Enhancing Linux System Security through Automated Detection of Privilege Escalation Attempts and mitigation

Network Security and Ethical Hacking,

School of Computing,

National Institute of Business Management (NIBM),

Colombo.

M. Ravindu Nethsara Fernando

COBSCEHNS-22.1P-020

Details of the Undergraduate Research Project

" Enhancing Linux System Security through Automated Detection of Privilege Escalation Attempts and mitigation
"

Name of the Researcher: Mihindukulasuriya Ravindu Nethsara Ferando

Institutional Affiliation: Ethical Hacking & Network Security, School of Computing, National Institute of Business Management (NIBM), Colombo.

Telephone no: 0706604248

e-mail: cobscehns221p-020@student.nibm.k

[illegible]

Signature of the supervisor

Name of Internal Supervisor (s): Mr. Dinusha

Institutional Affiliation(s):

Telephone no (s): +94 76 395 3691

e-mail (s): dinushamalshan064@gmail.com

Contents

Abstract

Honeypots are very useful to detect and analyze threats and intrusions in the current environment of increasing cyber threats. To capture and record user commands, this thesis explores how to create a honeypot environment using Docker and a customized shell wrapper. It is based on the use of predetermined commands in an Ubuntu operating system. We are able to capture commands, track users and thwart intruders to a honeypot shell by using a shell wrapper script. This enhances security standards and provides crucial information on the potential vulnerabilities that may exist.

1. Introduction

As the cyber threats are becoming more and more sophisticated, new and advanced security solutions must be developed. Another type of simulation systems, called honeypots, which are designed to lure intruders, are crucial for understanding the behavior of the attackers and, therefore, for improving the protection measures. Most of the honeypots are deployed to catch the outsiders. Following the COVID-19 pandemic, most companies have embraced remote work culture, which enables employees to work within their organization remotely. This has led to a rise in internal threats. This thesis explains how to set up a honeypot that can record user input and direct attempts to use specific commands to a honeypot environment by integrating Docker with a custom shell script.

Background

Docker Containers

Docker is an open-source platform made to use containerization to automate application deployment, scalability, and management. Code, libraries, dependencies, runtime, and other components required to run a program are all included in containers, which are small, executable, standalone packages. This guarantees that the program functions uniformly in various settings.

Honeypot

Honeypots are employed to deceive the attackers who attempt to penetrate into the networks. Honeypots help in giving the security experts a way of easily detecting, assessing and responding to the unlawful activities by creating the appearance of valuable or vulnerable targets. Besides serving early warning mechanisms, they offer information on the attackers' tactics, methods, and procedures (TTPs).

Shell wrapper

A shell wrapper is a script that acts as an interface between the user and another program or command. It helps in reducing, modifying or enhancing the frequency or manner of entering a command or a set of commands. Shell wrappers are usually scripted in shell scripting languages such as Bash, Sh, or other Unix Shells.

Problem Statement

The problem of network and computer security is substantially hampered in the contemporary inter-connected corporate environment by endogenous threats. These threats could be from the malicious insiders, or attackers who have gained an unlawful entrance to the internal network. The traditional security approaches are inadequate in dealing with internal threats particularly those that incorporate privilege escalation.

Problems with the Current Security Protocols:

Inadequate Internal Threat Detection:
The fact is that many of the security solutions that are currently being sold focus primarily on protecting against threats that come from the outside, which means that threats that originate from within the organization are not given enough attention. The threats are no longer external, and organizations are vulnerable to attacks that come from within the network.

Complexity of Privilege Escalation Attacks:
Privilege escalation attacks result to the attackers getting more access privileges which is a difficult situation to detect and counter. These incidents can result in the disruption of business, grant unauthorized people access to vital information, and allow them to carry out vital commands.

Absence of Monitoring and Response Tools:
It is not very common to find tools that are specifically designed to monitor threats that are internal and then act on them. What is more, most organizations have difficulties to detect such activities that could be indicative of attempts to privilege escalation or in progress.

Delayed Detection and reaction:
The internal threats like those coming from the disgruntled employees or the hacked internal account may take a long time before they are detected and the response made in the same.

Innovative Solutions Are Needed:

Solutions that are focused on the identification and mitigation of risks originating from the internal environment and, in particular, on the management of privilege escalation are urgently required in these circumstances. One of the ways to address these specific internal threats is to use honeypots which are the systems that lure and retain the attackers. Honeypots are also useful to the organizations as they can gain a lot of information about the actions and intents of attackers and malicious insiders.

Particular Omissions Covered by the Suggested Fix:

Targeted Internal Threat Detection: Honeypots are quite different from other security solutions that may be designed primarily for detecting privilege escalation and internal threats. This method is useful in identifying and preventing activity that is coming from the internal network due to its focus..

Behavioral Analysis: Through the creation of detailed logs of users' commands and actions, honeypots allow organizations to study actions and intentions of potential adversaries. This is very helpful in understanding the techniques that are used in privilege escalation attacks and in developing a good defense against them.

Proactive Threat Mitigation: The organizations can prevent potential risks by redirecting suspicious traffic to the honeypots. This way not only an attacker cannot harm crucial systems, but also will act in a controlled environment where one can watch how this is done.

Enhanced Security Posture A specific shell wrapper together with other containerization tools like Docker can be employed for the deployment of honeypots. This enhances the security status of the company as it presents an economical and flexible internal threat management system.

Study Context: Hence, this study aims at using a honeypot with a custom shell wrapper and Docker to specifically protect against internal threats for instance, attempts of privilege escalation. Honeypot environment redirecting users to some commands on an Ubuntu system these users actions and intents are monitored and analyzed in order to gain more knowledge about the threats.

Thus, by concentrating on internal threats and privilege escalation, this work aims to address a major deficiency in existing measures. The work will focus on the specifics of honeypots' application for detecting and preventing internal threats and compare its effectiveness with traditional security measures. The ultimate purpose of this research is to contribute to the improvement of organizations' ability to recognize and act on incidents that are initiated from within the network and thus help create more effective and preventative measures for combating internal threats.

Objectives

withThe main goal of this study is to design, implement, and evaluate a honeypot system to counter internal threats especially privilege escalation within an organization's network. This study aims to achieve the following specific objectives:This study aims to achieve the following specific objectives:

1. Design and Implementation of a Honeypot System:

Objective: To develop a honeypot system with Docker and a custom shell wrapper for internal threats' monitoring and analysis.

Approach: InThis therefore means that to entice and capture malicious activities this entails designing and/or setting up what is known as honeypot system which simulate a real system. Any command that includes 'sudo' for instance will be directed to the custom shell wrapper that in turn directs users to the honeypot for further analysis.

2. Detection and Logging of Privilege Escalation Attempts:

Objective: To identify and record privilege elevation by internal personnel or by intruders who have penetrated the organization's defenses.

Approach: Everything that will be done through the command line will be recorded, as well as the system will be configured to understand specific commands leading to the privileges escalation. These records will be safely kept for further reference by the committee to carry out further examination on the same.

3. Behavioral Analysis of Malicious Activities:

Objective: To study the action and the purpose of the users who are directed to the honeypot environment.

Approach: To study the action and the purpose of the users who are directed to the honeypot environment. Approach: Therefore, the study will establish the commands applied in privilege escalation by analyzing the commands and activities performed in the honeypot. This research will give an insight into the tactics and techniques applied by the accounts that are compromised or by the malicious insiders.

4. Evaluation of Honeypot Effectiveness:

Objective: To compare the efficiency of the honeypot system in detecting and preventing the internal threats as compared to the conventional security measures.

Approach: This work will also include a comparison of the effectiveness of the honeypot system in internal threats identification and prevention with the other conventional tools. We will look at such factors as the false alarms, response time as well as the detection ratio.

5. Documentation and Dissemination of Findings:

Objective: For the purpose of providing the detailed description of the research, its results, and recommendations within the thesis.

Approach: The last thesis statement will involve the detailed description of the development of the honeypot system, the implementation process, the analysis of the same and its assessment. To contribute to the existing literature on internal threat and its management, the findings will be shared with academic institutions and professional organizations.

Thus, these goals were chosen for this study to enhance knowledge regarding internal threats, and to give businesses a practical and effective means of discovering and combating privilege escalation attackers. The findings of this study can assist firms facing similar problems in developing more effective security procedures and enhancing their general security environment.

Significance of the Study

This paper is significant for the following reasons especially in the light of the fact that internal threats to cybersecurity are on the rise in organizations. The research focuses on the particular type of attack, which is privilege escalation, and the threats coming from inside the network, which is a specific but very important area of concern that is not always paid due attention to, even though the consequences can be devastating. Thus, the significance of this study can be summarized by the following considerations:

Enhancing Internal Security Measures:

Internal threats are especially dangerous, especially those which come from the compromised user accounts or other insiders. These attacks are hard to detect and prevent with the conventional security measures. This paper presents a tool for enhancing the internal security measures through identifying and capturing the internal threats through the creation of a honeypot system. Such information will help organizations to understand those threats and prevent them as they will be able to monitor and analyze threats coming from within the organization.

Addressing Privilege Escalation Risks:

Privilege escalation is another common approach used by the attackers to gain the root access and get hold of the confidential information and crucial system. When the attacker increases his or her privileges, the organization may incur severe consequences. Thus, the study's focus on privilege escalation attempts and their documentation provides valuable information that can be used to develop targeted countermeasures. Understanding what privilege escalation attacks are, and ways to identify them can be useful in preventing the attacks from exploiting organizations.

Developing a Honeypot Framework:

When it comes to discovering intricate insider dangers as well as efforts to escalate privileges, the existing security tools are often insufficient. New approach is given by honeypots which are the fake systems to lure and track intruders. This research is useful for the advancement of cybersecurity as it presents the development of a complex honeypot framework that can interact with existing security systems. Besides the other security tools, the framework's ability to record certain actions of particular users, and flag suspicious behavior enhances the security features.

Providing Real-World Application and Case Study:

That is why real-life testing and applicability of the security solution are crucial to determine its effectiveness. This paper has highlighted the use of honeypot system in the context of internal security with the aid of a case study of Ghana Education Service (GES). The conclusions of this case study shed light on the effectiveness of honeypot system in actual situations based on the ability of the system to detect and counter internal threat. This validation enhances the credibility and applicability of the research, hence relevant to other companies struggling with the similar security concerns.

Informing Security Best Practices and Policies:

In order to protect the enterprise from privilege escalation attacks and internal threats, it is necessary to use proper methods and solutions. Based on the findings of this study, the following strategies and policies can be formulated: The recommendations of the research can be of great usefulness to organizations in strengthening their internal security mechanisms. With such guidelines, the risk of internal threats might be reduced by enhancing the system's settings, user's privileges, training of staff, and organization's security measures.

Supporting Proactive Security Approaches:

Proactive security measures need to be taken in a bid to prevent and minimize on the likelihood of the threats. Due to their inherent design, honeypots are proactive security systems that attract and identify the intruders before they can cause much harm. This paper underlines the importance of the preventive security measures and describes, how internal threats can be detected and analyzed with honeypots. Thus, by using the described proactive measures, organizations will be able to increase their resilience to cyber threats as they will be more prepared to prevent potential security incidents.

The current work aims to fill a rather large gap in the literature on cybersecurity by focusing on privilege escalation and internal threats. It is a valuable contribution to the theoretical and applied literature on cybersecurity because of the findings and recommendations that emerged from the present study and that can help organizations enhance their security.

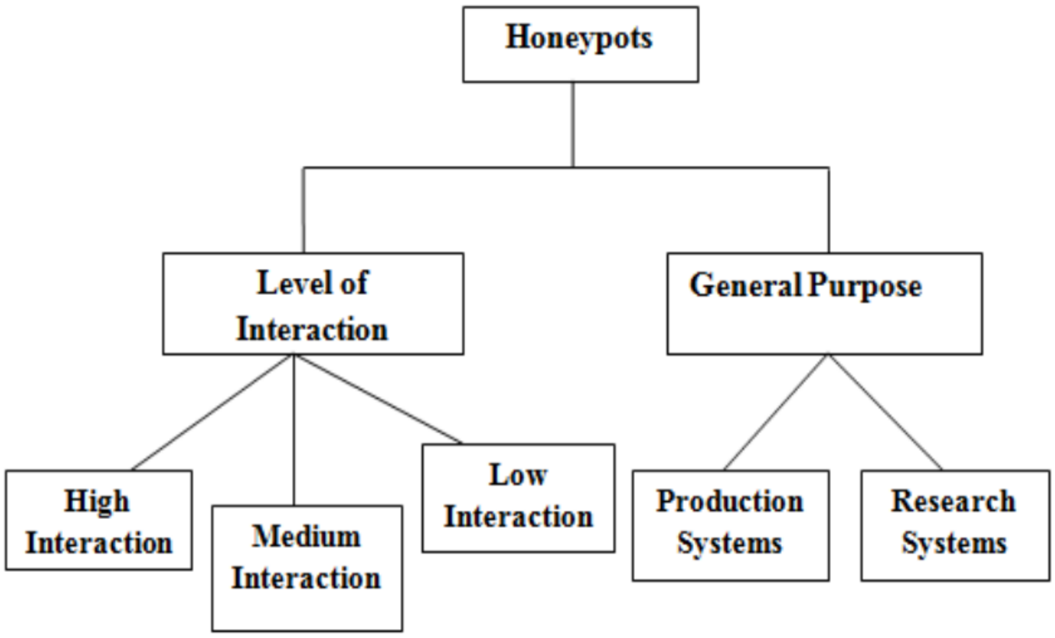
Literature Review

Honeypot Technologies

Honeypot technologies are modern day preventive and counter-action mechanisms that are used to lure the intruders into a certain environment that has been deliberately set up and equipped for the purpose of detecting, analyzing, and managing the intruders' activities. The TTPs of the cybercriminals can be observed by security experts without having to compromise the actual network as these decoy systems can be used to lure them in. Honeypot is mainly used to gain information about potential threats, improve defense mechanisms as well as stopping further advances. All types of honeypot technologies, their characteristics and their roles in cybersecurity are explained in this section.

Types of Honeypots

Honey pots can be categorized into multiple sections based on interaction level and based on purpose [2]



Based on Level of Interaction

1. Low Interaction
   These honeypots mimic a small number of apps and services. They don't provide full-fledged interaction capabilities; instead, they are made to mimic certain parts of a real system. Minimal risk of system compromise, low resource consumption, and ease of deployment and maintenance. Cons consist of Limited capacity to learn detailed information about the tactics and behavior of attackers. Some examples are Valhala Honeypot and Honeyd.

2. Medium Interaction
   Though they are still not fully functional systems, these honeypots offer more interaction than low-interaction honeypots. They provide an environment that is more realistic to fool adversaries. Improved understanding of attacker tactics and increased efficiency in acquiring information on exploits and attacks. Drawbacks, Greater resource needs and a moderate chance of system intrusion. Examples include Cowrie, which mimics a telnet and SSH service to watch attacker login attempts and commands.

3. High Interaction
   These are fully operational systems designed to mimic the real world in which they are placed. They permit extensive interaction between attackers and the system. Thorough data gathering, in-depth examination of the tactics used by attackers, helpful for researching complex attacks. High resource usage and a big risk in the event that the honeypot is compromised. Examples: Installed genuine operating systems, including Ubuntu OS and Windows Server.

Based on Purpose

1. Research Honeypots
   Mainly used in academic and research settings to better understand attacker behavior and create new defense strategies. used in controlled settings to research emerging threats, malware behaviors, and new attack vectors.

2. Production Honeypots
   Incorporated into a functioning network to improve security through the identification and mitigation of actual attacks. used by businesses to gather threat intelligence and reroute attackers away from important systems in order to secure their networks.

Key Components of Honeypot Systems

1. Decoy Services and Applications: Web servers, databases, email servers and SSH.services are the examples of the simulated services, they are the services that mimic real services for the purpose of attracting attackers.
2. Monitoring and Logging Mechanisms: Tools and software for recording and keeping track of all communications with the honeypot, such as file transfers, network traffic, login attempts, and commands that are executed.
3. Data Analysis Tools: Software for analyzing collected data to identify attack patterns, understand attacker methodologies, and generate actionable insights for improving security defenses.
4. Alerting and Notification Systems Mechanisms to alert security teams in real-time about ongoing attacks or suspicious activities detected by the honeypot.
5. Containment and Response Mechanisms: Techniques and equipment to keep the attacker inside the honeypot setting and stop them from moving laterally to target real network assets. automated reaction systems to lessen risks that are identified.

Deployment and Configuration

In this regard, in order to ensure that the honeypots would be easily within reach of those interested in attacking them with the aim of gaining privileges, the honeypots were strategically located throughout the network.

Positioning: put near strategic locations that contain critical data and networks as threats that are worth attacking. To mimic the actual use of users in the internal network environment, the following were incorporated.

Realistic Environment: It is set up with valid users, services, and programs which provide a realistic background. Weaknesses and configuration issues deliberately placed to act as points of compromise to the attackers.

Isolation and Segmentation: As a mean of ensuring that real network assets are not affected in the course of the study, honeypots were isolated from the main network. However, to ensure that any malicious activity that may be in the honeypot environment is well isolated, segmentation of the network was done.

Monitoring and Logging

In order to capture all the activity that is taking place in the honeypots, a lot of monitoring and logging tools were deployed and for the management of the tools used for logging, a central management was done.

Tools for Logging:

Syslog: It is used for logging the activity captured from the honeypots including, file transfer, commands executed and login attempts.

Snort: It has been applied as an Intrusion Detection System (IDS) to monitor network traffic and find out suspicious activities.

Centralized Log Management: All logs were sent to an analysis system that is centralized and the system has features such as alerting, correlation and log aggregation in real time.

The use of honeypot system was a good approach to identifying and preventing the privilege escalation on the Ghana Education Service network[14]. Honeypots were not only beneficial in revealing the tactics and behavior of the attackers but also in the identification of the possible threats and their solutions. Thus, other organizations can enhance their protection against internal threats and privilege escalation attacks by adopting and adjusting this approach.

Docker in Cybersecurity

Docker is a containerization technology that helps to manage and organize application delivery through containers. Its ability to ensure that the applications run uniformly across different environments, its deployment simplicity and efficiency of using resources have been among the factors that have helped boost its popularity. Docker has special opportunities and tools for cybersecurity recommendations that can be applied to enhance security measures for instance by creating a honeypot.

Benefits of Docker for Cybersecurity

Docker container has property of process isolation which means that each running program in its container will not affect other programs running in another container. This containment is necessary for limiting the effect of any possible security breach.
It has been identified that the use of containers is advantageous in enhancing the security postures since they can be easily provisioned, grow, or be decommissioned without disturbing the host or other containers.

With Docker, the risk of configuration drift and the resulting security issues is reduced since the application runs in the same manner in all environments.
Docker images avoid errors that come from misconfiguration because they allow for standardized and easily replicated deployment.

Hypnotize to normal virtual machines, Docker containers are small in size and effective since they do not have their own kernel. Due to its efficiency, the described architecture allows for implementing several security tools and honeypots on one host without overloading it. Docker helps accelerate the process of deployment of the application.
The present study reveals that security tools, which helps organizations to respond to new threats quickly, are effective. This allows the organization to easily increase or decrease its capacity as per the attack or to distribute the security measures across the extended network. Docker supports the idea of breaking the security features down into smaller services on the microservices level. Thus, it is possible to reduce the attack surface and overall increase the security by isolating, controlling, and maintaining each microservice.

Dockerized Honeypots

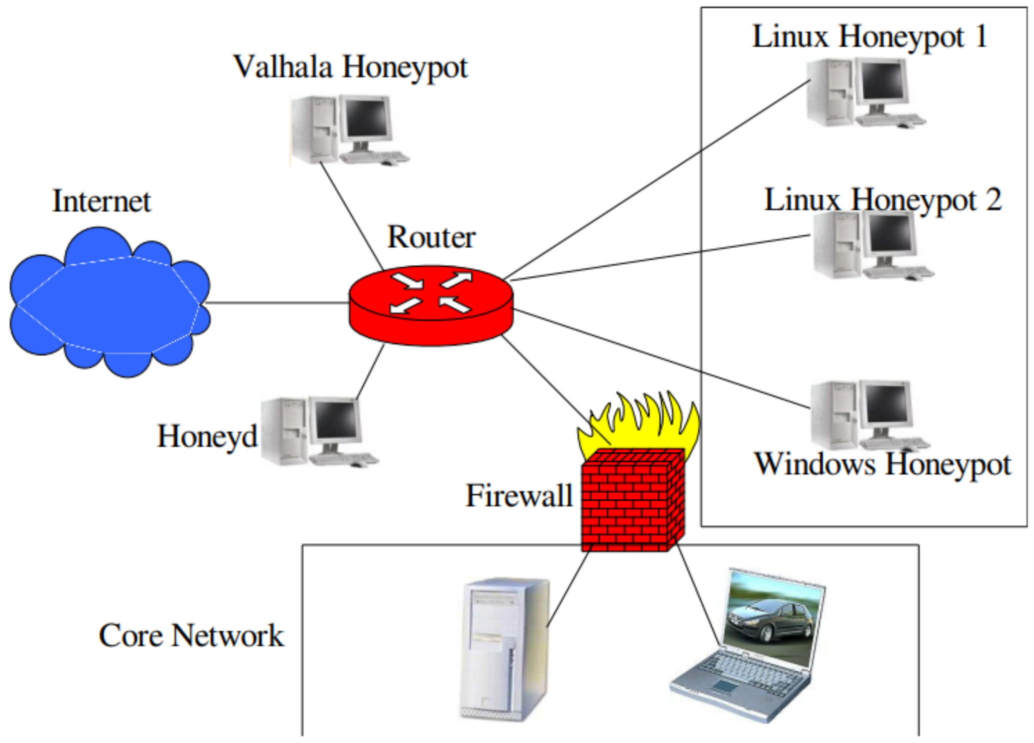
Honeypots can be deployed and managed with Docker, offering an adaptable and scalable method of setting up environments that trick potential attackers. There are various benefits to using Docker for honeypots
Honeypots don't require complicated setup procedures to be deployed across various environments when they are packaged into Docker images. Public repositories contain pre-built honeypot images that allow for quick deployment and customization. By simulating a larger network environment, multiple honeypot instances can be deployed using Docker, which increases the chance of drawing attackers in. Depending on the degree of threat activity, containers can be dynamically scaled to provide sufficient coverage and resource allocation. Since each honeypot is housed in a separate, isolated container, any compromise cannot impact other containers or the host system. To enhance the security, network policies and firewalls can be applied to the individual containers. The honeypots can be easily administered and controlled by means of a control panel even if multiple honeypots are deployed with the help of Docker orchestrators such as Kubernetes or Docker Swarm. They help in the gathering and processing of logs and metrics that are generated by honeypot and provide detailed information on attacker's behavior.

Case Study: Dockerized Honeypots in the Ghana Education Service Network

On January 20, 2015, the official website of the Ghanaian government was taken down by a Turkish person. The Foreign Affairs Ministry website was also compromised before this assault. Nine additional state agencies were also impacted by this attack [1]. Additionally, websites belonging to Kwame Nkrumah University of Science and Technology, the University of Cape Coast, and Presbyterian University College were compromised in the same year.

Deployment Architecture and methodology

By installing low, medium, and high engagement honeypots across our network, data was gathered. In order to better understand how assaults are conducted against our network, these honeypots were strategically placed across the system. Information was obtained from the honeypots placed around our network. The honeypots were set up to record every action and assault.
The logs were then sent to a remote management computer on our network, which is equipped with the software required to analyze and perform statistical analysis on the collected data. After the data was collected, it was analyzed to determine the different attacks that were made against our honeypots, how they were executed, which vulnerabilities were used, and where they originated. A hybrid honey architecture is used for this project.



Results and Analysis

In October and December of 2017, over a period of ninety days, the honeypots were set up and observed. Throughout the deployment and monitoring period, more than 5,000 distinct attack connections were received. The several devices and programs used in the honeypot configuration are shown in Table 1. Leonard and colleagues suggested a "wearable honeypot system" above. The base station and a few specific selected decoy nodes in the BAN exchange fake user health information thanks to this solution. An attack is detected if the traffic is modified in terms of content or arrival time.
Remote system management in client-server environments is accomplished via the use of protocols like RDP and VNC. An attacker may send harmful traffic using these protocols. Danchenko et al. [13] claim that administrators may gather attack data, analyze it, and utilize the information and insight gained from it to lessen such assaults by using honeypots to simulate remote desktop sessions.

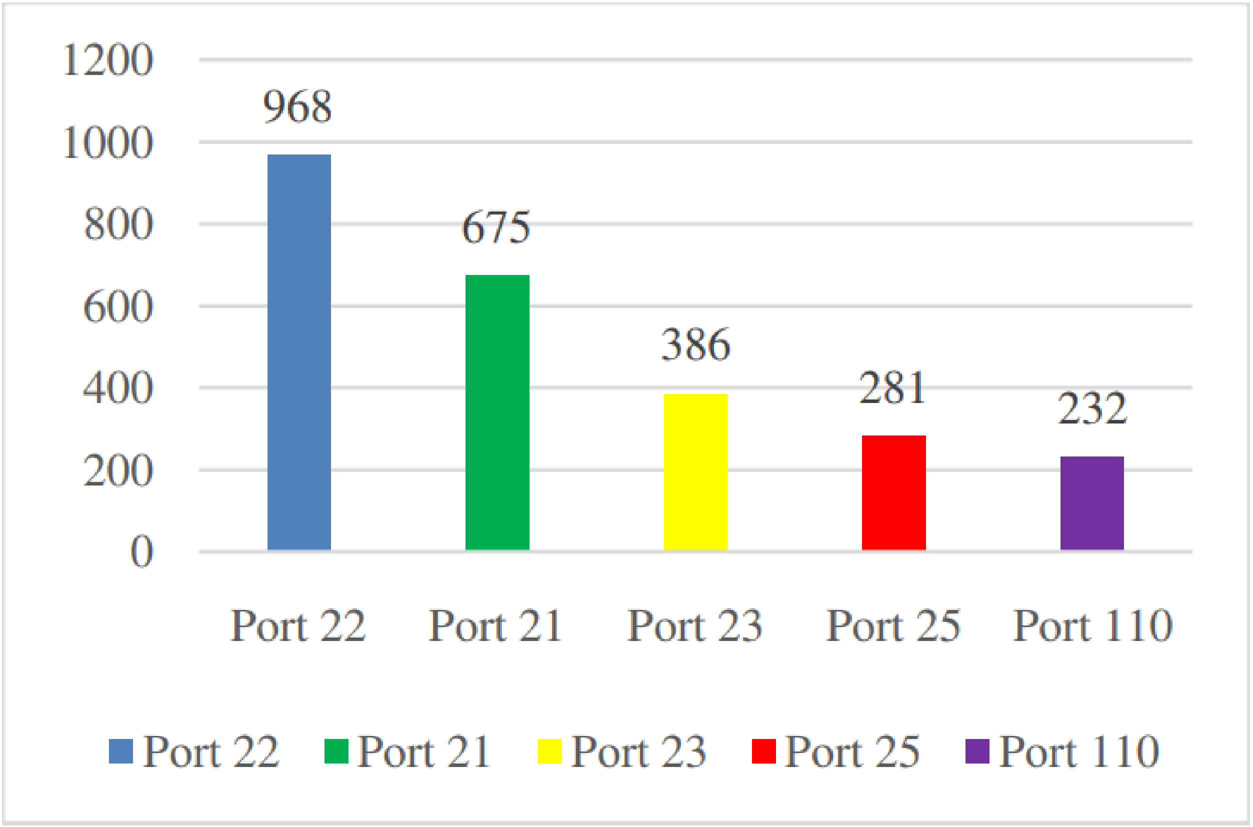
Attack Detection

The majority of the assaults targeted the network's low-interaction honeypots. The majority of the malicious actions used to target these honeypots were port searches using various hues. In addition to port scanning, several threats were noted, including SQL injection, spamming, HTTP authentication brute force assault, and SSH brute force attack. The majority of the assaults that were made against our honeypots were repeated.

Protocol Percentage	Connections	Percentage
TCP	2851	56.33%
ICMP	1442	28.49%
UDP	768	15.17%
TOTAL	5061	100%

It is evident that, with a total of 2851 assaults, or 56.33 percent, the bulk of attempts made against our honeypots were based on TCP connections. Following were ICMP and UDP, with 1442 and 768 assaults, or 28.49 and 15.17 percent, respectively.

Additionally, TCP ports were most often searched, and we discovered that **port 22** (SSH) was the most often examined port, port 22 had the most scans (968), followed by port 21 (675), and port 110 (least amount of scans).



From above results we can see that most attackers try to get a connection to port 22 which is SSH. Privilege escalations happen after an attacker gets a secure shell to the system.

According to the data, the most popular username in terms of frequency of usage is admin, with 467 login attempts. This is followed by root, mysql, guest, and test, with 456, 367, 321, and 245 login attempts, in that order. This is

USERNAME	ATTEMPTS
Root	467
Admin	456
Sql	367
Guest	321
Test	245

We can further confirm that attackers tried to access the privileged accounts the most. Which highlights the importance of gaining access to privileged accounts.

Conclusion

This paper uses the Ghana Education Service as a case study to install a hybrid honeypot in a corporate network.
The network saw the installation of the honeypot for ninety days, from October to December 2017. Analysis was done on the attack data that the honeypots had collected. The organization's core network was then strengthened using the results. Since a well constructed honeypot wastes an attacker's time, it has been shown to be an effective mitigation technique for the majority of attacks. The attacker often leaves the network believing that he has successfully compromised a real system, which gives the network administrator a chance to fix the vulnerability the attacker used in the honeypot setting. The system and network security of GES was enhanced by the installation of honeypots and their integration with other security measures already in place.
The following suggestions are made for the corporate network environment and Small Office Home Office (SOHO) based on our analysis of the attack logs. First of all, while deploying systems, system administrators should never utilize the default login credentials.

Internal risks or the potential for an attacker to access a legitimate user are not the subject of this study, which is centered on external threats. This is where my purpose's research becomes relevant. When an attacker manages to get access to a legitimate user or a legitimate user attempt to breach the network internally, he will be drawn into the honey pot by concentrating on internal risks. The user is moved to the honeypot by the frame works, which keep an eye on their orders and actions and identify any potentially dangerous activity.

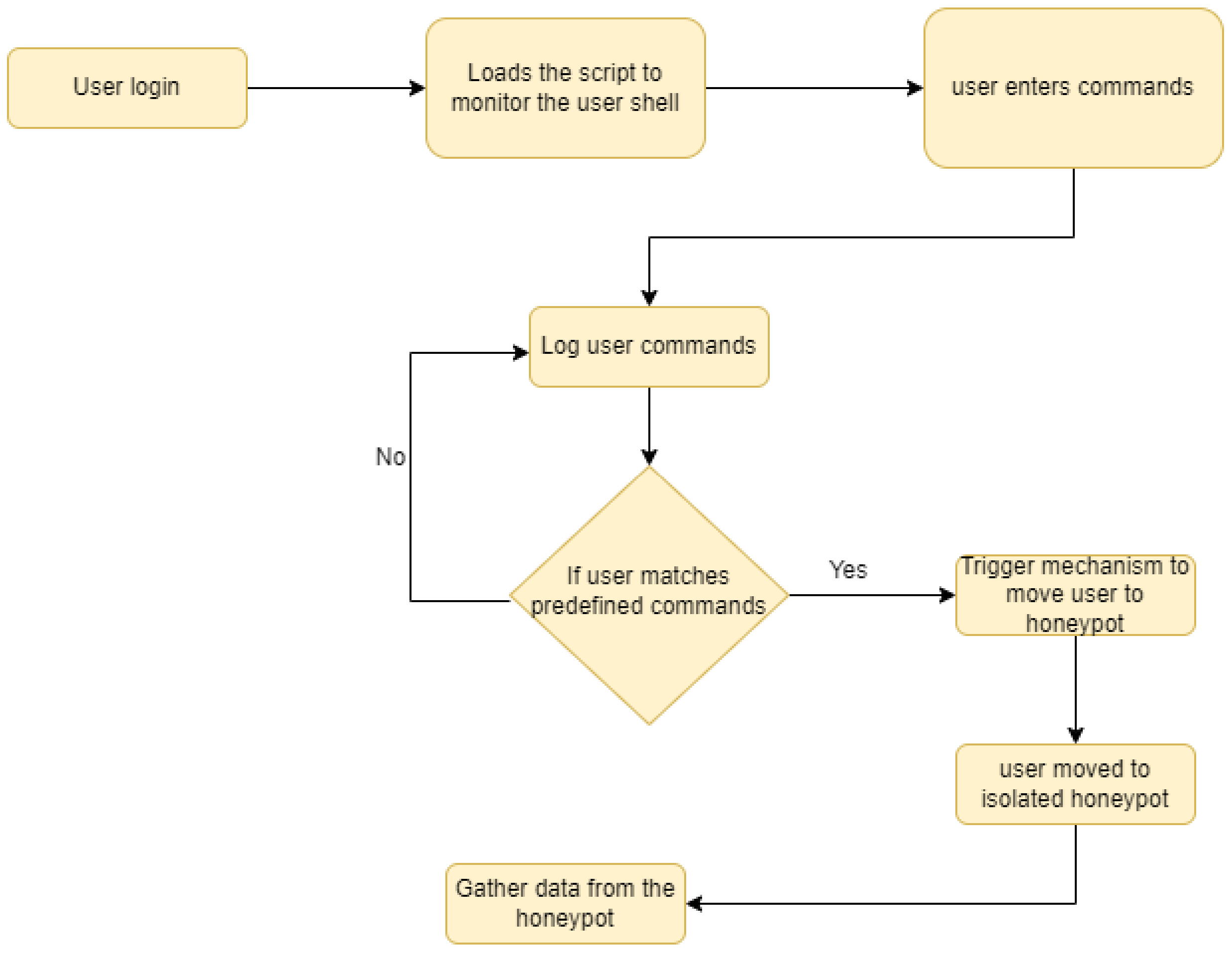
Methodology

Research design

Introduction

The general plan and methodology used to accomplish the study's goals are described in the research design section. The implementation and examination of a docker honeypot system, which is intended to identify and counteract attempts at privilege escalation made by an attacker after they have gained access to a legitimate user, is the main focus of this study. The goal of the project is to gather, examine, and evaluate information on internal risks, with an emphasis on enhancing network security and comprehending the actions of attackers.

Research approach

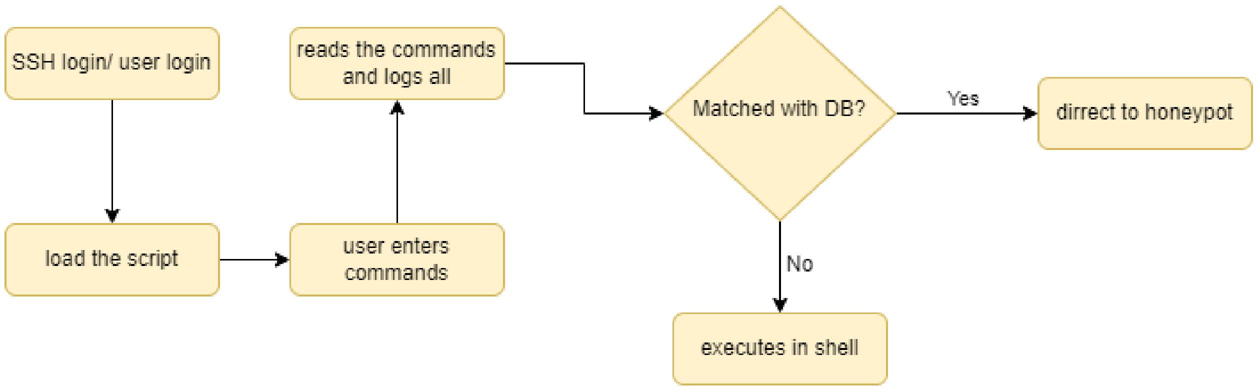

User login
Loads the script to monitor the user shell
user enters commands
Log user commands
No
If user matches predefined commands
Yes
Trigger mechanism to move user to honeypot
user moved to isolated honeypot
Gather data from the honeypot

The study strategy involves keeping an eye on a user who has gained unauthorized access to a system, analyzing their behavior to distinguish between harmful and lawful activities, and moving them to a honeypot if their conduct raises suspicions. It provides a solution to the postexploit issue since there aren't many sophisticated systems to identify a malicious user early on. Most of the time, the attack is discovered later on via logs, but by then it's too late since the attack has already caused enough damage to be detected.

Research methodology

Regarding the approach, I've used a three-step process: (1) identify user input in real time and monitor it; (2) utilize a shell wrapper, which is an active script that acts as a mediator between the user and the system shell. Second, the script actively monitors the input instructions to detect commands before they execute in the shell. This allows the script to initiate commands in the honeypot and direct the user to it. Finally, the system may be used by the user just like a valid system since the Docker container is designed to mimic the user's actual environment.

Shell wrapper



The purpose of the shell wrapper script is to record, intercept, and process user instructions—with specific attention paid to commands that are identified.

Important Elements: Logging and Command Interception:

records the username and the command for each command the user executes in /home/researcher/research/logs/shell_logs.log.
Engaging Mode:
The script enters an interactive shell mode with a customized prompt showing the hostname and username when no parameters are supplied.
records every command typed when in interactive mode.
determines if the command and a record in commands.txt match. If they match, it carries out certain operations like transferring files, pausing and uninstalling the test2 Docker container, and starting a new one.
Managing the Given Commands:
The script records and executes the supplied command when arguments are given.
The command outputs "DETECTED" and ends if it matches an item in commands.txt. If not, it

carries out the order as usual.
Script's primary logic
In the absence of any parameters, the script executes in interactive mode.
In the event that arguments are given, the script executes the command immediately.
This script improves security by tracking user activity and is helpful for controlling and monitoring certain operations, particularly in Docker container environments.

Triggering mechanism

1. **Command Logging**:
   - Every command, whether in interactive mode or provided as an argument, is logged by the log_command function.
2. **Command Detection**:
   - The script uses grep to check if the entered or provided command matches any entry in commands.txt.
   - The grep -Fxq "$cmd" "$DETECTED_COMMANDS_FILE" line checks for an exact match of the command in the file.
3. **Triggered Actions**:
   - If a match is found, specific actions are executed:
     - In interactive mode, the script checks if a Docker container named test2 is running. If so, it stops and removes the container.
     - Then, it runs a new Docker container named test2 with specified parameters.
     - Copies files into the container.
   - If no match is found, the command is executed normally using eval in interactive mode or exec for provided commands.

This setup allows the script to monitor and react to specific commands, making it useful for scenarios where certain commands need special handling, such as in a research environment or security-sensitive context.

Docker container

The docker container is built with using the ubuntu 20.04 as the testing Linux host is running ubuntu 20.04 to give the user similar experience as the host user so the attacker will not be suspicious working on the honeypot.

1. **Base Image**: Uses Ubuntu 20.04 as the base image.
2. **Package Installation**: Installs essential packages including SSH server and network tools.
3. **Root Password**: Sets a root password.
4. **User Creation**: Creates a non-root user named test2.
5. **SSH Setup**: Configures SSH settings for the test2 user.
6. **Environment Configuration**: Sets environment variables and the working directory for test2.
7. **Port Exposure**: Exposes port 22 for SSH.
8. **Service Startup**: Configures the container to start the SSH service and ensures log files are copied to a specific directory on exit.

The above configuration ensure that the honeypot mimics the user environment. Every command runs inside the docker container can be logged and can be analyzed for future references

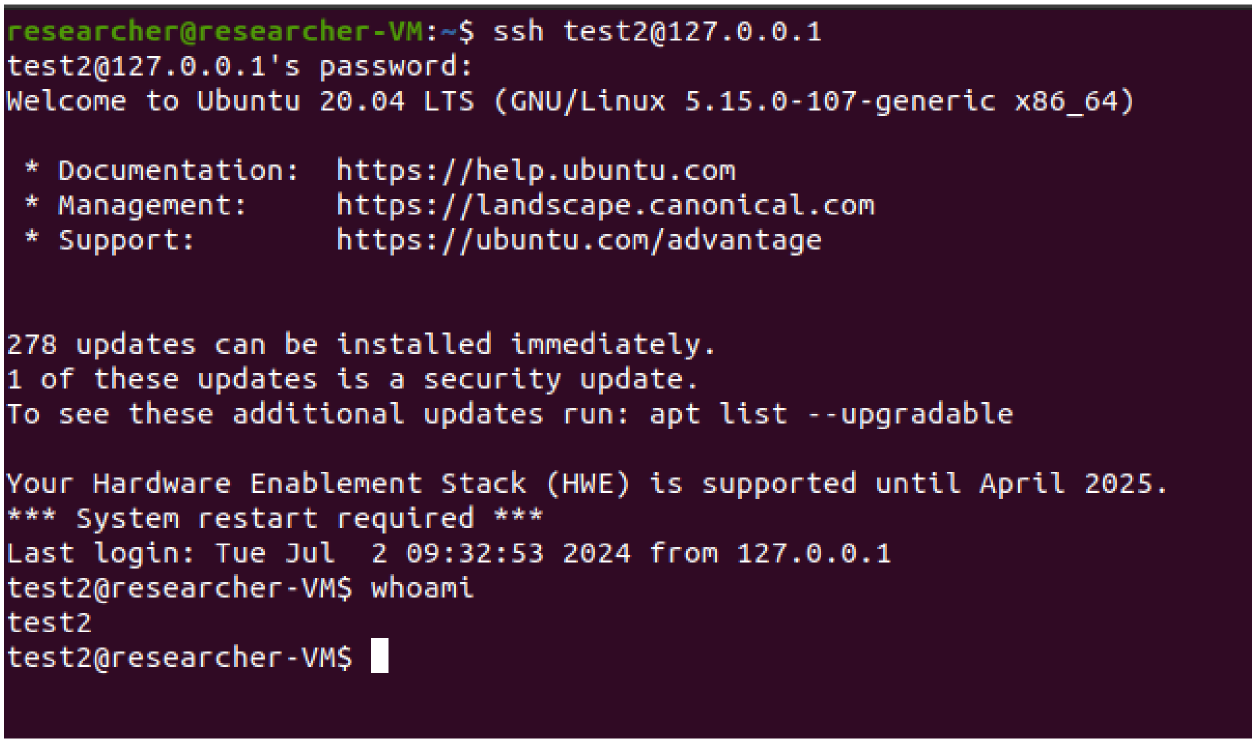
Evaluation

Details of the experiments and setup

The framework was built in an ubuntu 20.04 operating system. A test user named `test2` was created to simulate the attack scenario. When the user test2 is login to the system the user will load the script shell_1.0.sh instead of the traditional /bin/bash. The script will process commands before executed by /bin/bash hence monitoring the commands in real-time. Bash script and docker is used to build the framework. And future addons will be auto mated user behavior analysis instead of traditional engine to match the commands and a centralized dashboard to monitor the behavior of honeypot using Kibana and ELK stack.

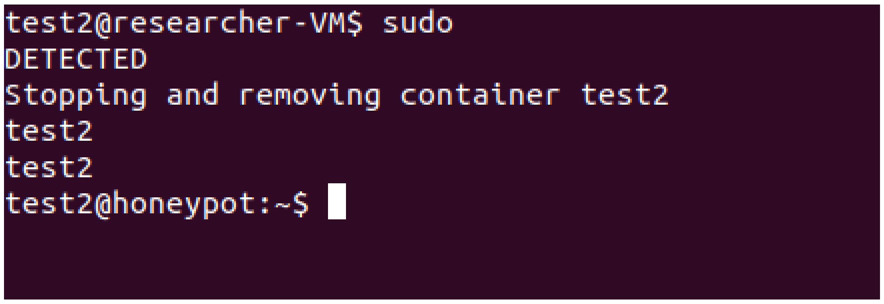
Proof of Concept

User logs into the system (test2:test2)



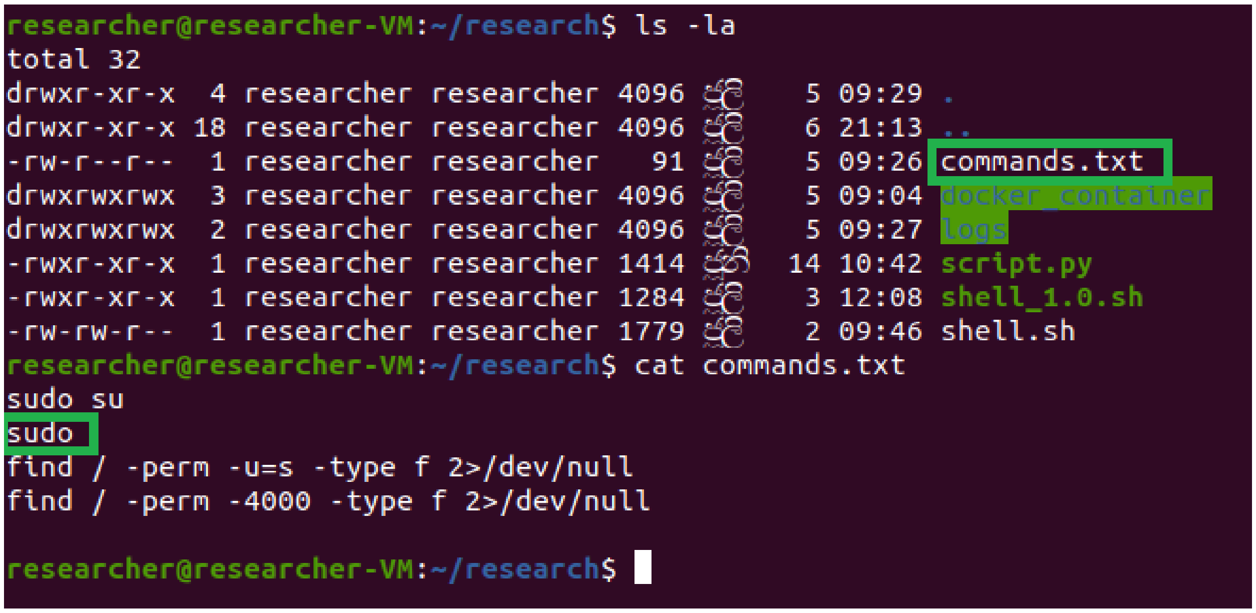
```
researcher@researcher-VM:~$ ssh test2@127.0.0.1
test2@127.0.0.1's password:
Welcome to Ubuntu 20.04 LTS (GNU/Linux 5.15.0-107-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage


278 updates can be installed immediately.
1 of these updates is a security update.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.
*** System restart required ***
Last login: Tue Jul  2 09:32:53 2024 from 127.0.0.1
test2@researcher-VM$ whoami
test2
test2@researcher-VM$
```

When the user enters command `sudo` as this is included in the commands.txt it will trigger the honeypot and redirect the user to the honeypot. Host name is renamed to honeypot as for demonstration purposes.



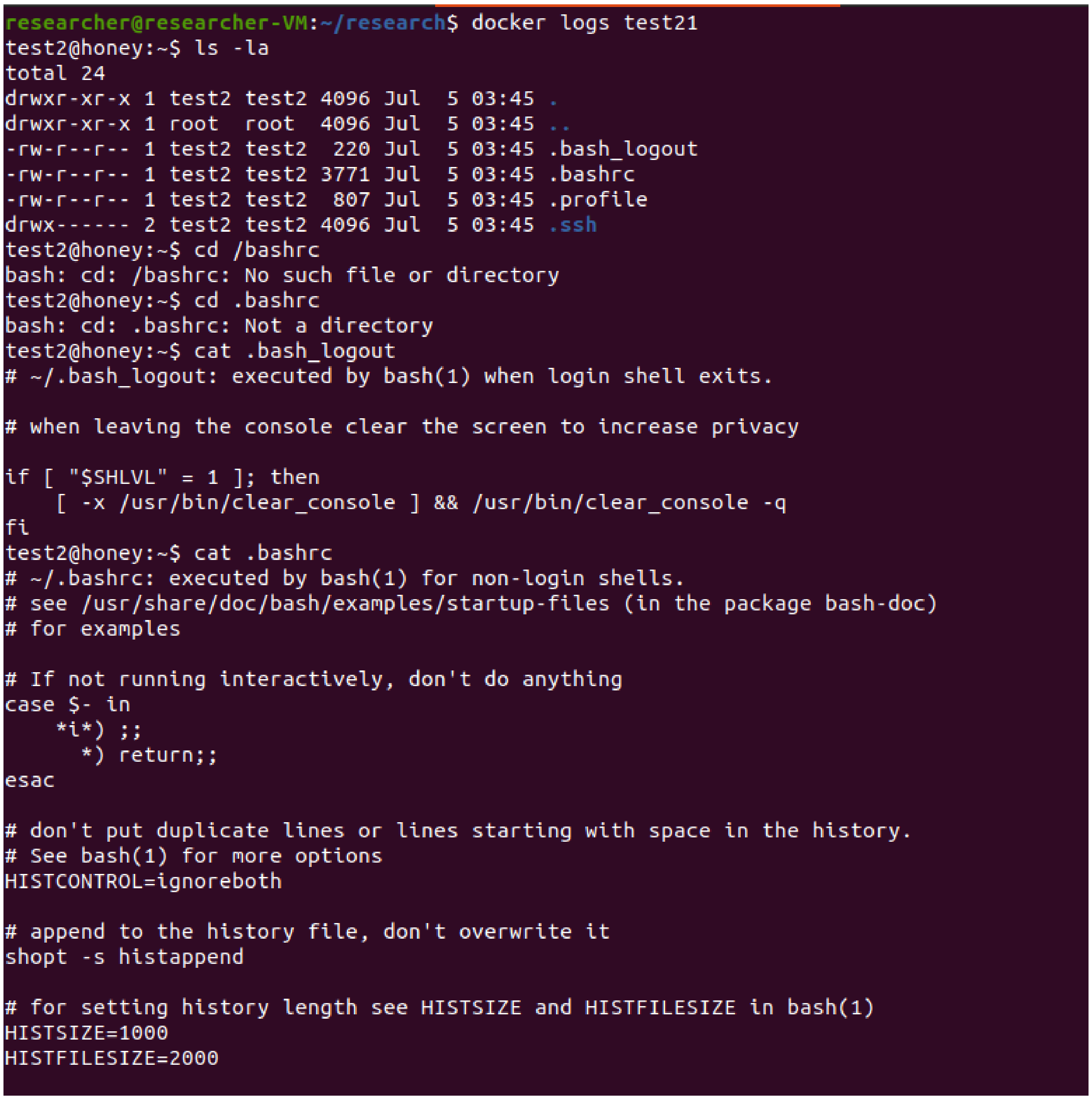
```
test2@researcher-VM$ sudo
DETECTED
Stopping and removing container test2
test2
test2
test2@honeypot:~$
```

Commands.txt file



```
researcher@researcher-VM:~/research$ ls -la
total 32
drwxr-xr-x  4 researcher researcher 4096 ජූනි   5 09:29 .
drwxr-xr-x 18 researcher researcher 4096 ජූනි   6 21:13 ..
-rw-r--r--  1 researcher researcher   91 ජූනි   5 09:26 commands.txt
drwxrwxrwx  3 researcher researcher 4096 ජූනි   5 09:04 docker_container
drwxrwxrwx  2 researcher researcher 4096 ජූනි   5 09:27 logs
-rwxr-xr-x  1 researcher researcher 1414 ජූලි  14 10:42 script.py
-rwxr-xr-x  1 researcher researcher 1284 ජූනි   3 12:08 shell_1.0.sh
-rw-rw-r--  1 researcher researcher 1779 ජූනි   2 09:46 shell.sh
researcher@researcher-VM:~/research$ cat commands.txt
sudo su
sudo
find / -perm -u=s -type f 2>/dev/null
find / -perm -4000 -type f 2>/dev/null

researcher@researcher-VM:~/research$ 
```

Once the user uses the honeypot we can analyze the workflow logs of the user (docker logs <imagename> is the command used)



```
researcher@researcher-VM:~/research$ docker logs test21
test2@honey:~$ ls -la
total 24
drwxr-xr-x 1 test2 test2 4096 Jul  5 03:45 .
drwxr-xr-x 1 root  root  4096 Jul  5 03:45 ..
-rw-r--r-- 1 test2 test2  220 Jul  5 03:45 .bash_logout
-rw-r--r-- 1 test2 test2 3771 Jul  5 03:45 .bashrc
-rw-r--r-- 1 test2 test2  807 Jul  5 03:45 .profile
drwx------ 2 test2 test2 4096 Jul  5 03:45 .ssh
test2@honey:~$ cd /bashrc
bash: cd: /bashrc: No such file or directory
test2@honey:~$ cd .bashrc
bash: cd: .bashrc: Not a directory
test2@honey:~$ cat .bash_logout
# ~/.bash_logout: executed by bash(1) when login shell exits.

# when leaving the console clear the screen to increase privacy

if [ "$SHLVL" = 1 ]; then
    [ -x /usr/bin/clear_console ] && /usr/bin/clear_console -q
fi
test2@honey:~$ cat .bashrc
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
    *i*) ;;
      *) return;;
esac

# don't put duplicate lines or lines starting with space in the history.
# See bash(1) for more options
HISTCONTROL=ignoreboth

# append to the history file, don't overwrite it
shopt -s histappend

# for setting history length see HISTSIZE and HISTFILESIZE in bash(1)
HISTSIZE=1000
HISTFILESIZE=2000
```

Strengths and Limitations

Strengths

Threat Reporting:

The honeypot is only concerned with privilege escalation, which takes place after an account is accessed. This all-encompassing method enables the identification of zero-day assaults and the capture of both simple and complex attack routes, offering a full picture of any internal threats.

Scalability:

The use of Docker for containerization makes it simple to scale up and down honeypot installations. Quickly deploying more honeypots in the form of containers is possible without requiring major adjustments to the underlying infrastructure. The system can adjust to growing traffic and novel threat environments because to its scalability.

Safety and Isolation:

This way, the threat actions are isolated from the production network and do not affect critical services since honeypots are isolated. This division is very useful in ensuring that the attackers cannot use the honeypots to launch other attacks in the company.

Realistic Attack Environment:

High-interaction honeypots include Ubuntu Linux 20. 04 offer a real life scenario for the attackers. These honeypots enable the analyst to monitor sophisticated attack vectors and privilege escalation tactics, which in turn provide better and more meaningful information regarding the attackers' actions..

Limitations

Resource intensive:
Above all, the high-interaction honeypot set up and management requires significant computational resources. Compared to low and medium interaction honeypots these honeypots require more CPU cycles, memory and storage as they are real systems in emulation.

Evading detection:
There is a possibility that the system effectiveness will be lowered if skilled attackers can recognize honeypots and avoid engaging with them. It has been established that honeypots can be easily discerned by attackers through methods like fingerprinting, and thus, fewer attacks are noted and the data gathered is incomplete.

Complexity of Management:
Another challenge is to keep a set of honeypots with different configuration levels and levels of interaction. There is a lot of effort and expertise needed to ensure that all the honeypots are properly configured, and managed, as well as monitored.

False Positives:
There is a possibility of producing a high number of false positives with the centralized logging system generating a huge amount of log data. Sometimes, it takes time and effort to differentiate between the legitimate actions and actual attempts at attack.

Conclusion

In this thesis, the deployment of a honeypot system based on Docker with the objective of detecting and preventing privilege escalation attempts, paying special attention to internal threats in the organization's network, was analyzed. The system used Docker for the container management which allowed the deployment of honeypots in an efficient and easy to manage manner. Thus, our work was aimed at addressing the increasing menace of internal threats which are often more challenging to prevent than external ones. To this end, we aimed at identifying and studying the conducts and aims of the malicious insiders and attackers who have reached some level of access by targeting privilege escalation strategies.

The docker honeypot system had the following strengths; it had a realistic attack environment, it could be easily scaled up, and it provided real time solution to the privilege escalation. The system was able to capture several types of attacks and provided valuable information concerning the activities of the attackers. However, it had some drawbacks: it demanded significant resources, the opponent could avoid detection by knowledgeable people, and it was not easy to manage.

In light of the above, it can be suggested that the use of the docker honeypot system can be an effective approach to improving the organization network's security. Thus, the organization's existing security measures and procedures can be improved based on the understanding of the attack data gathered. By offering a workable and scalable method for identifying and reducing internal threats through the use of containerization and cutting-edge honeypot technologies, this research advances the field of cybersecurity.

Future work

Subsequent research endeavors could concentrate on tackling the highlighted constraints, like enhancing the identification of proficient assailants who could identify honeypots and further mechanizing the administration and examination procedures to diminish intricacy and resource demands. Furthermore, extending the deployment of honeypots to cover external threats and additional attack vectors may offer enterprises a more complete security solution.

Additionally using a user behavior analysis tool will reduce the false positives and make the detection more sophisticated. By integrating elk stack to the docker the data can be extracted to a real time dashboard to monitor real time.

Appendices

Implementation

This chapter goes into great detail on how to implement the shell wrapper triggering mechanism and reroute the user to docker container. Each step's detailed procedure is covered in detail.

Docker-Based Honeypot Setup

This section provides a detailed description of the implementation process for the docker honeypot system designed to detect and mitigate privilege escalation attempts within the network. The implementation involves deploying a docker honeypots which mimics the user environment, leveraging Docker for containerization, and integrating logging and monitoring tools to capture and analyze attack data.

The step by step for installing docker in ubuntu

Step 1: Update Repository Packages

Ensure your package index and software repositories are up to date:

```
sudo apt update
```

Step 2: Install Docker from APT Repository

Install Docker using the `docker.io` package, which is available in the official Ubuntu repositories:

```
sudo apt install docker.io
```

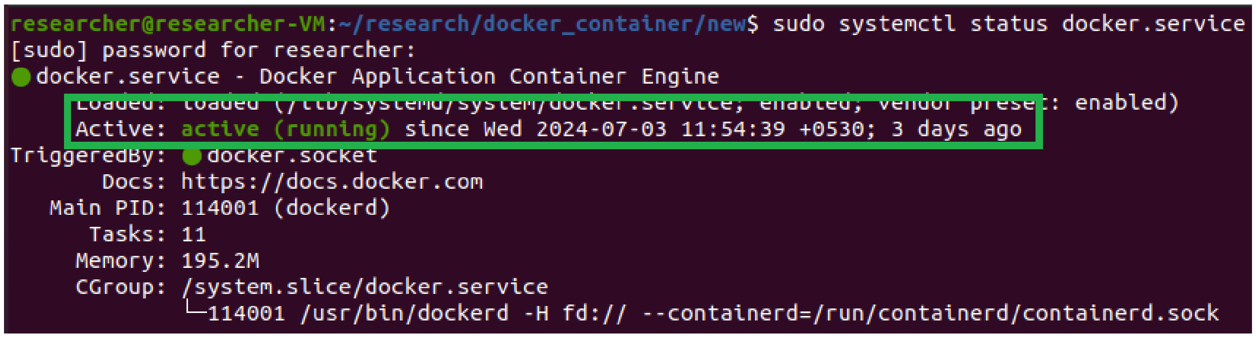
Step 3: Start and Enable Docker Service

Docker should start automatically after installation. If not, you can start and enable it with:

```
sudo systemctl start docker
sudo systemctl enable docker
```

Check docker status

```
sudo systemctl status docker
```

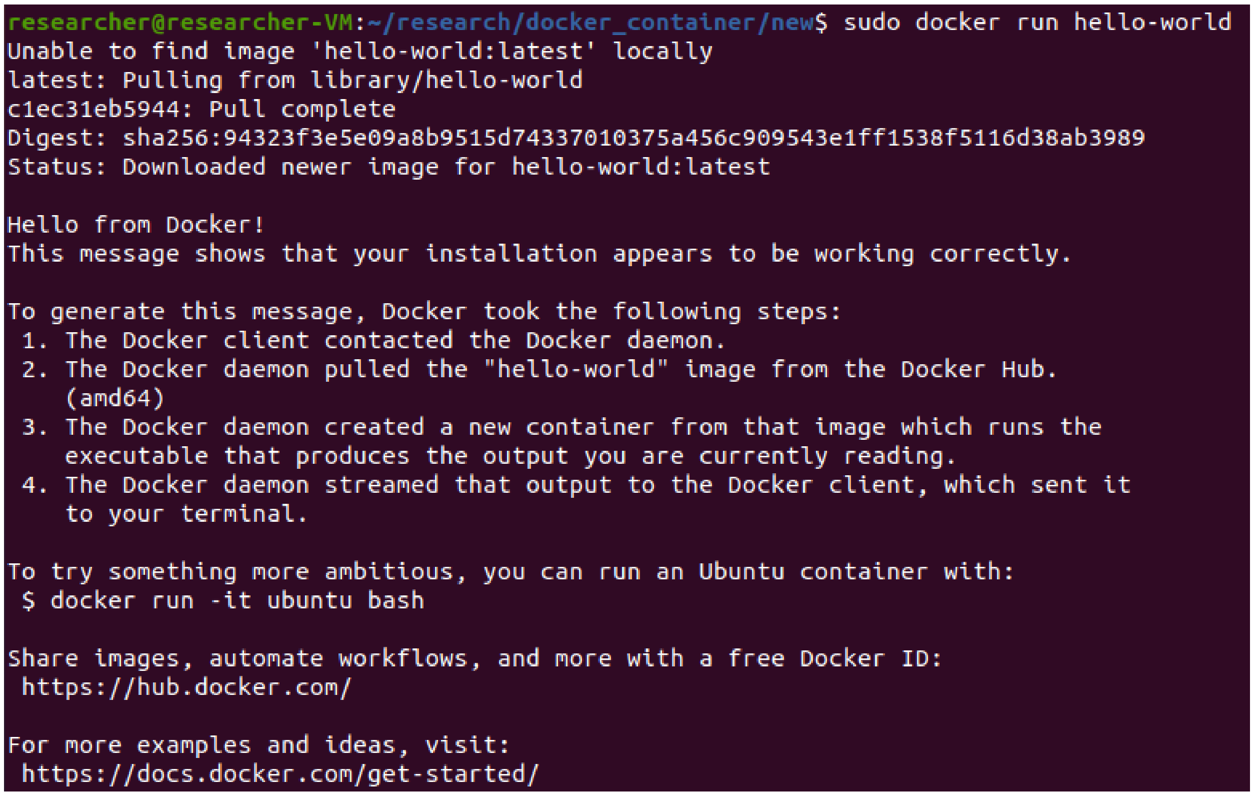


```
researcher@researcher-VM:~/research/docker_container/new$ sudo systemctl status docker.service
[sudo] password for researcher:
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2024-07-03 11:54:39 +0530; 3 days ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 114001 (dockerd)
      Tasks: 11
     Memory: 195.2M
     CGroup: /system.slice/docker.service
             └─114001 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
```

Step 4: Verify Docker Installation

Check that Docker has been installed correctly by running the `hello-world` image:

```
sudo docker run hello-world
```



```
researcher@researcher-VM:~/research/docker_container/new$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:94323f3e5e09a8b9515d74337010375a456c909543e1ff1538f5116d38ab3989
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Docker container creation

The docker file defines the nature of the container this is used to make the environment similar to the original user environment

Step 1: Create a file named `Dockerfile` and inset the below snippet. The snippet is used to mimic the user environment to the docker container

```
# Use an official Ubuntu image
FROM ubuntu:20.04

# Install necessary packages
RUN apt-get update && \
    apt-get install -y \
    apt-transport-https \
    ca-certificates \
    curl \
    gnupg \
    openssh-server \
    wget \
    iputils-ping \
    iproute2 \
    net-tools \
    && apt-get clean && \
    rm -rf /var/lib/apt/lists/*

# Set root password (change 'rootpassword' to the desired password)
RUN echo 'root:rootpassword' | chpasswd

# Create a non-root user
RUN adduser --disabled-password --gecos "" test2

# Set up SSH configuration
RUN mkdir -p /home/test2/.ssh && \
    chmod 700 /home/test2/.ssh && \
    chown test2:test2 /home/test2/.ssh
```

```
# Disable SSH banner and MOTD

RUN sed -i 's/^PrintMotd.*/PrintMotd no/' /etc/ssh/sshd_config && \
    sed -i 's/^PrintLastLog.*/PrintLastLog no/' /etc/ssh/sshd_config && \
    sed -i 's/session    optional     pam_motd.so/#session    optional     pam_motd.so/'
/etc/pam.d/sshd && \
    sed -i
's/session    optional     pam_lastlog.so/#session    optional     pam_lastlog.so/'
/etc/pam.d/sshd

    ENV PATH="/home/test2/bin:${PATH}"
    ENV USER=test2
    WORKDIR /home/test2


    # Set the working directory
    WORKDIR /home/test2



# Copy the sanitized passwd file
#COPY passwd_sanitized /etc/passwd

# Expose the SSH port
EXPOSE 22

# Start SSH service
#CMD ["/bin/bash", "-c", "/usr/sbin/sshd -D"]
CMD ["/bin/bash", "-c", "trap 'cp -r /var/log/*
/home/researcher/research/docker_container/new/logs' EXIT; /usr/sbin/sshd -D"]
```

Step 2: Building docker image

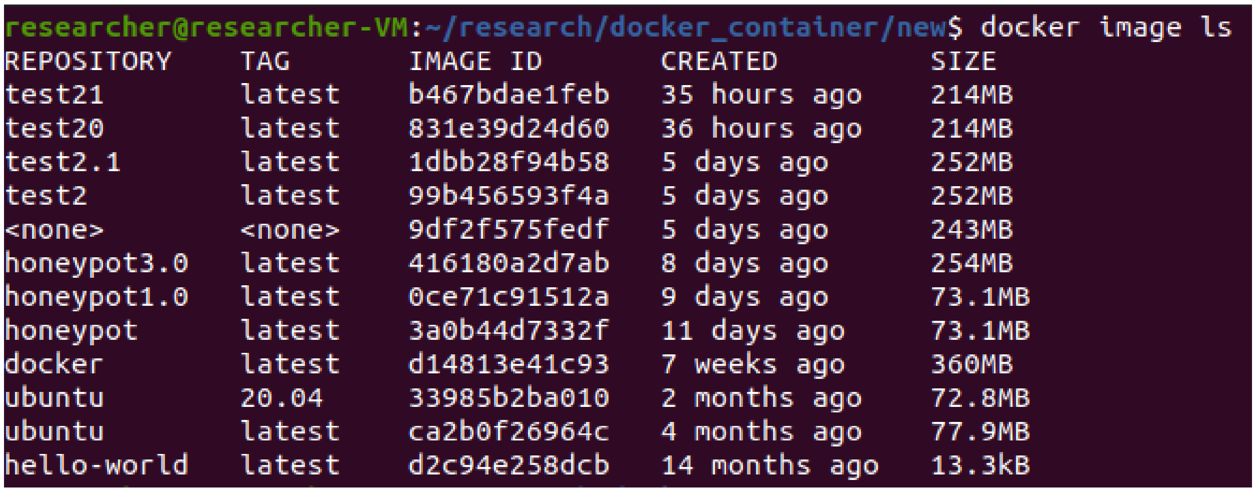
Once the docker file is made you can build the docker container using a base image in this case its ubuntu 20.04

```
sudo docker build -t <image name> .
```

It will create a base image for the docker file which can be used to spin up the container.

Step 3: Check the docker image

```
sudo docker image ls
```



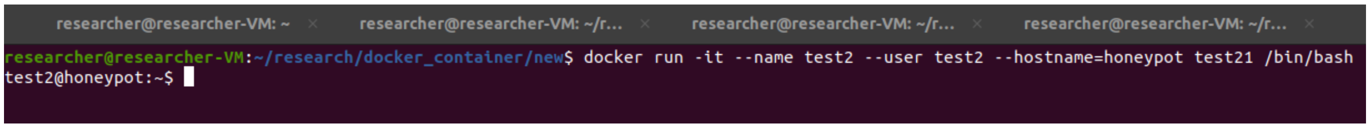
```
researcher@researcher-VM:~/research/docker_container/new$ docker image ls
REPOSITORY     TAG       IMAGE ID       CREATED         SIZE
test21         latest    b467bdae1feb   35 hours ago    214MB
test20         latest    831e39d24d60   36 hours ago    214MB
test2.1        latest    1dbb28f94b58   5 days ago      252MB
test2          latest    99b456593f4a   5 days ago      252MB
<none>         <none>    9df2f575fedf   5 days ago      243MB
honeypot3.0    latest    416180a2d7ab   8 days ago      254MB
honeypot1.0    latest    0ce71c91512a   9 days ago      73.1MB
honeypot       latest    3a0b44d7332f   11 days ago     73.1MB
docker         latest    d14813e41c93   7 weeks ago     360MB
ubuntu         20.04     33985b2ba010   2 months ago    72.8MB
ubuntu         latest    ca2b0f26964c   4 months ago    77.9MB
hello-world    latest    d2c94e258dcb   14 months ago   13.3kB
```

The created images will show up

Step 4: running the container

After the image is created we can run the container

```
docker run -it --name test2 --user test2 --hostname=honeypot test21 /bin/bash
```

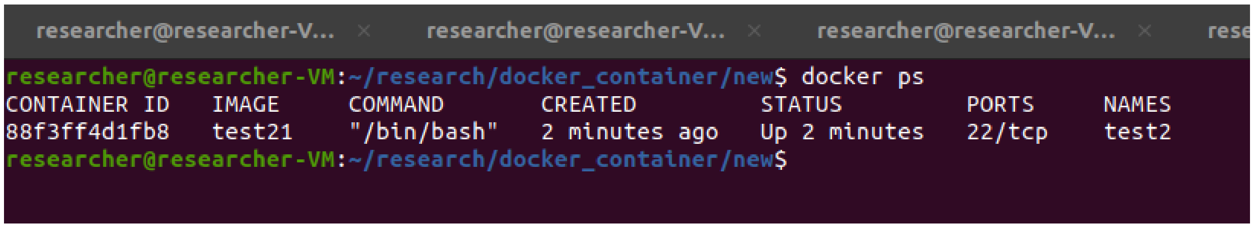


researcher@researcher-VM: ~
researcher@researcher-VM: ~/r...
researcher@researcher-VM: ~/r...
researcher@researcher-VM: ~/r...

```
researcher@researcher-VM:~/research/docker_container/new$ docker run -it --name test2 --user test2 --hostname=honeypot test21 /bin/bash
test2@honeypot:~$
```

As we can see the docker container is up and running we can confirm by below command as well

```
docker ps
```



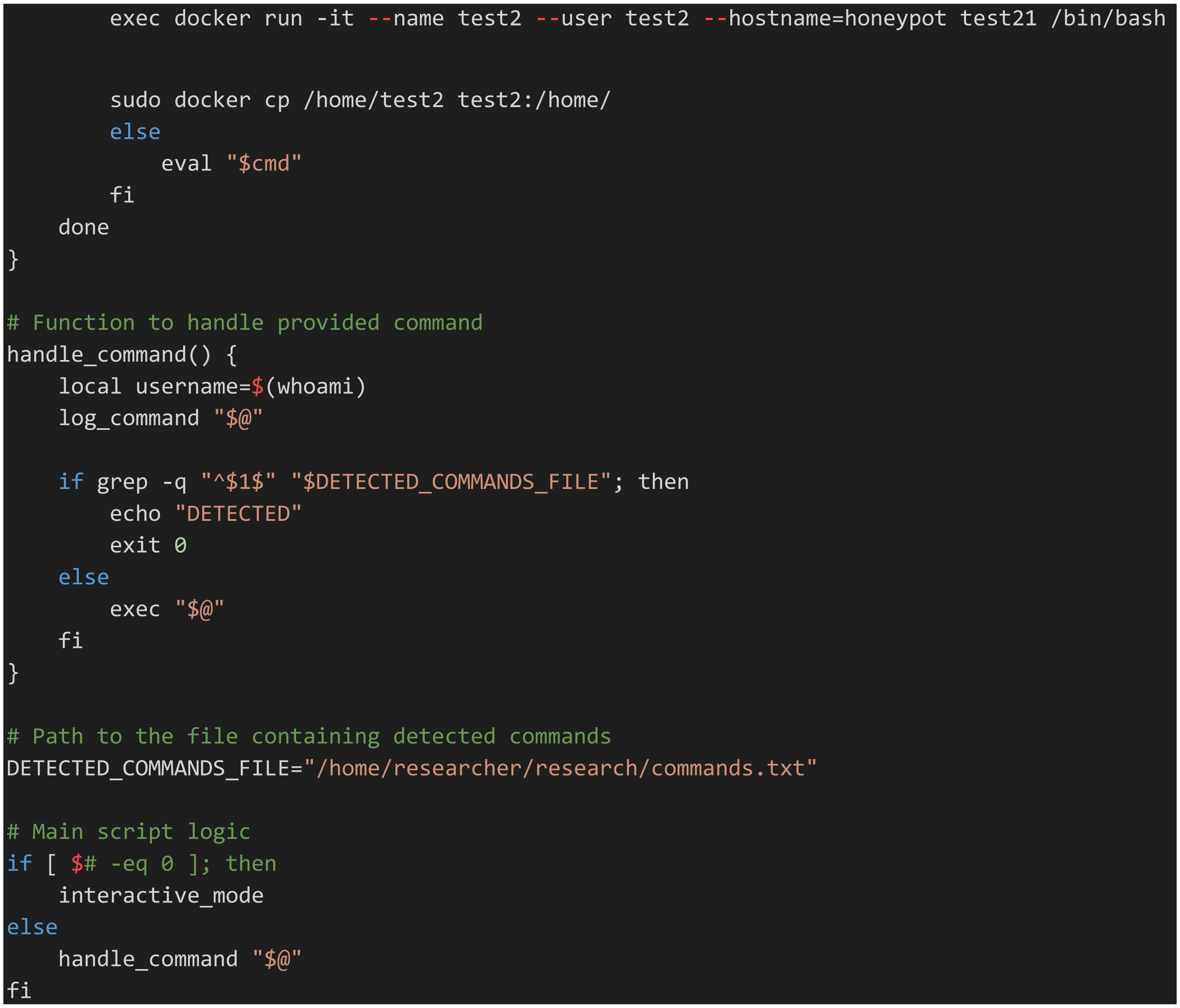
researcher@researcher-V...
researcher@researcher-V...
researcher@researcher-V...
rese

```
researcher@researcher-VM:~/research/docker_container/new$ docker ps
CONTAINER ID   IMAGE     COMMAND       CREATED         STATUS         PORTS     NAMES
88f3ff4d1fb8   test21    "/bin/bash"   2 minutes ago   Up 2 minutes   22/tcp    test2
researcher@researcher-VM:~/research/docker_container/new$
```

The above steps will setup the base honeypot which the threat actors will be redirected to.

Shell wrapper

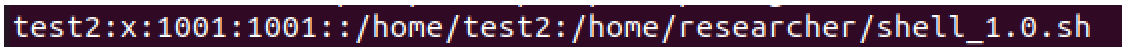
```
#!/bin/bash

# Function to intercept and log command
log_command() {
    local username=$(whoami)
    echo "[$username] Command intercepted: $@" >>
/home/researcher/research/logs/shell_logs.log #/var/log/shell_logs.log
}

# Function to handle interactive mode
interactive_mode() {
    local username=$(whoami)
    local PS1="$username@$HOSTNAME$ "

    while true; do
        read -p "$PS1" cmd
        log_command "$cmd"

        if [ "$cmd" = "exit" ]; then
            echo "Exiting wrapper..."
            break
        fi

        if grep -Fxq "$cmd" "$DETECTED_COMMANDS_FILE"; then
            echo "DETECTED"
        if docker ps -f name=test2 -q >/dev/null; then
            echo "Stopping and removing container test2"
            docker stop test2
            docker rm test2
        else
            echo "Container test2 is not running"
        fi
```



```
        exec docker run -it --name test2 --user test2 --hostname=honeypot test21 /bin/bash


        sudo docker cp /home/test2 test2:/home/
        else
            eval "$cmd"
        fi
    done
}

# Function to handle provided command
handle_command() {
    local username=$(whoami)
    log_command "$@"

    if grep -q "^$1$" "$DETECTED_COMMANDS_FILE"; then
        echo "DETECTED"
        exit 0
    else
        exec "$@"
    fi
}

# Path to the file containing detected commands
DETECTED_COMMANDS_FILE="/home/researcher/research/commands.txt"

# Main script logic
if [ $# -eq 0 ]; then
    interactive_mode
else
    handle_command "$@"
fi
```

Step 1: Save this snippet as shell.sh

Step2: Edit the /etc/passwd file on host to start up the wrapper for intended users



```
test2:x:1001:1001::/home/test2:/home/researcher/shell_1.0.sh
```

Stpe3: Grant user permission to execute the script

Sudo chmod +x shell_1.0.sh

Create database

Step 1: create a file named commands.txt

Step 2: Include all the commands to be detected to trigger redirrection

References

[1]M. Ansah, “Gov’t of Ghana website hacked | citifmonline,” citifmonline. 2015.

[2] NA, “KNUST’s Official Website Hacked!.,” 233 Live News, 2015. [Online]. Available: https://233livenews.wordpress.com/2015/11/24/knust s-official-website-hacked/. [Accessed: 20-Apr-2016].

[3] M. A. Lihet and V. Dadarlat, “How to build a honeypot System in the cloud,” in 2015 14th RoEduNet International Conference - Networking in Education and Research (RoEduNet NER), 2015, pp. 190–194.

[4] N. Kuze, S. Ishikura, T. Yagi, D. Chiba, and M. Murata, “Detection of vulnerability scanning using features of collective accesses based on information collected from multiple honeypots,” in NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium, 2016, pp. 1067–1072.

[5] J. Zhai and K. Wang, “Research on applications of honeypot in Campus Network security,” in Proceedings of 2012 International Conference on Measurement, Information and Control, 2012, vol. 1, pp. 309–313.

[6] S. Kumar, R. Sehgal, and J. S. Bhatia, “Hybrid honeypot framework for malware collection and analysis,” in 2012 IEEE 7th International Conference on Industrial and Information Systems (ICIIS), 2012, pp. 1–5.

[7] I. S. Kim and M. H. Kim, “Agent-based honeynet framework for protecting servers in campus networks,” IET Inf. Secur., vol. 6, no. 3, pp. 202– 211, 2012.

[8] W. Z. A. Zakaria, F. M. Maksom, and K. Abdullah, “Observing the presence of mobile malwares using low-interaction honeypot,” in 2016 IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE), 2016, pp. 117–121.

[9] U. Upadhyay and G. Khilari, “SQL injection avoidance for protected database with ASCII using SNORT and HONEYPOT,” in 2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), 2016, pp. 596–599.

[10] Y. L. Tsai, B. Y. Lee, and J. G. Chang, “Automated Malware Analysis Framework with Honeynet Technology in Taiwan Campuses,” in 2012 IEEE 18th International Conference on Parallel and Distributed Systems, 2012, pp. 724–725.

[11] Z. Zhan, M. Xu, and S. Xu, “Characterizing Honeypot-Captured Cyber Attacks: Statistical Framework and Case Study,” IEEE Trans. Inf. Forensics Secur., vol. 8, no. 11, pp. 1775–1789, 2013.

[12] A. M. Leonard, H. Cai, K. K. Venkatasubramanian, M. Ali, and T. Eisenbarth, “A honeypot system for wearable networks,” in 2016 IEEE 37th Sarnoff Symposium, 2016, pp. 199–201.

[13] N. M. Danchenko, A. O. Prokofiev, and D. S. Silnov, “Detecting suspicious activity on remote desktop protocols using Honeypot system,” in 2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), 2017, pp. 127–128.

[14]Titarmare, N. (no date) *(PDF) an overview of honeypot systems*. Available at: https://www.researchgate.net/publication/332113726_An_Overview_of_Honeypot_Systems (Accessed: 04 July 2024).