DS Journal of Cyber Security
Volume 2 Issue 4, 10-18, Oct - Dec 2024
ISSN: 2584-0665/ https://doi.org/10.59232/CYS-V2I4P102

Original Article

Analyzing Attacker's Data through Honeypot

K. Dhivya[1], G.K. Dharan[2*], T. Kowshik[3], N. Logha Surya[4], M. Mihaash Dharan[5]

[1,2,3,4,5]*Department of Computer Science and Engineering (Cyber Security), Sri Shakthi Institute of Engineering And Technology, Tamilnadu, India.*

[2]dharangandhi23cys@srishakthi.ac.in

Received: 03 October 2024; Revised: 09 November 2024; Accepted: 26 November 2024; Published: 21 December 2024

Abstract - This paper details a honeypot deployment leveraging Cowrie on an Ubuntu system to detect and analyse malicious behaviour. Honeypots are specialized systems designed to mimic vulnerable network assets, attracting attackers and capturing their activities for analysis. In this setup, Cowrie, a medium-interaction honeypot, simulates an SSH and Telnet interface, allowing attackers to engage with the system. Doing so captures each command entered, enabling a detailed examination of intrusion techniques. The combination of Honeyd and Cowrie offers a dual-layered approach, with Honeyd creating a network of virtual hosts and Cowrie focusing on interaction with attackers. This system logs unauthorized access attempts, records attacker commands, and analyses strategies, revealing valuable insights into emerging cyber threats. By studying these interactions, security professionals can enhance defensive strategies, anticipating tactics used in real-world attacks. The collected data aids in refining proactive security measures, fostering a robust security posture. The results underscore the significance of honeypots in cybersecurity research, making them a valuable tool in strengthening network defences against evolving threats.

Keywords - Honeypot, Intrusion Detection, Cowrie, Telnet, Botnets.

1. Introduction

This project is based on the implementation and analysis of honeypot systems, which are a serious module in the field of cybersecurity. Honeypots are trap systems positively designed to lure attackers by copying real, vulnerable systems to detect, nurse, and analyse malicious activity. By acting as tricks, honeypots can capture thorough information about cybercriminals' attack methods, tools, and plans. The research focuses on understanding how honeypots can effectively enhance the security of vulnerable systems, specifically in contexts like Internet of Things (IoT) devices and network services like SSH. As IoT devices become increasingly dominant in everyday life, their security has become a growing fear.

Tok (2019) [1] confers how the Mirai malware, which targets IoT systems, uses weak security configurations to compromise large numbers of devices and launch botnet attacks. This has painted the need for more advanced security measures, including deploying honeypots to attract and study attackers targeting IoT networks. Honeypots can help identify how attackers compromise these devices, offering valuable data for developing stronger, defined strategies.

The need for effective honeypot systems is further emphasized by the prevalence of brute-force password-guessing attacks, particularly against SSH services. Owens (2008) [3] illustrates how attackers often exploit weak or default passwords to gain unauthorized access to SSH servers, making them one of the most common attack vectors for cybercriminals. A honeypot system simulating SSH services can be a powerful tool in capturing such attacks and studying attacker behaviour in real time. Melese and Avadhani (2016) [5] show how honeypots can be deployed specifically to capture attacks targeting SSH, providing key insights into attackers' tools and techniques to break into systems. This approach helps to build a profile of attacker behaviour, revealing the tactics and tools used in successful and unsuccessful attack attempts. Furthermore, Dal (2019) [9] emphasizes that honeypots can simulate IoT environments, making them essential for studying attacks targeting such systems. This is particularly relevant given the increasing sophistication of attacks aimed at IoT devices.

Additionally, Kheirkhah et al. (2013) [14] highlight the role of honeypots in capturing and studying SSH attacks, while Johnson (2017) [15] demonstrates how honeypot systems can be effectively implemented in real-world scenarios to analyse network attacks. In this project, the implementation and analysis of honeypots will provide valuable insights into how attackers exploit faintness, helping to shape more real defence strategies against emerging threats in the cyber landscape. By perusing attacker tactics through honeypots, this research aims to contribute to the ongoing efforts to secure IoT devices and old network services from gradually classy cyberattacks.

2. Literature Survey

Honeypots play a crucial role in enhancing the detection of malicious activities in SSH systems. A significant study on attacker behavior following SSH compromises highlighted key patterns of attacker actions, specifically focusing on how attackers adapt after gaining unauthorized access. By profiling these behaviors, researchers help shape effective strategies for early detection and rapid response. Ramsbrock, Berthier, and Cukier (2007) demonstrate how understanding these post-compromise actions is essential for developing defensive measures to mitigate evolving attack tactics targeting SSH systems [2].

Understanding the methods and passwords attackers use to address brute-force SSH attacks is crucial. Research focused on brute-force SSH attacks provides an in-depth analysis of how attackers exploit weak password policies to infiltrate systems. Such research helps identify vulnerable access points and contributes to strengthening security measures. Owens (2008) emphasizes the importance of multi-factor authentication and improved password management to mitigate the risk of brute-force attacks and safeguard SSH systems from unauthorized access [3].

Securing cloud environments presents a unique challenge due to the cloud infrastructure's density and shared nature. Real security in cloud computing requires understanding vulnerabilities that cyber attackers exploit to gain unauthorized access or cause system troubles. Al Awadhi, Salah, and Martin (2013) explored the security risks in cloud environments, offering valuable perceptions of the weaknesses that must be addressed. Their work stresses the importance of merging traditional security trials with modern solutions to enhance cloud structure protection [6].

As IoT devices multiply, so do the security trials they present. Analysing malware's static and active behaviour, especially concerning IoT devices, helps notice classy attacks intended at these systems. Saxena, Bachhan, and Majumdar (2015) focused on how ARM-based boards can be utilized for such analyses, providing key insights into malware's interaction with embedded systems. Their findings underscore the necessity of using these systems for more effective detection of malware behaviors in IoT devices where traditional security measures are insufficient [7].

Honeypots have become a critical tool for understanding attack trends. These decoy systems simulate vulnerable environments to attract attackers, providing an invaluable resource for studying attack methodologies. Benzer and Arikan (2018) explored how honeypots serve as both a research and defense tool, contributing to better detection techniques and more secure systems. Their work highlights the growing importance of honeypots in modern cybersecurity, focusing on their ability to capture and analyze attack data to enhance system defenses [8].

Medium-interaction honeypots provide a balanced approach to monitoring and capturing attacks without exposing real systems to unnecessary risks. Researchers can understand attack strategies without compromising system integrity by analyzing the attack data gathered from such honeypots. Yüksel (2018) studied the effectiveness of medium-interaction honeypots and framed them as a practical solution to monitor attack patterns, offering valuable insights into attack tactics while maintaining system safety [10].

In response to the growing need for efficient network attack detection systems, Singh (2011) proposed a honeypot system that captures and analyzes network attack traffic. His system provides real-time insights into attack patterns, aiding in developing more effective defensive measures. Singh's work emphasizes the importance of dedicated honeypots to detect and understand various types of network attacks, helping improve overall network security by offering insights into attackers' methods [11].

Erdem, Kara, and Itkinci (2015) developed HoneyThing, a honeypot system specifically designed for IoT devices to address security challenges in IoT environments. Their research focuses on the need for customized honeypot solutions to simulate IoT systems and analyze the unique attacks targeting these devices. HoneyThing provides insights into attack strategies and helps fortify the security of IoT networks by identifying potential vulnerabilities and risks [12].

With the quick growth of IoT devices, the need for active security solutions has become more grave than ever. Razali, Muruti, Razali, Jamil, and Mansor (2019) reviewed IoT honeypots and their deployment strategies, highlighting their importance in the fight against IoT-targeted attacks. Their work provides an overview of IoT security's unique challenges. It emphasizes how honeypots designed for IoT environments can help researchers better understand attack patterns, ultimately improving security measures for IoT networks [13].

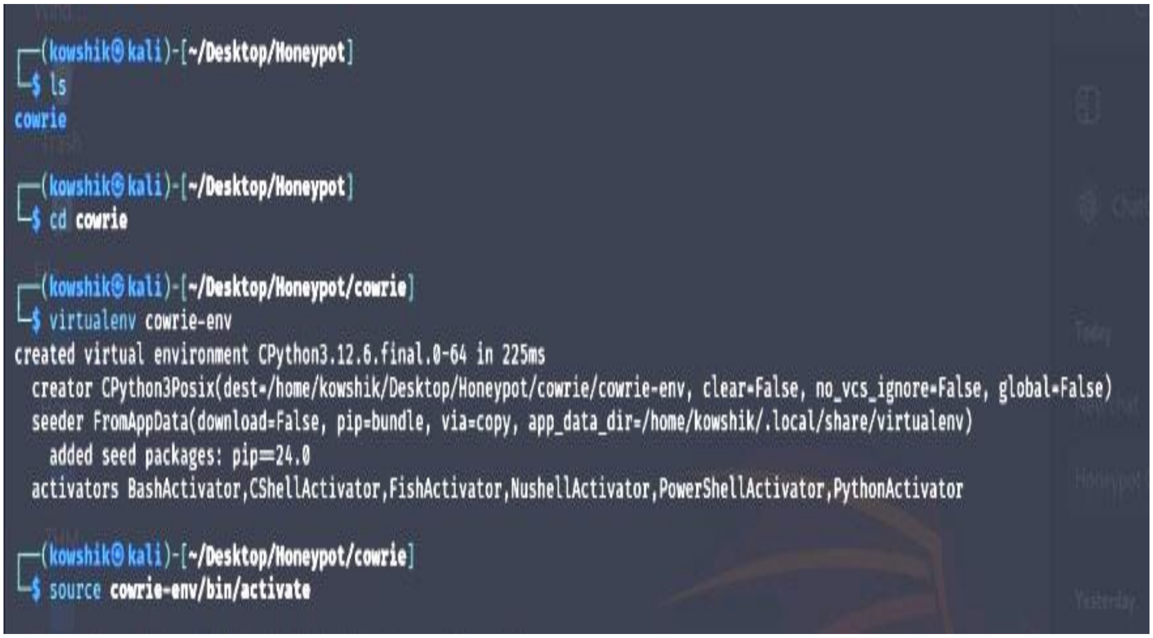
3. Proposed Honeypot

The methodology involves setting up a honeypot environment using both Honeyd and Cowrie. Honeyd is an unimportant Honeypot tool designed to simulate multiple computer-generated honeypot systems, each with unique services and weaknesses. On the other hand, Cowrie is a high-interaction honeypot designed to capture detailed command executions from attackers. The system is deployed on an Ubuntu server, with the following steps taken to implement the Honeypot:

- Install and configure Cowrie on the Ubuntu machine to simulate different network services.
- Set up Cowrie to capture and log SSH brute force attacks and file system interactions.
- Create fake directories and systems in Cowrie to emulate real-world server environments, attracting various types of attackers.
- Use logging and monitoring tools to capture all incoming traffic and activities on the Honeypot.
- Analise the captured data for attack patterns, attempting to detect credential stuffing, port scanning, and social engineering techniques.
- Data analysis includes examining attack logs, identifying the most common exploits, and evaluating the success of different honeypot configurations in detecting malicious activity.
- Environment setup and Configuration setup are shown in Figures 1 and 2, respectively.

4. Honeypot Design

The proposed system is a honeypot network security solution using the Cowrie tool to simulate a vulnerable server environment that mimics SSH and Telnet services. This Honeypot is designed to attract mean attackers and record their actions, providing valuable insights into realworld attack vectors and intruder behaviour.



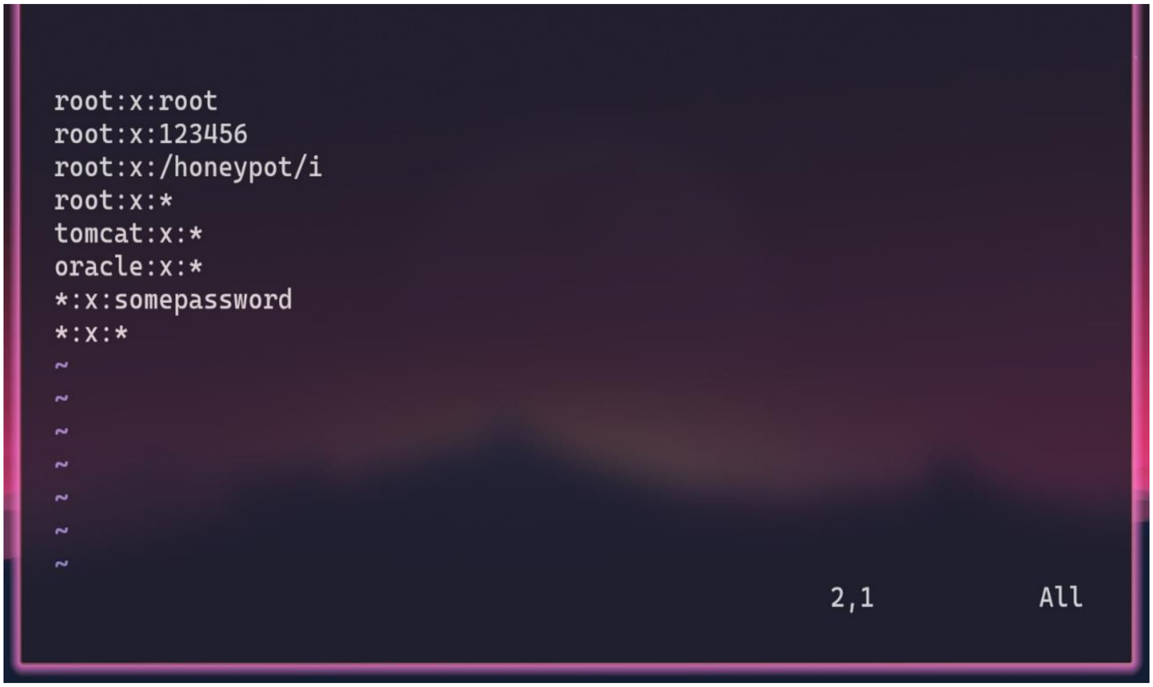
```
┌──(kowshik㉿kali)-[~/Desktop/Honeypot]
└─$ ls
cowrie

┌──(kowshik㉿kali)-[~/Desktop/Honeypot]
└─$ cd cowrie

┌──(kowshik㉿kali)-[~/Desktop/Honeypot/cowrie]
└─$ virtualenv cowrie-env
created virtual environment CPython3.12.6.final.0-64 in 225ms
  creator CPython3Posix(dest=/home/kowshik/Desktop/Honeypot/cowrie/cowrie-env, clear=False, no_vcs_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, via=copy, app_data_dir=/home/kowshik/.local/share/virtualenv)
    added seed packages: pip==24.0
  activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator

┌──(kowshik㉿kali)-[~/Desktop/Honeypot/cowrie]
└─$ source cowrie-env/bin/activate
```

Fig. 1 Environment setup



```
root:x:root
root:x:123456
root:x:/honeypot/i
root:x:*
tomcat:x:*
oracle:x:*
*:x:somepassword
*:x:*
~
~
~
~
~
~
~
                                     2,1          All
```

Fig. 2 Configuration setup

4.1. Honeypot Overview

The Cowrie honeypot will operate as a low-interaction honeypot, which means it will allow attackers to cooperate with a fake environment without giving them access to the actual original system. This setup is both secure and effective, as it prevents real harm to the host machine while still collecting essential data about potential interruptions.

4.2. *Honeypot Objectives*

The primary goal of the proposed system is to:

- Capture and log all attempted connections to the honeypot server, including
- Login attempts, executed commands, and file transfers.
- Analyze collected data to identify patterns, such as commonly used attack techniques, brute-force attempts, and IP sources.
- Gain insights into the tactics and tools used by attackers, which can help improve defensive measures on real production systems.

4.3. Honeypot Workflow

4.3.1. Attack Simulation

When an attacker attempts to connect to the Honeypot, they will quickly be presented with a fake SSH or Telnet login. Cowrie will emulate a basic filesystem and respond to commands in a way that mimics a real Linux environment, allowing the attacker to believe they are interacting with a real server.

4.3.2. Data Collection

All login attempts, commands executed, and files transferred during the session are recorded. Cowrie logs this data in real time, capturing the specific behavior and sequence of actions taken by the intruder.

4.3.3. Data Analysis

Logged data is regularly analyzed to identify patterns such as popular usernames and passwords used in brute-force attacks, IP addresses from which attacks originate, and the types of commands attackers attempt to execute. This information helps to understand attacker behavior and adapt future security measures.

The workflow of the Honeypot design is shown in Figure 3, respectively.

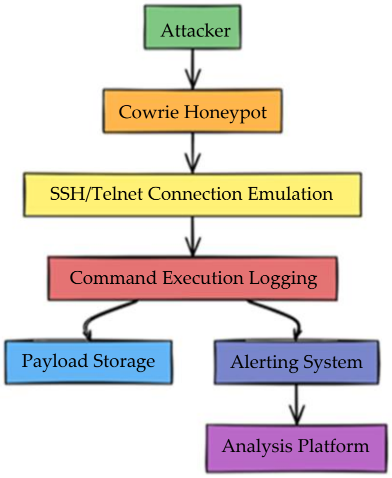


Fig. 3 Flow diagram

4.4. Expected Outcomes

By implementing this proposed system, we aim to collect detailed data on unauthorized access attempts, including typical attack patterns and the tools used by attackers. This data will be instrumental in enhancing the organization's security strategy, providing actionable insights into how attackers target similar systems in the real world.

5. Architecture Diagram

The architecture of the honeypot system is designed to simulate network services through the Honeypot and capture command activity through Cowrie. An architecture diagram illustrating this process should include:

- Honeypot simulating network services to attract attackers.
- Cowrie intercepting and logging commands.
- Data flow to the host system for storage and analysis

The architecture diagram of Honeypot is shown in Figure 4 respectively.

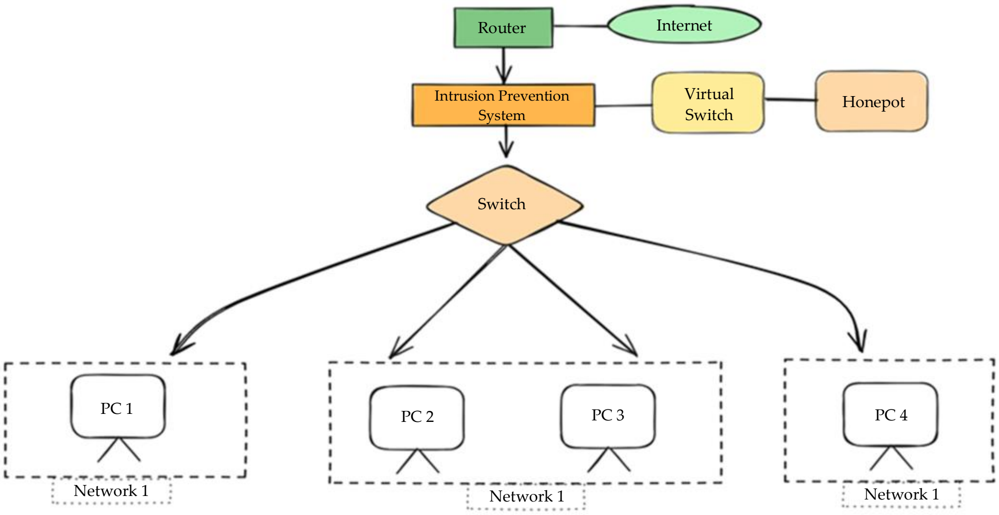


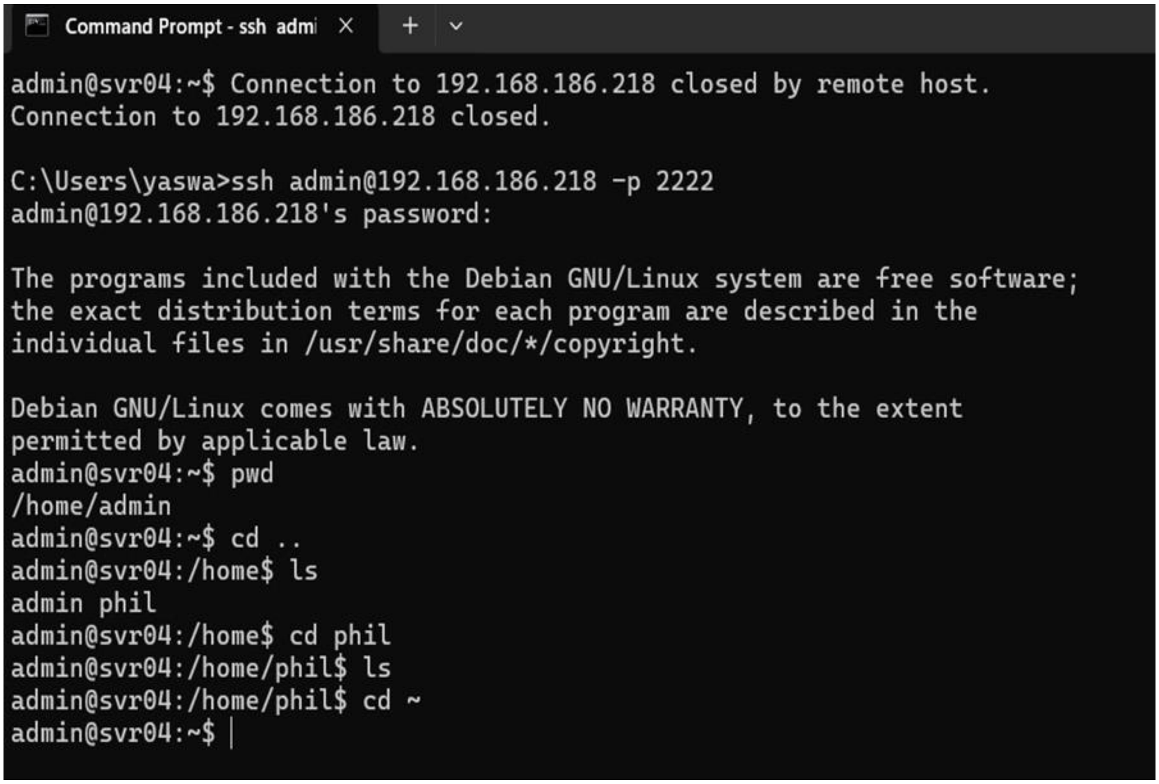
Fig. 4 Architecture diagram

This honeypot network architecture depicts an attacker from the internet attempting to access a protected LAN. The firewall, router, and honeypot components intercept and monitor suspicious traffic, while the Intrusion Detection System (IDS) analyze network activity to detect potential threats.

6. Results and Discussion

During the initial deployment of the Honeypot, multiple unauthorized access attempts were successfully captured, and various command sequences were logged within Cowrie. A detailed statistical analysis of these command logs revealed common patterns that align with reconnaissance and privilege escalation attempts-key indicators of attackers probing for weaknesses. Commands like ls, cat, and Sudo were frequently observed, suggesting attackers explored system directories, viewed critical files, and tested permissions to gain elevated access. By analyzing the frequency and sequence of these commands, clear patterns emerged, such as repeated attempts to list directory contents or access sensitive files, highlighting typical attack vectors and steps in the attacker's process.

This analysis not only emphasizes the areas attackers are most interested in, but it also provides valuable insights into specific vulnerabilities that require further hardening. For example, repeated attempts to use Sudo indicate a strong focus on privilege escalation, a critical area for strengthening access controls. Additionally, plotting command frequency or creating a graph of command types and their respective frequencies can visually illustrate which commands are most targeted by unauthorized users. This graphical representation would provide a quick, at-a-glance view of attacker priorities and behaviours, supporting a more strategic approach to improving security measures. When combined with proactive defences, these findings serve as an essential feedback loop for continuously adapting and enhancing the system's security posture against recurring and emerging threats. Connection request and Captured log details about the attack were shown in Figures 5 and 6 respectively.



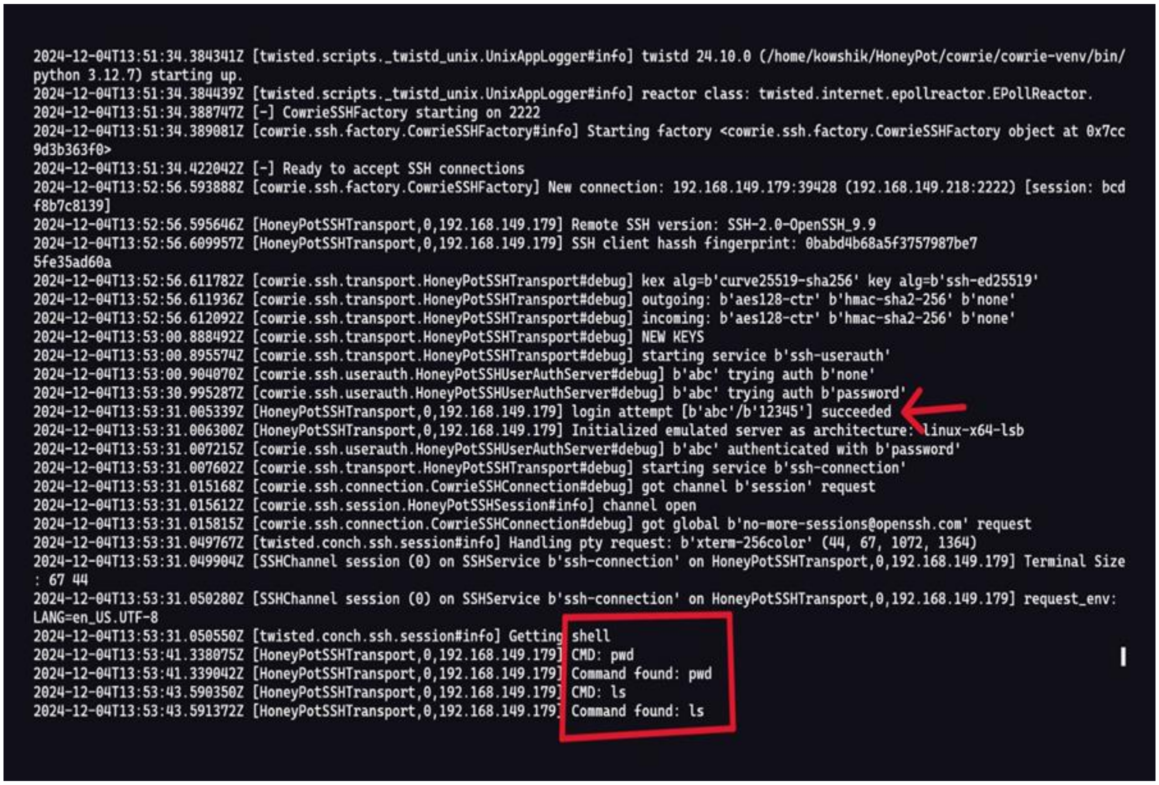
```
admin@svr04:~$ Connection to 192.168.186.218 closed by remote host.
Connection to 192.168.186.218 closed.

C:\Users\yaswa>ssh admin@192.168.186.218 -p 2222
admin@192.168.186.218's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
admin@svr04:~$ pwd
/home/admin
admin@svr04:~$ cd ..
admin@svr04:/home$ ls
admin phil
admin@svr04:/home$ cd phil
admin@svr04:/home/phil$ ls
admin@svr04:/home/phil$ cd ~
admin@svr04:~$ |
```

Fig. 5 Connection request



```
2024-12-04T13:51:34.384341Z [twisted.scripts._twistd_unix.UnixAppLogger#info] twistd 24.10.0 (/home/kowshik/HoneyPot/cowrie/cowrie-venv/bin/
python 3.12.7) starting up.
2024-12-04T13:51:34.384439Z [twisted.scripts._twistd_unix.UnixAppLogger#info] reactor class: twisted.internet.epollreactor.EPollReactor.
2024-12-04T13:51:34.388747Z [-] CowrieSSHFactory starting on 2222
2024-12-04T13:51:34.389081Z [cowrie.ssh.factory.CowrieSSHFactory#info] Starting factory <cowrie.ssh.factory.CowrieSSHFactory object at 0x7cc
9d3b363f0>
2024-12-04T13:51:34.422042Z [-] Ready to accept SSH connections
2024-12-04T13:52:56.593888Z [cowrie.ssh.factory.CowrieSSHFactory] New connection: 192.168.149.179:39428 (192.168.149.218:2222) [session: bcd
f8b7c8139]
2024-12-04T13:52:56.595646Z [HoneyPotSSHTransport,0,192.168.149.179] Remote SSH version: SSH-2.0-OpenSSH_9.9
2024-12-04T13:52:56.609957Z [HoneyPotSSHTransport,0,192.168.149.179] SSH client hassh fingerprint: 0babd4b68a5f3757987be7
5fe35ad60a
2024-12-04T13:52:56.611782Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] kex alg=b'curve25519-sha256' key alg=b'ssh-ed25519'
2024-12-04T13:52:56.611936Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] outgoing: b'aes128-ctr' b'hmac-sha2-256' b'none'
2024-12-04T13:52:56.612092Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] incoming: b'aes128-ctr' b'hmac-sha2-256' b'none'
2024-12-04T13:53:00.888492Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] NEW KEYS
2024-12-04T13:53:00.895574Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] starting service b'ssh-userauth'
2024-12-04T13:53:00.904070Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'abc' trying auth b'none'
2024-12-04T13:53:30.995287Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'abc' trying auth b'password'
2024-12-04T13:53:31.005339Z [HoneyPotSSHTransport,0,192.168.149.179] login attempt [b'abc'/b'12345'] succeeded
2024-12-04T13:53:31.006300Z [HoneyPotSSHTransport,0,192.168.149.179] Initialized emulated server as architecture: linux-x64-lsb
2024-12-04T13:53:31.007215Z [cowrie.ssh.userauth.HoneyPotSSHUserAuthServer#debug] b'abc' authenticated with b'password'
2024-12-04T13:53:31.007602Z [cowrie.ssh.transport.HoneyPotSSHTransport#debug] starting service b'ssh-connection'
2024-12-04T13:53:31.015168Z [cowrie.ssh.connection.CowrieSSHConnection#debug] got channel b'session' request
2024-12-04T13:53:31.015612Z [cowrie.ssh.session.HoneyPotSSHSession#info] channel open
2024-12-04T13:53:31.015815Z [cowrie.ssh.connection.CowrieSSHConnection#debug] got global b'no-more-sessions@openssh.com' request
2024-12-04T13:53:31.049767Z [twisted.conch.ssh.session#info] Handling pty request: b'xterm-256color' (44, 67, 1072, 1364)
2024-12-04T13:53:31.049904Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,0,192.168.149.179] Terminal Size
: 67 44
2024-12-04T13:53:31.050280Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,0,192.168.149.179] request_env:
LANG=en_US.UTF-8
2024-12-04T13:53:31.050550Z [twisted.conch.ssh.session#info] Getting shell
2024-12-04T13:53:41.338075Z [HoneyPotSSHTransport,0,192.168.149.179] CMD: pwd
2024-12-04T13:53:41.339042Z [HoneyPotSSHTransport,0,192.168.149.179] Command found: pwd
2024-12-04T13:53:43.590350Z [HoneyPotSSHTransport,0,192.168.149.179] CMD: ls
2024-12-04T13:53:43.591372Z [HoneyPotSSHTransport,0,192.168.149.179] Command found: ls
```

Fig. 6 Captured log detail about the attack

7. Conclusion and Future Enhancement

A honeypot system organized with Honeyd and Cowrie on an Ubuntu platform has been confirmed to be a powerful tool for understanding and opposing real-world cyber threats. This dual setup captures a broad range of attack behaviours, providing invaluable insights into the types of Tactics, Techniques, and Procedures (TTPs) attackers use. Honeyd, a low-interaction honeypot, emulates various network topologies and services, attracting attackers' initial scans and probes. By simulating different IP addresses and operating systems, it collects data on the reconnaissance phase, highlighting common targets and frequent attack types. Meanwhile, Cowrie, a high-interaction SSH and Telnet honeypot, simulates a realistic command shell environment, allowing attackers to execute commands. Cowrie logs each action, capturing detailed data on the attacker's behaviour, including attempts to escalate privileges, download files, or install malware. This configuration allows for comprehensive behavioural analysis and threat intelligence collection, revealing the specific commands, payloads, and methods attackers use to exploit vulnerabilities. The combination of Honeyd and Cowrie enhances training and testing efforts, as security teams can observe real attack methods in a controlled environment and practice response protocols. Additionally, Cowrie captures malware samples when attackers attempt to download files, enabling secure analysis of these samples to create new detection signatures. The insights gathered from this honeypot setup are crucial for strengthening Intrusion Detection Systems (IDS), refining firewall rules, and hardening the security posture of an organization. This system, therefore, not only detects but also anticipates and mitigates potential attacks by informing more resilient security practices, ultimately boosting the organization's defences against evolving cyber threats.

Several enhancements are planned to increase the functionality and impact of the Honeypot. Firstly, creating fake directories within Cowrie will provide additional layers of deception, encouraging attackers to reveal further behaviors and command sequences. Additionally, enabling remote access to the honeypot machine over any network will allow for flexible monitoring and control, making the system accessible from various locations and furthering its potential as a research and security tool.

References

[1] Mevlüt Serkan Tok, "Internet of Things Botnets: A Case Study on Mirai Malware," *TOBB University of Economics and Technology*, Graduate School of Engineering and Science, 2019. [Google Scholar] [Publisher Link]

[2] Daniel Ramsbrock, Robin Berthier, and Michel Cukier, "Profiling Attacker Behavior Following SSH Compromises," *37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, Edinburgh, UK, pp. 119-124, 2007. [CrossRef] [Google Scholar] [Publisher Link]

[3] Jim Owens, and Jeanna Matthews, "A Study of Passwords and Methods Used in Brute-Force SSH Attacks," *USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, 2008. [Google Scholar] [Publisher Link]

[4] Ioannis Koniaris, Georgios Papadimitriou, and Petros Nicopolitidis, "Analysis and Visualization of SSH Attacks Using Honeypots," *EuroCon 2013*, Zagreb, Croatia, pp. 65-72, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[5] Solomon Z. Melese, and P.S. Avadhani, "Honeypot System for Attacks on SSH Protocol," *International Journal of Computer Network and Information Security*, vol. 8, no. 9, pp. 19-26, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[6] Eman Al Awadhi, Khaled Salah, and Thomas Martin, "Assessing the Security of the Cloud Environment," *2013 7th IEEE GCC Conference and Exhibition (GCC)*, Doha, Qatar, pp. 251-256, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[7] Utkarsh Saxena, Om Prakash Bachhan, and Rana Majumdar, "Static and Dynamic Malware Behavioral Analysis Based on ARM Based Board," *2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, India, pp. 272-277, 2015. [Google Scholar] [Publisher Link]

[8] S.M. Arikan, and R Benzer, "A Security Trend: Honeypot," *Acta Infologica*, vol. 2, no. 1, pp. 1-11, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[9] F. Dal, "*A Case Study on the Analysis of Attacks against Internet of Things Systems with Honeypot Systems*," Master Thesis, 2019.

[10] Seda Yüksel, "*Analyzing the Medium-Interaction Honeypot: A Case Study*," Thesis, The Graduate School of Natural and Applied Sciences of Çankaya University, 2018. [Google Scholar] [Publisher Link]

[11] Abhay Nath Singh, and R.C. Joshi, "A Honeypot System for Efficient Capture and Analysis of Network Attack Traffic," *2011 International Conference on Signal Processing, Communication, Computing and Networking Technologies*, Thuckalay, India, pp. 514-519, 2011. [CrossRef] [Google Scholar] [Publisher Link]

[12] Ömer Erdem, "*Honeything: A Trap System for the Internet of Things*," Thesis, Department of Information Security Engineering, İstanbul Şehir University, 2015. [Google Scholar] [Publisher Link]

[13] Mohamad Faiz Razali et al., "IoT Honeypot: A Review from Researcher's Perspective," *2018 IEEE Conference on Application, Information and Network Security (AINS)*, Langkawi, Malaysia, pp. 93-98, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[14] Esmaeil Kheirkhah et al., "An Experimental Study of SSH Attacks by Using Honeypot Decoys," *Indian Journal of Science and Technology*, vol. 6, no. 12, pp. 1-12, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[15] J.M. Johnson, "*Security*," Naval Postgraduate, pp. 1-55, 2017.