level. Upon attack, Honeyd can passively attempt to identify the remote host. But this work was not platform independent.

Honeybot

HoneyBOT is a Windows medium-interaction honeypot by Atomic Software Solutions and was written using pHp. It originally began as an attempt to detect by the Code Red and Nimda worms in 2001 and has been released for free public use since 2005. HoneyBOT allows attackers to upload files to a quarantined area in order to detect Trojans and rootkits. The limitation of this work is that, it was not time efficient.

Nepenthes

Nepenthes is a low-interaction honeypot designed to imitate the vulnerabilities that worms use to spread and to capture the worms themselves. Because of its modular architecture, Nepenthes makes it easy to add new security flaws. Extra modules include utilities such as file download and submission functions, and a shellcode handler. Even though Nepenthes finds fresh exploits for previously discovered flaws, it still takes specialized knowledge to imitate fresh flaws and successfully interact with malware. Unknown exploits are highlighted in the log files; this information can be used to develop new modules or more interesting dialogue to promote downloads *(Nawrocki et al; 2016).*

HoneyD

Niels Provos developed HoneyD, a low-interaction honeypot, as a side project. It simulates a range of virtual network environments and operating systems, with the OS executing services to intercept and log attack packets. Since response packets are framed to look as

though they are from a particular operating system, giving the impression that the system is real, HoneyD refers to each of its systems as a personality engine *(Sadasivam et al; 2015).*

Jackpot

Jackpot is a spam-fighting SMTP relay honeypot. It is designed in Java and has a number of configuration options to maximize the effectiveness of spam tracking and trapping. Three methods are used to classify spam automatically: (i) using different antispam databases; (ii) automatically identifying whether a message is spam or a relay test; and (iii) simulating a very slow server by delaying replies. Jackpot presupposes sophisticated attackers who send test emails to their personal email accounts in an attempt to find honeypots. In order to prevent honeypot detection, Jackpot silently drops messages to potential victims' inboxes while delivering relay test messages to the attacker's inbox. Client IP addresses and spam messages are all recorded. A user-friendly HTML GUI is offered by Jackpot *(Nawrocki et al; 2016).*

Kippo

Kippo is more of a medium interaction honeypot than a low interaction one. It focuses primarily on the brute force attack log and, more crucially, the attacker's entire shell interaction. Kippo records an attacker's entire shell interaction whenever they attempt to take over the SSH *(Ahmad et al; 2023).*

Proposed Honeypot Systems

This section focuses on the recent evolution of honeypots by describing the new types of honeypots recently proposed. Protecting the privacy of sensitive information in network systems has grown more challenging as a result of the increased focus on identifying unwanted intrusions. While there are several security measures offered, each one has its

drawbacks. Most of the research being done now uses machine learning algorithms to find intrusions by gathering data using different information technologies, especially honeypots.

In **2015**, An overview of intrusion prevention systems (IPS), intrusion detection systems (IDS), and honeypot systems was given by **Baykara**. The Baykara made recommendations and talked on potential deployments and designs for the above stated technologies. Furthermore, Campbell showcased their survey study, which examined emerging patterns in Honeypot studies. The following is how the writers summarized and spoke about their findings: the first was the publication year, secondly was the publication kind, thirdly was the publication's place of origin and fourthly was the topic themes that were found from a few sources that were pertinent to their research.

Fan et al, in 2015 developed the creation and standardization of a common (technology-independent) Honeynet descriptive language for the concurrent functioning of several Honeynet platforms is a new area of study in honeypot research. Fan et al. also provided an overview of popular descriptive languages and suggested one that could be used to manage and configure various honeynet deployments. HoneyGen, a versatile configuration tool, featured this language. Primarily, the authors recommended firstly to expand the HoneyGen tool to accommodate other Honeynet platforms and to also investigate the automated conversion between the technology-independent language that is provided and the proprietary deployment languages used by Honeynet platforms.

Selvaraj et al. (2015) proposed an innovative approach to enhance network security through the integration of honey pot techniques with packet data analysis. This method involves the deployment of a virtual honey pot system designed to attract and trap potential

intruders, thereby collecting valuable data regarding their activities. One key aspect of this methodology is the utilization of malware samples as training data for packet data analysis. This allows for the identification of suspicious network behavior and the development of effective intrusion detection mechanisms. By redirecting intruders to a fictitious database, rather than the genuine one, the system not only thwarts potential attacks but also facilitates the gathering of essential information such as IP and TCP addresses. This data can then be used for further analysis and to bolster the organization's overall cybersecurity posture.

Qiul et al, In 2016 developed a machine learning-based algorithm to forecast assaults. His idea was to develop an intrusion prevention system with machine learning and honeypot techniques. This hybrid approach prevents invasions by combining a honeypot method with a machine-learning algorithm. **Li et al**. also established a honeypot model that addresses the drawbacks of the available tools by adapting Feng's link protection mechanism. By using the common SNMP protocol, the architects of that model are able to improve communication and management among the defensive mechanisms. A honeypot functions as a defense system by keeping an eye on the defense system and deciding whether to accept or deny assaults based on the state of the system.

Luo et al, In 2017 based on machine learning proposed the setting up of an intelligent honeypot to increase IoT device security. An IoT scanner was created to search the internet for potentially dangerous interactions and educate the honeypot on how to utilize an IoT learner model that can optimize a model to react to each interaction. This allowed the honeypot to remember each device's response. The approach presented by (Owezarski et al., 2014) uses unsupervised anomaly and honeypot data to classify assaults using clustering approaches such as density-based clustering, subspace clustering, and evidence

accumulation. By leveraging social honeypots to find out about harmful accounts on social networks such as Facebook and MySpace, the authors of (Lee et al., 2010) propose an automatic classification of spam utilizing machine learning techniques e.g. Support vector machines (SVM's).

Also, **In 2018 Fan et al**. published a thorough analysis of honeypot systems. The authors examined emergent trends, software tools and features, virtualization techniques and configuration, honeypot ideas, classifications, architectural aspects (decoy and captor), and deployments (centralized and distributed). Different Honeypot and Honeynet deployment designs in an Internet of Things (IoT) network were described in the work of Oza et al. (usually for collecting threat intelligence data essential for analysis and mitigation of assaults). A succinct summary of cloud-based honeypot and honeynet principles, architectural deployments, and models was also provided by Veni et al.

In 2017, Fraunholz et al. provided a thorough overview of deception technologies, such as honeypots and honeynets. They covered the basic ideas and common software tools, architectural implementations, and the moral and legal implications of deception technology in their work. Supervisory Control and Data Acquisition (SCADA) and Honeypot-based SCADA are two types of Industrial Control System (ICS) information monitoring systems that Lu et al. provided an overview of, while Razali et al.'s survey work reviewed various concepts, functionality, software, and architecture of Internet-of-Things (IoT) Honeypot. Furthermore, a study was conducted about the use of Honeypot technology in Industrial Control Systems. The authors recommended: 1) additional examination of IoT device attacks based on security metrics (malware login/password, source IP, source ports, type and distribution of attack source, etc.) that describe the

attacker; 2) creation of an intelligent IoT honeypot that can adjust to the level of interaction of the attacker.

(Pandire et al; 2018) implemented the NICE-A Deployment methodology, where a lightweight network intrusion agent, NICE-A, is installed on a cloud server to actively monitor and scan for vulnerabilities. This process generates Scenario Attack Graphs (SAGs) to analyze potential attack scenarios.

In terms of Attack Detection & Analysis, the system employs attack analyzers to classify different types of attacks and recommend appropriate countermeasures. It also redirects detected attacks to honeypots to safeguard the primary system from harm.

The Honeypot Mechanism is an integral part of the defense strategy, as it entices attackers away from critical systems towards decoy environments. This allows for the monitoring of attackers' activities, including tracking their IP addresses and capturing signature attacks, all without their knowledge.

Additionally, VM Profiling continually updates the database with information about attacks targeting open ports, while the network controller evaluates the severity of detected attacks and proposes corresponding countermeasures to mitigate the threat effectively.

In 2019, Zobal reviewed the state-of-the-art Honeypot technologies. A summary of terms, ideas, categories, software tools, advantages, moral and legal dilemmas, and difficulties were covered in their study. Bhagat et al. also provided a succinct synopsis of the ideas, classifications, and common architectural deployments and functions of honeypots inside networks. They recommended further research be done on honeypot installations to fend against both internal and external threats, as well as their uses in other types of networks.

In 2020, Lee at al. presented a study of botnets, botnet assaults, and honeypots used to capture such attacks. The authors proposed that further study be done on the use of honeypots in smart industrial IoT networks, particularly in order to enhance attack detection and response times.

Additionally, Matin et al. examined machine learning-based honeypot-based malware detection and collection systems. They looked at the use of machine learning models and methodologies to study the use of honeypots in malware detection.

In 2021, Franco et al. carried out and presented a survey of honeypot and honeynet systems used in Internet of Things (IoT) and Cyber Physical System (CPS) application domains. They further explored the ideas of honeypots and honeynets, as well as the lessons discovered from diverse deployments, and created a unique taxonomy for categorization purposes. The authors proposed the following strategies: creating Honeypot systems to detect and mitigate insider attacks; researching recently emerging technologies, platforms, and domains; researching established but untested protocols; researching efficient deployment sites and remote management; and researching efficient anti-Honeypot and ant-detection methods.

Khan et al. (2023) conducted a study on Honeypot Deployment, examining various honeypots like HoneyPi, Honeyed, and T-Pot in a network environment. They detailed the Raspberry Pi configuration, testing procedures, and results analysis to evaluate the system's effectiveness in detecting and mitigating potential threats. The study provides valuable insights for improving network security.

Honeypot Versus Ids's

The primary advantage of employing honeypot technology is in the detection domain, as its straightforward design readily handles the difficulties that IDSs typically encounter. The problem of detections and missed detections can be greatly reduced by implementing a honeypot. Unlike an Intrusion Detection System (IDS) a honeypot treats every traffic as an attack. Moreover, IDS systems often depend on defined attack signatures, from their vendors, which means they may not be able to raise alarms, for new and unknown attacks. The fact that the IDS must handle a lot of traffic since they function in a productive environment presents another issue. As a result, they generate enormous log sets that are challenging for network managers to examine. (M. Dacier & F. Pouget, 2004)

1. Cybersecurity trends and developments.

Significant recent developments in cyber security have been made achievable by new algorithms, techniques, and systems (Kapczynski et al., 2019; Huang et al., 2019; Dwivedi et al., 2011). Asymmetrical methods, which promise to address systems that are intrinsically unsafe, are bringing about a new age in technology (Barbulescu et al., 2017). The features of various approaches are shown in table 2 below.

Paper	**Method detail**	**Mitigated attacks**	**Vulnerabilities/Limitations**
(Kapczynski and Lawnik, 2019)	Using cyphers with adjusting key lengths	This system is built to withstand a variety of assaults, including plaintext, related-key, and side-channel attacks.	Execution space and time are greatly expanded.

(Aggarwal and Maurer, 2016)	Utilizing the generic ring method for RSA factoring	RSA's reduction of factoring problems	Various attacks involving cryptoanalysis are feasible.
(Hwang et al., 2016)	Certificates are encrypted with pair less cryptography.	Defends against assaults by employing specific cypher messages	The design is vulnerable to Denial-of-Service assaults since it depends so heavily on bandwidth.
(Fujisaki, 2018)	This approach uses appropriate-length public-key encryption based on a binary string.	A defense against man-in the-middle attacks.	Attacks that interfere with service may result in denial of service.
(Hazay et al., 2018)	For a solution to the factoring problem, use two-party distributed factoring.	Defend yourself against malicious attacks.	Both the application's size and execution time significantly rise.
(Dwivedi, 2011)	Distributed transform encoders for retrieval of messages.	Ensure security against brute force attacks.	There are known plain text attacks that potentially take advantage of this vulnerability.
(Biswas and Mohit, 2016)	Implementing RSA within DES	Protection against a variety of threats.	Brute force attacks and known-cypher text attacks are both feasible.

(Chie, 2018)	Creating session keys using schemes for key agreements.	Passive and active defense models.	Attacks by third parties are possible.
(Barbulescu and Duquesne, 2017)	Making a new key size suggestion using a variation of NFS.	Prevent replay, impersonation, and denial-of-service attacks.	It cannot be accessed by a multiserver environment.

Table 2.1: Various approaches and their features

Fu et al. (2006) conducted the latency analysis of **honeyed** honeypots. Because of the extremely high stimulus accuracy of honeypots like honeyd, the connection delay in certain virtual networks is a multiple of one millisecond or 10 milliseconds. Using the Neyman-Pearson decision theory, they were able to get a high detection rate as a consequence of their approach. Wenda and Ning (2012) claim that consumers can detect honeypots by examining the virtual environment or assessing network monitoring features. A technique of recognizing honeypot network-level activities and offering services was provided by Defibaugh-Chavez et al. (2006) by leveraging the fact that interactive honeypots are always deployed with firewalls and intrusion detection systems. Employing user-mode Linux and virtual machine files, Holz and Raynal developed a honeypot detection method based on the feedback information.

In recent years, there has been a sharp increase in the quantity of assaults on these sites. Furthermore, 20% of all cyberattacks in 2020 will originate from cloud-based platforms. As a result, it's critical to design a system that can fool the attacker and watch the attempt

in great detail, gathering useful data to stop similar assaults in the future. Honeypots, which are specially created virtual systems that imitate the activity of the actual assets that we must safeguard, come in helpful in this situation. Since no system is flawless, honeypots have a number of serious disadvantages. Attacks are likely to happen since the system is made to be attacked, which is one of the fundamental problems. After being breached, the honeypot could serve as a springboard for other attacks. These assaults could be made against a different business or an internal system. Honeypots therefore provide a risk. Consequently, a liability concern arises. If another business is attacked using your honeypot, you can face legal action. The honeypot will decide how much danger is exposed.

The number of individuals using cloud infrastructure to apply their intricate fixes for a wide range of issues is increasing at an exponential rate. This move has the drawback that since no system is impenetrable, these cloud systems are always vulnerable to cyberattacks. Although many cloud systems already have built-in defenses, using a honeypot adds an additional layer of protection and gives us important information about the attack route. In the future, create a hybrid honeynet system to solve the aforementioned issue. This system can combine the finest features of several honeypot kinds to create an even more superior system.

2. Comparative study on various IDS techniques.

Algorithm/ Technique used	**Reference paper**	**Purpose of IDS**	**Advantage**	**Limitation/future scope**

KDD and Clustering	(Chen et al. 2017)	Detecting novel anomalies referred to as NEC.	The presence of high-quality tagged datasets is not required.	A large number of false positives and a high number of false positives.
Decision tree, random forest, K-NN	(Anbar et al. 2016)	To accurately detect potential attack.	Generate remarkable and effective outcomes in identifying assaults based on IPV4.	IPV6 attack cannot be detected yet.
GPFISClass (genetic programming fuzzy inference system for classification)	(Ahmim, A., & Zine, N.G. 2015)	Solving the classification problem in IDS.	Higher classification accuracy.	Introducing the new hybrid GFS that combines neural networks with GFS.

Epigenetic algorithm	(Ghazi et al. 2016)	The future offspring of this couple.	It helps more precisely prevent the curable by preventing illnesses based on external factors unrelated to the sequenced genes.	A shorter time is spent to obtain optimal solutions when there are fewer iterations.

Table 2.1: Various IDS approaches and their features

3. Review based on Machine Learning Algorithms

These days, a lot of network intrusion detection system research studies apply machine learning techniques. Numerous open-source datasets have made it possible to construct a wide range of ways to address the problem of identifying and evading attacks. Hybrid approaches that make use of layered and hierarchical models, identify anomalies, and improve machine learning approaches by adding knowledge-based methods are often created by combining techniques like machine learning and independent data science tools. Most of these strategies, in general, aim to reduce false alarms caused by recurrent assaults while also recognizing novel attacks. In intrusion detection, a variety of machine learning methods are employed, such as neural network-based algorithms, fuzzy logic techniques, and SVMs.

Based on rule-based and decision tree approaches, **Ahmed et al.** developed IDS by combining multiple classification techniques, including J-Rip, REP trees, and Forest PA. Input characteristics are used to categorize the first two methods as benign or hostile. As opposed to REP trees, Forest PA uses the original input characteristics and both the first

and second classifiers' output. We have achieved 97% accuracy with the CICIDS-2017 dataset, and we scored a 95% detection rate. If compared to other methods, we achieved better results. Various studies use hybrid methods rather than single ones. According to (*Sharma et al. in 1998***)**, NBC and NB-Tree combined to improve both the recall of attacks (attack detection ratio) and the accuracy of minority class attacks (including R2L and U2R). By using reduced feature sets to forecast each type of attack using periodic tests and domain knowledge, this system combines both accuracy and recall for small and critical threats. False positives in the IDS are kept to a manageable level. According to the study, this methodology achieves 99.05% accuracy overall and is better at categorizing minor groups when compared to the usual methods.

Moradi and Zulkernine's neural network-based intrusion detection system also detects different kinds of attacks, which allows it to tackle a multi-class problem. Various neural network topologies are assessed to identify intrusions in order to establish the ideal number of hidden layers. In order to identify intrusions based on offline evaluation techniques, the authors employed MLP. They further employed a preliminary validation technique throughout the training to enhance the neural network's rationalization capabilities. According to the test findings, a neural network with two hidden layers identified the logs properly 91% of the time,

and with only one layer, 87% of the time. The suggested method does not cover all classes, but it works well for a three-class scenario. Zhang et al.'s results included the requirement for artificial neural networks to identify intrusions. Regretfully, choosing the neural network's most widely applicable and workable framework is not easy. They need a lot of data, even though they frequently handle noisy data very well.

They planned to build on the standardized and easier-to-use tiered intrusion detection method developed by Gupta et al. to provide a more accurate but computationally costly method. The organization defines integrity, confidentiality, and data integrity using three levels of protection. Features including user identity, connection initiation, and destination layer IP addresses are handled at the packet level in the first layer. assaults such as a probe, U2R, R2L, and DoS assaults are also picked up by the availability layer. It puts into practice things like what information is gathered, how many files have been read, etc. Ibrahim and colleagues tackled the issue of optimizing intrusion detection system (IDS) performance and introduced a multi-layered approach to identify intrusions (*Ibrahim et al., 2012).* The attributes that are best for each layer are determined by the authors using machine learning techniques, taking into account file modifications, permissions, and data integrity. They employed Nave Bayes, C5 decision tree techniques, and MLP neural networks algorithms to identify the best characteristics in order to optimize storage capacity and improve performance. In order to maximize performance and optimize the limited storage space, the authors employed C5 decision tree techniques, Naive Bayes, and MLP neural networks with gain ratios to choose the best features for each layer. The suggested multiple-layer model produced less false alarms than previous methods like MLP neural networks and Nave Bayes because it was more accurate when employing gain ratio as a feature selection strategy. A limited number of invasions can, however, be identified by this system.

CHAPTER 3

METHODOLOGY

The Proposed System

In this cyberattack detection system, the use of cloud for the deployment of the honeypot system was considered after reviewing different methods of deploying different honeypot infrastructure. The study suggested the use of cloud, this was considered for ease of scalability, cost-effectiveness, ease of deployment, global reach, security, and accessibility. The proposed system uses a cowrie honeypot to detect cyber-attacks. The logs gotten after deployment of the honeypot will be analyzed to help people gain insights into common and recurring attacks for attack methods used frequently by attackers.

Cloud Platform Selection

Linode stands out in the competitive landscape of cloud hosting providers by offering a comprehensive suite of benefits to users. Firstly, its reliability is unmatched, underpinned by enterprise-grade hardware and a global network of data centers, ensuring high uptime and minimal downtime. This reliability is complemented by exceptional performance, thanks to SSD storage and powerful processors, enabling users to meet demanding application requirements. Moreover, Linode's scalability empowers users to effortlessly adjust resources to match changing needs, ensuring optimal performance without overprovisioning. The platform's flexibility allows for complete customization of server environments, accommodating diverse use cases and preferences. Cost-effectiveness is another hallmark, with transparent billing and hourly pricing ensuring users only pay for what they use. Security is paramount at Linode, with robust measures in place to safeguard data and infrastructure. Additionally, the platform boasts excellent customer support,

providing timely assistance through various channels. In essence, Linode emerges as a top choice for those seeking reliable, high-performance, flexible, cost-effective, and secure cloud hosting solutions.

The strategic placement of honeypots in various geographical regions is made possible on Linode server by global deployment capabilities, which also make it easier to gather a variety of cyberattack data from around the globe. Because of its global reach, threat detection and analysis are more effective because they shed light on the strategies, methods, and sources of cyber threats in different parts of the world. The capacity to quickly scale resources up or down helps organizations to efficiently gather and analyze large volumes of attack data during peak periods while optimizing costs during quieter ones, particularly as cyber threats evolve and vary in intensity. Because a cloud infrastructure is virtualized, it is simple to contain and isolate honeypot instances, which reduces the potential impact of successful attacks or breaches.

The type of system needed can be selected when setting up a Linode server. Ubuntu was selected for this configuration. Dallas, Texas, is a good location for the server because it is close to a lot of users. For specifications, the server has two gigabytes of RAM to help it function smoothly even when it is handling multiple tasks at once and one CPU core. Additionally, 50GB of storage capacity was assigned to the system so as prevent running out of space.

Honeypot deployment`

For this Project, the cowrie honeypot framework was considered due to its ease of deployment. Cowrie is an open-source honeypot software designed to emulate SSH and Telnet services, attracting and monitoring malicious activity from attackers. As a honeypot,

Cowrie is not a real system but rather a simulated environment intended to deceive and gather information about potential threats. It mimics the behavior of SSH and Telnet services, creating virtual systems that appear vulnerable to attackers.

To install the cowrie honeypot, the linode server is connected to via terminal by entering the SSH command (ssh root@45.33.26.114) and the password used to set up linode server. The port is changed to a higher port, to make it look more real. Necessary dependencies such as python is installed

Installing Cowrie:

- Update the package index -apt update
- Install Cowrie: apt install cowrie

Creating a New User on Cowrie:

- Create a new user account for Cowrie: cowrie-adduser

Logged back in to cowrie:

- Using the SHH access in the root terminal followed by the port number

(ssh root@45.33.26.114 -p 33333)

- Entered the Server password
- Logged back in to cowrie using root

Monitoring and Logging

To check the honeypot for new connections and monitor the amount of access to the log, cowrie logs are typically located in /var/log/cowrie/. The cat command can be used to view the logs: (cat cowrie.log)

Log Data

Log Data Description: There were 1198 recorded attempts to access the honeypot in the log data. In the log file, each attempt is represented by a row with different data spread across 8 columns:

- Timestamps: Shows the attempt's date and time.
- Session ID: A special number assigned to every session.
- IP Address: The IP address that served as the attempt's original source.
- Username: The username that was tried to log in with.
- Password: The one that was tried to gain access.
- Success: indicates the degree of success or failure of the attempt.
- Country: The IP address approximate geographic location.

4. Log preprocessing

Data Cleaning: Finding and fixing mistakes, inconsistencies, or anomalies in a dataset is known as data cleaning. Ensuring data quality and reliability is the goal of this process to enable accurate analysis. The log data underwent cleaning processes to rectify any inconsistencies or errors that could affect the analysis. Standardizing timestamp formats is one of the tasks that data cleaning will entail in the context of log data and also deleting duplicate entries that came from several attempts at logging in.

5. Removal of Null Values:

Null values refer to missing or undefined data entries within the dataset. These can arise due to various reasons such as incomplete logging, errors in data transmission, or system malfunctions.

Removing null values is crucial to ensure the integrity and accuracy of the analysis. Entries with missing data may skew the results or lead to erroneous conclusions. In the context of the log data, removal of null values likely involved scanning each row for any missing entries across the columns and excluding those rows from further analysis.

- Data Analysis: To learn more about the types of attempts, the log data was examined. This analysis included:
- Geographical Analysis: Determining the attackers' location (country) by using their IP addresses.
- Username and Password Analysis: Analyzing the usernames and passwords that the attackers have tried in order to spot trends or common tactics.

6. Metrics used in Honeypot log data analysis

Number of Attacks

The gathered log data provides valuable insights into various aspects of cyber-attacks, including the number of attack attempts or incidents recorded by the honeypot within a specific timeframe. This metric serves as a fundamental measure of the level of malicious activity directed at the honeypot, allowing for the evaluation of overall security posture. By monitoring the quantity of attacks over time, trends, spikes, or patterns in malicious activity can be identified, potentially indicating coordinated attacks or changes in attackers' behavior.

Attack Origin

Analyzing the attack origin from the gathered log data offers crucial information about the geographical or network origin of malicious actors. This can be achieved by mapping

inferred countries based on IP addresses or examining the IP addresses linked to attack attempts. Understanding the global distribution of attackers enables the customization of defensive tactics, such as implementing firewall regulations specific to certain regions or enhancing the exchange of threat intelligence with relevant authorities or institutions.

Attack Type

Insight into adversaries' tactics, techniques, and procedures (TTPs) can be gained by categorizing attacks according to their nature or methodology. Network scanning, brute-force attacks, malware downloads, exploit attempts, phishing, and other techniques are examples of attack types.

To determine the exact kind of attack, examine patterns in usernames, passwords, URLs, and other indicators.

Finding recurring attack patterns or signatures can help with the creation of proactive defenses like intrusion detection systems (IDS), rules based on signature detection, or threat hunting techniques.

System Architecture

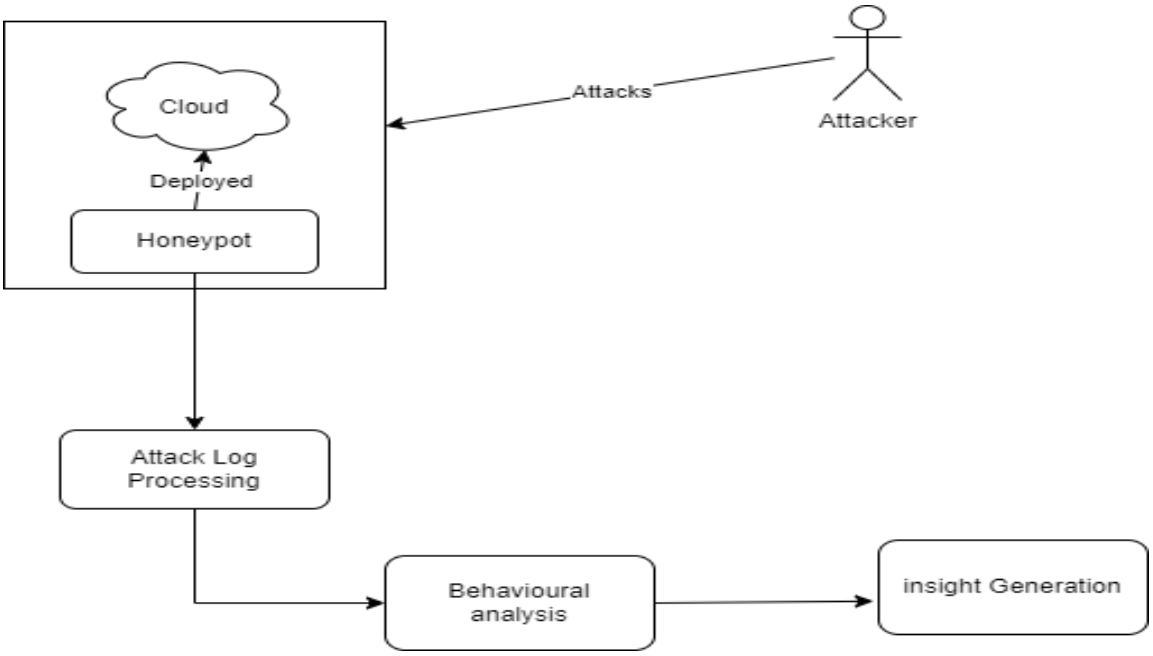

Figure 3.1: Architectural representation of Honeypot system

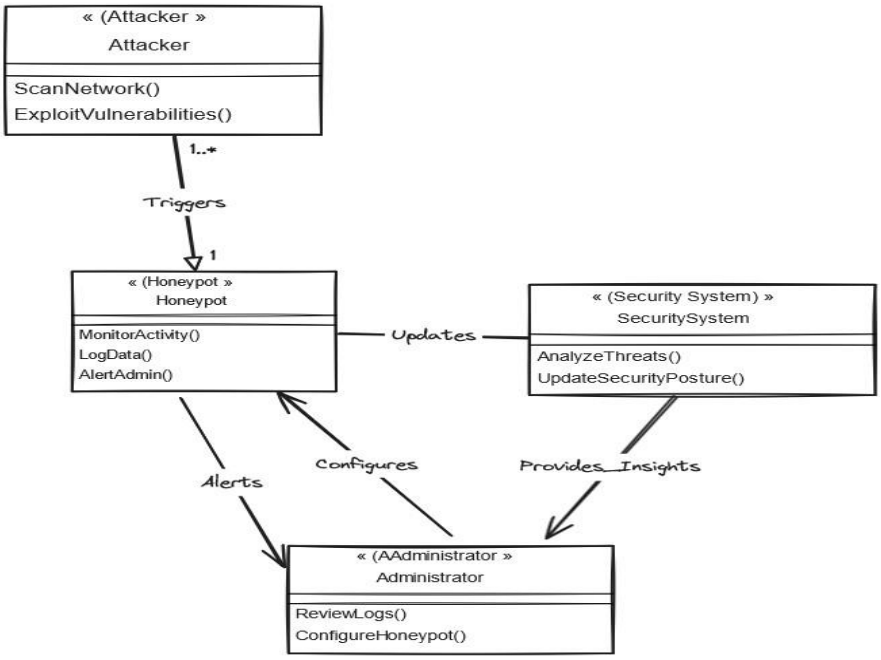

Figure 3.2: use case diagram

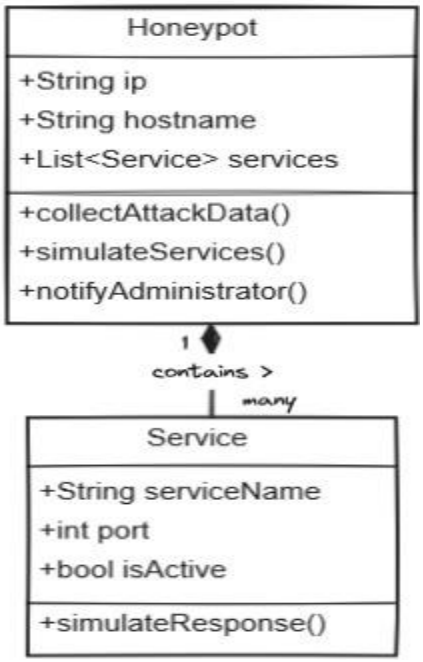

Figure 3.3: class diagram

CHAPTER FOUR

IMPLEMENTATION

Cloud Setup and Honeypot Deployment

This project made use of a Linode; a cloud server that provides Linux-based Virtual machines. On the Linode server the specification of the types of system required was selected and deployed on the cloud. Ubuntu was selected for this configuration with the location at Dallas, Texas, this allocation was selected for the server because it is close to a lot of users. For specifications, the server has two gigabytes of RAM to help it function smoothly even when it is handling multiple tasks at once and one CPU core. Additionally, 50GB of storage capacity was assigned to the system so as prevent running out of space.

After creating the Linode an SSH command key is generated, this key allows us to access the Linode from our terminal. To install and setup the cowrie the following commands and steps were used

i. Change ssh port

   sudo nano /etc/ssh/sshd_config - Edit the configuration file to change the listening port to port 3333. Save the changes and exit the editor.

   sudo systemctl restart ssh – Restart the SSH service to apply the changes

   sudo systemctl status ssh – Check the status of the SSH service to ensure it is working properly.

ii. Install python dependencies: Install necessary updates for the Linode and Python dependencies for Cowrie.

sudo apt update and sudo apt upgrade

sudo apt-get install git python3-virtualenv libssl-dev libffi-dev build-essential libpython3-dev python3-minimal authbind virtualenv

iii. Create user account: Create a new user for Cowrie and switch to that user.

sudo adduser --disabled-password cowrie _ creates a new user for the cowrie

su – cowrie- switch into cowrie

iv. Download cowrie: Clone the Cowrie repository from GitHub.

git clone http ://github.com/cowrie/cowrie

v. Setup virtual environment: Set up a virtual environment for Cowrie and install required Python packages.

virtualenv cowrie-env _ sets up cowrie virtual environment

source cowrie-env/bin/activate

pip install --upgrade pip

pip install --upgrade -r requirements.txt

vi. Configure Cowrie: Copy the default Cowrie configuration file and make any necessary adjustments.

cp /etc/cowrie.cfg.dist cowrie.cfg

vii. Iptables: redirect SSH (port 22) and Telnet (port 23) traffic to Cowrie ports (2222 and 2223).

sudo iptables -t nat -A PREROUTING -p tcp --dport 22 -j REDIRECT --to-port 2222

sudo iptables -t nat -A PREROUTING -p tcp --dport 23 -j REDIRECT --to-port 2223

viii. Start cowrie: Start the Cowrie honeypot.

/cowrie start

ix. View Live Logs: Monitor live Cowrie logs to observe any incoming connections or activity.

tail -f ./var/log/cowrie/cowrie.log

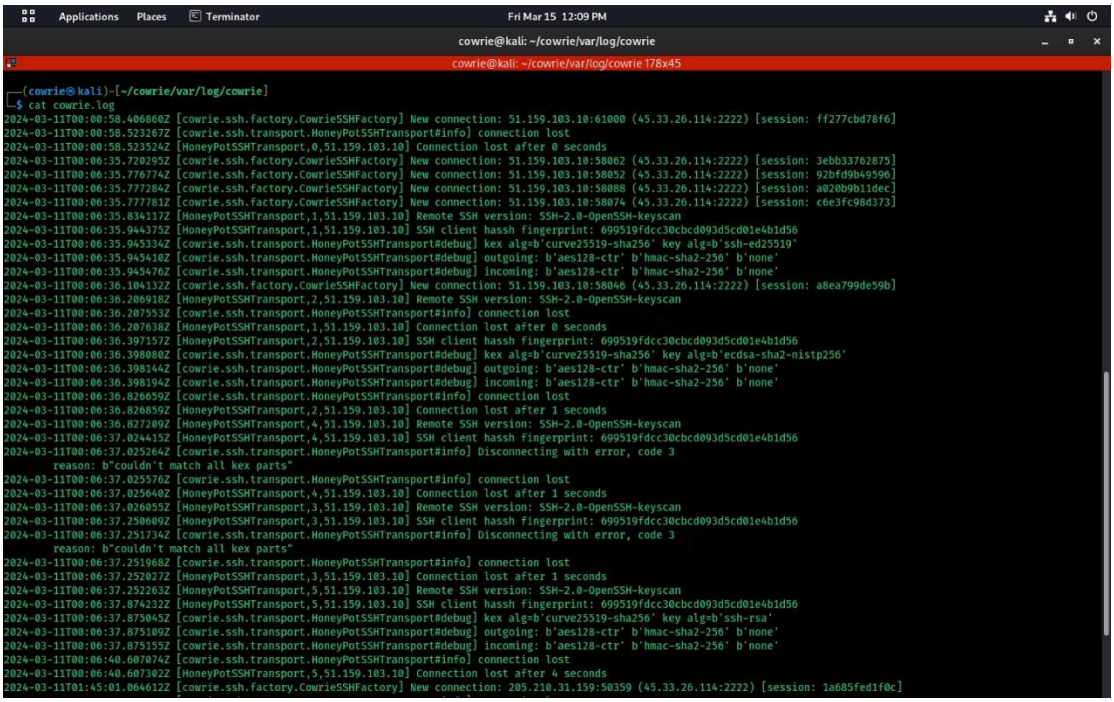

Figure 4.1: Live connection monitoring from Cowrie honeypot

Analysis of log

The log contains 1197*10 format, it contains 10 columns

- Timestamps: Shows the attempt's date and time.
- Session ID: A special number assigned to every session.

- IP Address (from IP address, to IP address): The IP address that served as the attempt's original source.
- Username: The username that was tried to log in with.
- Password: The one that was tried to gain access.
- Success: indicates the degree of success or failure of the attempt.
- Country: The IP address approximate geographic location.

CATEGORY	**DESCRIPTION**	**RESULT**
Basic information	Shape of the dataset	1197 * 10
Basic information	Column names	id, ymd, time, session, from_ip_address, to_ip_address, username, password, success, country
Basic information	Data types of columns	Integer & Object
Summary Statistics	Number of records	1197
Counting Values	Counts of success values	Failed: 869, Successful: 328
Counting Values	Counts of country values	United States: 694, France: 242, China: 175, South Korea: 21
Missing Values	Missing values in each	password: 3

	column	
Successful Logins	Successful logins by username	admin: 15, oracle: 28, root: 218, ubuntu: 17

Table 4.1: Data Analysis result summary

Visualization of Log

7. Geo graphical analysis

According to the system log data, the United States accounted for the highest proportion of attempted system accesses, comprising 57.98%. Following closely behind, France constituted 20.22%, with China contributing 14.62%. It's noteworthy that a significant portion of these access attempts originated from Asia and Europe, indicating a widespread presence of potential attackers across these regions.

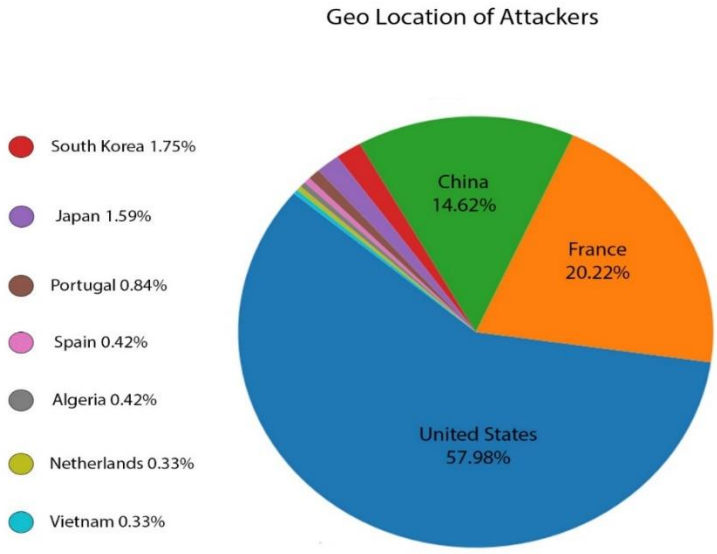

Figure 4.2: Pie chart representation of location of attackers

8. IP Addresses

In the log data, specific IP addresses stand out for their persistent attempts to intrude. Notably, IP address 154.27.68.195 recorded the highest number of attempts, totaling 441. Following closely behind is 138.68.224.69, which made 222 attempts, trailed by 51.15.17.105 with 212 attempts, and 219.134.218.250 with 131 attempts. These repeated intrusion attempts from these IPs raise concerns about potential security vulnerabilities and the need for enhanced protective measures.

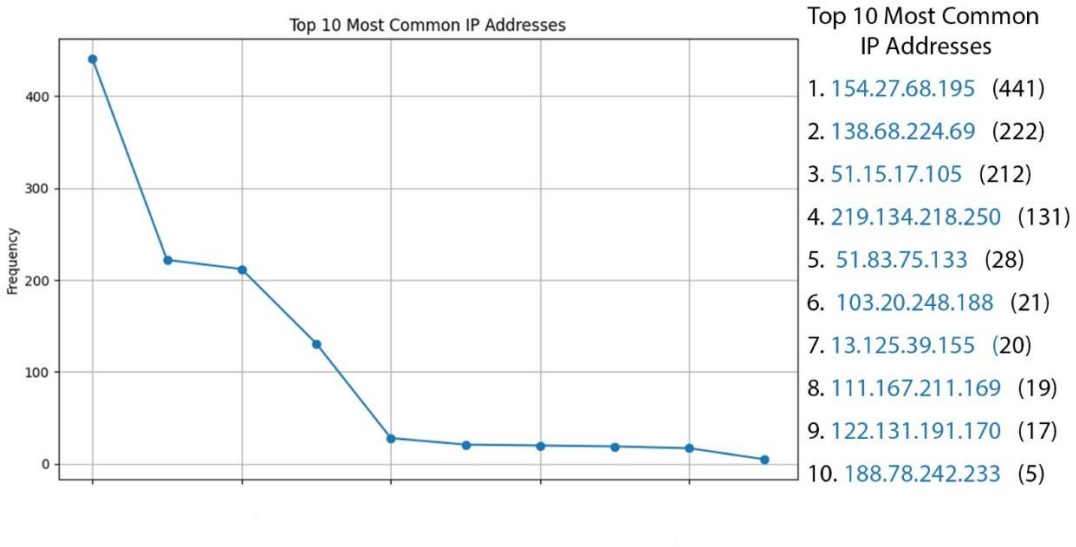

Figure 4.3: Frequency Polygon of top 10 common IP

9. Usernames and Password

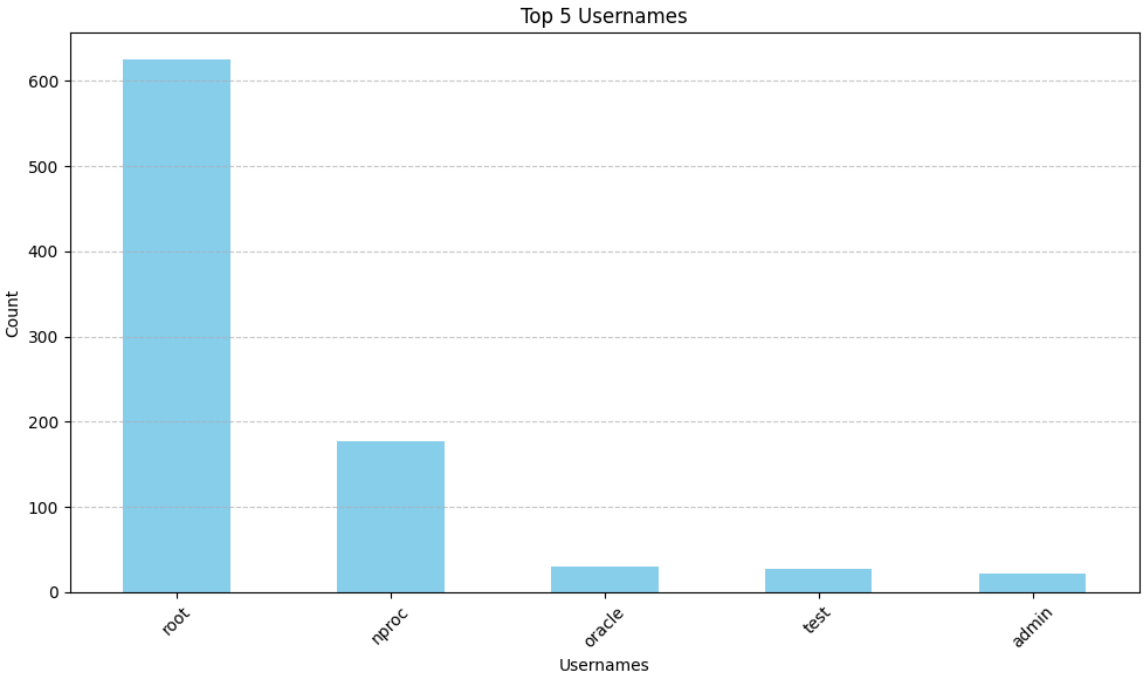

Figure 4.4: Bar chart of top 5 most used usernames

The bar chart showcases the top 5 frequently inputted usernames. At the forefront is "root," which notably appeared 625 times. Following closely is "nproc" with a count of 177, then "oracle" at 30, "test" at 27, and "admin" at 22.

Username	**Count**
root	625
nproc	177
oracle	30
test	27
admin	22

Table 4.2: Username count

The table below illustrates instances of repetitive username-password combinations. In particular, the logs showed that the username "nproc" appeared 177 times, always used together with the password "nproc." Subsequently, "root" became the most commonly used username, and "password" was the most commonly used password for it, as shown in the table provided. A unique combination that was also observed was "Test" combined with the password "test."

Username/Password	Count
Nproc/nproc	177
Root/password	10
Test/test	9
Root/root	9
Root/1234	7

Table 4.3: Username/Password count

10. Success rate

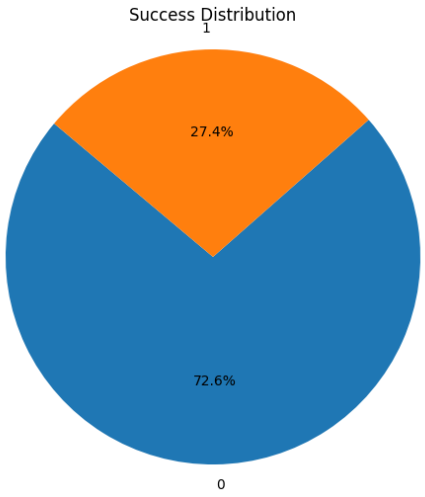

Figure 4.5: Pie chart representation of successful and unsuccessful connections

From the log, successful attacks were not as common as unsuccessful ones, based on the log data. In the honeypot log successful attack were denoted by 1 in the success column and 0 as unsuccessful and about 27.4% of these attacks were successful. On the other hand, attacks that failed, represented by 0, made up the majority at roughly 72.6%. This implies that although successful attacks did happen, they were comparatively uncommon occurrences in relation to the total number of failed attempts.

Insight Generation

These suggestions are based on the geographic analysis and additional information gathered from the log data:

- Enhanced Security Measures: Organizations should put in place enhanced security measures specific to the United States, France, and China, considering the large percentage of attempted system accesses that come from these countries. More monitoring, tighter access restrictions, and the geo-blocking of shady IP addresses are a few examples of this.
- IP Address Whitelisting and Blacklisting: 154.27.68.195, 138.68.224.69, 51.15.17.105, and 219.134.218.250 are a few examples of IP addresses that organizations should think about adding to their whitelist and putting on their blacklist of suspicious or frequently offending IP addresses. This will lessen the danger that persistent attackers pose.
- Password Policy Enforcement: The frequency of recurring username-password pairings, especially "nproc" with the password "nproc" and "root" with popular passwords like "password," highlights the need for strong password policies. It is

recommended that organizations enforce policies requiring complex and unique passwords, provide regular password security training to users, and explore the implementation of multi-factor authentication (MFA) where practical.

- Analysis and Monitoring of Usernames: Constantly keep an eye on and examine usernames that are frequently entered, like "root," "nproc," "oracle," "test," and "admin." Any unusual or suspicious username activity needs to be looked into right away as it could be a sign of malicious activity.

CHAPTER FIVE

SUMMARY, CONCLUSION, AND RECOMMENDATIONS

Summary

The goals of the study, as described in this paper, have been met by this effort. First of all, chapter one provided a clear overview of the problems that motivated this study as well as the goal and objectives required to address the issue that was found. The project main goal was to deploy a honeypot in a controlled environment and gather cyber-attacks which were then analyzed and insights were given based on the log that was gathered and analyzed. In order to accomplish this, it made use of the Linode cloud server, cowrie honeypot and python libraries for analysis of the log data.

A Honeypot system was defined and limitations on different projects that has been done. Different honeypot frameworks and their functionality was discussed. Also, different methods of deploying honeypots were examined.

The process for attaining implementation was then described. It is detailed in stages how the server was set up and the cowrie honeypot was deployed for gathering attacks. The log was gathered and analyzed mainly using python programming language. Recommendations on how to prevent these attacks were then given

Conclusion

In conclusion, our project successfully implemented a honeypot system in a cloud environment, allowing for the analysis of attacker activities and methodologies. Through

meticulous platform selection, installation, and configuration, we gained valuable insights into cyber threats, enhancing our network's defense capabilities. Moving forward, this project could be improved on by exploring other honeypots frameworks to further improve threat detection and response techniques by working with the cybersecurity community and improving on this project.

Recommendation

This project made use of linode cloud server to deploy the honeypot. However, in future research the honeypot should be deployed on an organizations network, and the use of other honeypot frameworks should be considered to gather other types of attacks and an analysis tool can be installed to automatically analyze the data gathered from the system.

REFRENCES

Baykara, M., & Das, R. (2015). A Survey on Potential Applications of Honeypot Technology in Intrusion Detection Systems. *International Journal of Computer Networks and Applications, 2*, 203-211.

Campbell, R. M., Padayachee, K., & Masombuka, T. (2015). A Survey of Honeypot Research: Trends and Opportunities. *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST), London*, 14-16 December 2015, 208-212.

Fan, W., Fernández, D., & Villagrá, V. A. (2015). Technology Independent Honeynet Description Language. *2015 3rd International Conference on Model-Driven Engineering and Software Development (MODELSWARD), Angers*, 9-11 February 2015, 303-311.

Bercovitch, M., Renford, M., Hasson, L., Shabtai, A., Rokach, L., & Elovici, Y. (2011). HoneyGen: An Automated Honeytokens Generator. *Proceedings of 2011 IEEE International Conference on Intelligence and Security Informatics, Beijing*, 10-12 July 2011, 131-136.

Fan, W., Du, Z., Fernández, D., & Villagrá, V. A. (2018). Enabling an Anatomic View to Investigate Honeypot Systems: A Survey. *IEEE Systems Journal, 12*, 3906-3919.

Oza, A. D., Kumar, G. N., & Khorajiya, M. (2018). Survey of Snaring Cyber Attacks on IoT Devices with Honeypots and Honeynets. *2018 3rd International Conference for Convergence in Technology (I2CT), Pune*, 6-8 April 2018, 1-6.

Veni, K., Prabakaran, S., & Sivamohan, S. (2018). A Survey on Honeypot and Honeynet Systems for Intrusion Detection in Cloud Environment. *Journal of Computational and Theoretical Nanoscience, 15*, 2949-2953.

Lu, K.-C., Liu, I.-H., Sun, M.-W., & Li, J.-S. (2018). A Survey on SCADA Security and Honeypot in Industrial Control System. In: *Recent Trends in Data Science and Soft Computing*, Springer International Publishing, Cham, 598-604.

Razali, M. F., Razali, M. N., Mansor, F. Z., Muruti, G., & Jamil, N. (2018). IoT Honeypot: A Review from Researcher's Perspective. *2018 IEEE Conference on Application, Information and Network Security (AINS), Langkawi*, 21-22 November 2018, 93-98.

Bhagat, N., & Arora, B. (2019). Honeypots and Its Deployment: A Review. In: Rathore, V. S., et al., Eds., *Emerging Trends in Expert Applications and Security*, Springer, Berlin, 505-512.

Zobal, L., Kolář, D., & Fujdiak, R. (2019). Current State of Honeypots and Deception Strategies in Cybersecurity. *2019 11th International Congress on Ultra-Modern Telecommunications and Control Systems and Workshops (ICUMT), Dublin*, 28-30 October 2019, 1-9.

Matin, I. M. M., & Rahardjo, B. (2020). The Use of Honeypot in Machine Learning Based on Malware Detection: A Review. *2020 8th International Conference on Cyber and IT Service Management (CITSM), Pangkal Pinang*, 23-24 October 2020, 1-6.

Lee, S., Abdullah, A., & Zaman, N. (2020). A Review on Honeypot-Based Botnet Detection Models for Smart Factory. *International Journal of Advanced Computer Science and Applications, 11*, 418-435.

Franco, J., Aris, A., Canberk, B., & Uluagac, A. S. (2021). A Survey of Honeypots and Honeynets for Internet of Things, Industrial Internet of Things, and Cyber-Physical Systems. *IEEE Communications Surveys & Tutorials, 23*, 2351-2383.

Ameen, N., Tarhini, A., Shah, M. H., Madichie, N., Paul, J., & Choudrie, J. (2021). Keeping customers' data secure: A cross-cultural study of cybersecurity compliance among the Gen-Mobile workforce. *Computers in Human Behavior, 114*.

Alhayani, B., Mohammed, H. J., Chaloob, I. Z., & Ahmed, J. S. (2021). Effectiveness of artificial intelligence techniques against cybersecurity risks apply of IT industry. *Materials Today: Proceedings*.

Lallie, H. S., Shepherd, L. A., Nurse, J. R., Erola, A., Epiphaniou, G., Maple, C., & Bellekens, X. (2021). Cyber security in the age of covid19: A timeline and analysis of cyber-crime and cyber-attacks during the pandemic. *Computers & Security, 105*, 102248.

Zhang, D., Feng, G., Shi, Y., & Srinivasan, D. (2021). Physical safety and cyber security analysis of multi-agent systems: A survey of recent advances. *IEEE/CAA Journal of Automatica Sinica, 8*(2), 319-333.

Dixit, P., & Silakari, S. (2021). Deep learning algorithms for cybersecurity applications: A technological and status review. *Computer Science Review, 39*, 100317.

Anthi, E., Williams, L., Rhode, M., Burnap, P., & Wedgbury, A. (2021). Adversarial attacks on machine learning cybersecurity defenses in industrial control systems. *Journal of Information Security and Applications, 58*, 102717.

Kim, K., Alfouzan, F. A., & Kim, H. (2021). Cyber-Attack Scoring Model Based on the Offensive Cybersecurity Framework. *Applied Sciences, 11*(16), 7738.

Gunduz, M. Z., & Das, R. (2020). Cyber-security on a smart grid: Threats and potential solutions. *Computer networks, 169*, 107094.

He, Q., Meng, X., Qu, R., & Xi, R. (2020). Machine Learning-Based Detection for Cyber Security Attacks on Connected and Autonomous Vehicles. *Mathematics, 8*(8), 1311.

Yang, M., & Cheng, H. (2019). Efficient Certificateless Conditional Privacy-Preserving Authentication Scheme in VANETs. *Hindawi Mobile Information Systems*.

Ahmad, W., Raza, M. A., Nawaz, S., & Waqas, F. (2023). Detection and Analysis of Active Attacks using Honeypot. *Int. J. Comput. Appl, 184*(50), 27-31.

Somwanshi, A. A., & Joshi, S. A. (2016). Implementation of honeypots for server security. *International Research Journal of Engineering and Technology (IRJET), 3*(03), 285-288.

Titarmare, N., Hargule, N., & Gupta, A. (2019). An Overview of Honeypot Systems. *International Journal of Computer Sciences and Engineering, 7*(2), 394-397.

Mirlekar, S., & Kanojia, K. P. (2022). Role of Intrusion Detection System in Network Security and Types of Cyber Attacks-A Review. *International Journal, 7*(9), 28-32.

Kelly, C., Pitropakis, N., Mylonas, A., McKeown, S., & Buchanan, W. J. (2021). A comparative analysis of honeypots on different cloud platforms. *Sensors, 21*(7), 2433.

Kemppainen, S., & Kovanen, T. (2018). Honeypot utilization for network intrusion detection. In *Cyber Security: Power and Technology*, 249-270.

Selvaraj, R., Kuthadi, V. M., & Marwala, T. (2016). Honey pot: A major technique for intrusion detection. In Proceedings of the Second International Conference on Computer and Communication Technologies: IC3T 2015, Volume 2 (pp. 73-82). Springer India.

Nawrocki, M., Wählisch, M., Schmidt, T. C., Keil, C., & Schönfelder, J. (2016). A survey on honeypot software and data analysis. *arXiv preprint arXiv:1608.06249*.

Pandire, P. A., & Gaikwad, V. B. (2018, July). Attack detection in cloud virtual environment and prevention using honeypot. In 2018 International Conference on Inventive Research in Computing Applications (ICIRCA) (pp. 515-520). IEEE.

Khan, A. H., Khan, W. U., Hamid, I., Abbas, A. W., Chaudhry, M. H., & Arfeen, N. U. Analysis and Implementation of Honeypot Framework for Enhancing Network Security. organization, 11, 12.

APPENDIX

Converting the Cowrie JSON to CSV TABLE

```
import json
import csv

json_filename = 'cowrie.json'

csv_filename = 'cowrie.csv'

# Field names
fields = ["id", "ymd", "time", "session", "from_ip_address",
"to_ip_address", "username", "password", "success",
"country"]

with open(json_filename, 'r') as json_file,
open(csv_filename, 'w', newline='') as csv_file:
    # Create CSV writer
    writer = csv.DictWriter(csv_file, fieldnames=fields)

    # Write header
    writer.writeheader()

    # Process each line of JSON data
    for line in json_file:
        # Parse JSON
        data = json.loads(line)

        # Filter out any keys not present in the fields list
        filtered_data = {}
        for key in fields:
            if key in data:
                filtered_data[key] = data[key]
            else:
                filtered_data[key] = ""

        # Write data row
        writer.writerow(filtered_data)
```

CREATING A DETAILED ANALYSIS SUMMARY OF THE TABLE

```
import csv


def analyze_csv(csv_file):
    username_password_counts = {}

    # Read the CSV file
    with open(csv_file, 'r') as file:
        reader = csv.DictReader(file)
        for row in reader:
            username = row['username']
            password = row['password']
            combination = f"{username}/{password}"
            if combination not in username_password_counts:
                username_password_counts[combination] = 0
            username_password_counts[combination] += 1

    # Find the top 5 username/password combinations
    top_combinations =
sorted(username_password_counts.items(), key=lambda x: x[1],
reverse=True)[:5]

    # Print the result
    print("Top 5 username/password combinations:")
    for combination, count in top_combinations:
        print(f"Username/Password: {combination}, Count:
{count}")


analyze_csv('cowrie.csv')
```

“root” username count

```
import csv
```

```
def analyze_csv(csv_file):
    nproc_passwords = []

    # Read the CSV file
    with open(csv_file, 'r') as file:
        reader = csv.DictReader(file)
        for row in reader:
            username = row['username']
            password = row['password']
            if username == 'root':
                nproc_passwords.append(password)

    with open("table.csv", 'w', newline='') as outfile:
        writer = csv.writer(outfile)
        writer.writerow(['Password'])
        for password in nproc_passwords:
            writer.writerow([password])


    print("Random passwords used under username 'root':")
    for password in nproc_passwords:
        print(f"Username: root, Password: {password}")


analyze_csv('cowrie.csv')
```

Detailed summary of the data

```
import pandas as pd

# Read CSV file into DataFrame
df = pd.read_csv('cowrie_log.csv')

# Basic Information
print("Shape:", df.shape)
print("Columns:", df.columns)
print("Data Types:", df.dtypes)
print("\nInfo:")
```

```
print(df.info())

# Summary Statistics
print("\nSummary Statistics:")
print(df.describe())

# Counting Values
print("\nSuccess Counts:")
print(df['success'].value_counts())
print("\nCountry Counts:")
print(df['country'].value_counts())    # Count occurrences
by country

# Missing Values
print("\nMissing Values:")
print(df.isnull().sum())   # Get count of missing values in
each column

# Correlation (excluding non-numeric columns)
numeric_columns = df.select_dtypes(include=['int64',
'float64']).columns
print("\nCorrelation:")
print(df[numeric_columns].corr())   # Get correlation
between numerical columns

# Grouping
print("\nGrouping by Country:")
print(df.groupby('country').size())
print("\nSuccessful Logins by Username:")
print(df[df['success'] == 1].groupby('username').size())

# Unique Values
print("\nUnique Usernames:")
print(df['username'].unique())
```