



Identifying the True Origin of IP Addresses Concealed by VPN Proxy Services

Aashik Naggie V¹, Mr. S. Vignesh²

¹MSc Computer Science Rathinam College of Arts and Science Coimbatore, Aashiknaggie282@gmail.com

²Department of Computer Science Rathinam College of Arts and Science Coimbatore

ABSTRACT :

In contemporary cybercrime landscapes, the utilization of pseudonymous IP addresses has become a prevalent strategy among malevolent actors seeking to obscure their true digital footprints and bolster their anonymity. This practice poses significant challenges to law enforcement and cybersecurity professionals tasked with identifying and prosecuting such offenders. To address this issue, our team is undertaking the development of a comprehensive solution designed to effectively distinguish between genuine IP addresses and those originating from proxies or virtual private networks (VPNs). The primary objective of our initiative is twofold: first, to create a mechanism capable of accurately identifying whether an IP address is authentic or derived from a proxy or VPN; and second, to implement functionality enabling the tracking of genuine IP addresses and the provision of detailed information regarding the geographic location of VPN users. By achieving these goals, our solution aims to significantly enhance the capabilities of law enforcement agencies and cybersecurity experts in combating cybercrime and holding perpetrators accountable for their actions. Cybercriminals employ a wide array of sophisticated tools and techniques to conceal their digital identities and evade detection. Among these tactics, the use of proxy servers and VPNs stands out as particularly effective in obfuscating the origin of malicious activities. Proxy servers act as intermediaries between a user's device and the internet, masking the user's true IP address and routing internet traffic through a separate server. Similarly, VPNs encrypt internet traffic and route it through remote servers, thereby concealing the user's true IP address and location. Our solution acknowledges the challenges posed by these evasion tactics and endeavors to circumvent them by leveraging advanced algorithms and analytical methodologies. By analyzing patterns of network traffic and employing machine learning algorithms, our solution can effectively identify suspicious IP addresses associated with proxies or VPNs. Additionally, it can correlate this information with other data sources to uncover the genuine IP address concealed behind the proxy or VPN IP. The development of a robust solution capable of distinguishing between genuine IP addresses and those associated with proxies or VPNs represents a crucial step forward in the ongoing battle against cybercrime. By providing law enforcement and cybersecurity professionals with the means to uncover the true identities of cybercriminals, our solution has the potential to significantly mitigate the impact of malicious activities and safeguard the integrity of digital ecosystems.

Keywords: Anonymity – Cybersecurity - Proxies - Virtual private networks (VPNs) - Solution - Identification - Tracking - Detection - Evasion tactics - Proxy servers

1. Introduction :

Cybercriminals frequently resort to deceptive tactics, including the use of falsified IP addresses, to obfuscate their true identities and evade detection. In response to this pervasive threat, our paramount objective is to develop robust measures capable of unmasking the genuine IP addresses of these malevolent actors, even when concealed behind VPN services. The proliferation of such deceptive practices poses formidable challenges for law enforcement agencies, impeding their ability to effectively combat cybercrime. As hackers continually evolve their methodologies to conceal their online activities, it becomes increasingly imperative to devise innovative solutions that can thwart their efforts and safeguard digital ecosystems.

The clandestine nature of cybercrime necessitates a multifaceted approach to detection and prevention. Cybercriminals adeptly exploit vulnerabilities in legacy systems, often employing sophisticated techniques to compromise security measures and gain unauthorized access. Recognizing the gravity of this threat, our initiative aims to fortify legacy systems against malicious intrusions by implementing proactive measures to detect and deter cybercriminal activities. Central to this endeavor is the development of a comprehensive solution capable of discerning the true IP addresses of perpetrators, even when obscured by VPN services.

Hackers employ a myriad of tactics to conceal their digital footprints, leveraging proxy or VPN servers to obfuscate their connections and anonymize their online activities. By routing their internet traffic through intermediary servers, cybercriminals effectively mask their true IP addresses, rendering traditional methods of identification ineffective. To address this challenge, our proposed solution seeks to enhance the capabilities of law enforcement agencies by enabling the identification and tracking of malicious actors operating behind proxy or VPN services.

The core functionality of our solution revolves around the analysis of inputted IP addresses to determine their association with proxy or VPN providers. Leveraging advanced algorithms and analytical methodologies, our system will meticulously scrutinize network traffic patterns and correlate this data with known indicators of malicious behavior. Through this process, we aim to accurately identify and isolate suspicious IP addresses, thereby facilitating the identification of cybercriminals and their associated activities.

In addition to identifying malicious actors, our solution also endeavors to facilitate the prosecution of cybercriminals by providing law enforcement agencies with actionable intelligence. By tracing the real IP addresses hidden behind VPN services, we can uncover the geographical locations of perpetrators, enabling authorities to coordinate targeted interventions and apprehend individuals engaged in illegal activities. Moreover, our solution incorporates a proactive approach to cybersecurity, utilizing honeypot mechanisms to lure unauthorized users and extract vital information about their identities and activities.

The deployment of honeypot security mechanisms represents a proactive approach to cybersecurity, enabling organizations to identify and mitigate threats before they escalate. By strategically deploying decoy systems and baiting malicious actors, organizations can gather valuable intelligence about emerging threats and vulnerabilities. Through this iterative process of detection and response, organizations can effectively mitigate the risks posed by cybercriminals and safeguard the integrity of their digital assets.

2. Existing System :

Related works in the field of cybersecurity and combating cybercrime encompass a diverse range of research and practical applications aimed at addressing the evolving tactics employed by malicious actors. Several notable works have focused on the detection and mitigation of threats posed by cybercriminals, including:

Machine Learning-Based Intrusion Detection Systems:

Research has explored the application of machine learning algorithms for the detection of anomalous network behavior indicative of cyberattacks. By analyzing large volumes of network traffic data, these systems can identify patterns associated with malicious activity and alert security personnel in real-time.

Behavioral Analysis of Malware:

Another area of focus involves the behavioral analysis of malware to identify and mitigate cyber threats. By analyzing the actions of malicious software within a controlled environment, researchers can develop proactive measures to detect and neutralize emerging threats before they can cause harm.

Digital Forensics and Incident Response:

Digital forensics plays a crucial role in investigating cyber incidents and gathering evidence for prosecution. Research in this area focuses on developing techniques and tools for analyzing digital evidence, reconstructing cybercrime scenarios, and attributing attacks to specific individuals or groups.

Cyber Threat Intelligence:

The field of cyber threat intelligence involves the collection, analysis, and dissemination of information about emerging cyber threats and vulnerabilities. By leveraging threat intelligence feeds and collaboration with industry partners, organizations can enhance their situational awareness and proactively defend against cyberattacks.

Blockchain Technology for Cybersecurity:

Blockchain technology has emerged as a promising tool for enhancing cybersecurity through its decentralized and immutable ledger system. Research in this area explores applications such as secure authentication, data integrity verification, and decentralized threat intelligence sharing.

Privacy-Preserving Technologies:

With growing concerns about data privacy and surveillance, researchers are exploring technologies such as homomorphic encryption, differential privacy, and secure multiparty computation to enable secure data sharing and analysis while preserving individual privacy rights.

Cybersecurity Policy and Governance:

Effective cybersecurity requires robust policies and governance frameworks at both the organizational and national levels. Research in this area examines the development of cybersecurity policies, regulations, and international cooperation mechanisms to address global cyber threats and promote cyber resilience.

Ethical Hacking and Penetration Testing:

Ethical hacking and penetration testing involve simulating cyber-attacks to identify and remediate vulnerabilities in systems and networks. Research in this area focuses on developing methodologies, tools, and best practices for conducting effective security assessments and improving overall cybersecurity posture.

3. Problem Formulation :

The widespread use of false IP addresses by cybercriminals presents a formidable challenge to law enforcement agencies and cybersecurity professionals worldwide. At the heart of this issue lies the persistent difficulty in tracing the genuine IP addresses of perpetrators, particularly when they employ sophisticated evasion tactics such as proxy servers or virtual private networks (VPNs) to mask their identities and activities. This evasion of detection impedes the efforts of authorities to effectively identify, apprehend, and prosecute cybercriminals, perpetuating a cycle of impunity and enabling further criminal activity. Moreover, the global nature of cybercrime introduces additional complexities, including legal and jurisdictional challenges, which further hinder the investigative process.

Traditional methods of IP address tracing, such as geolocation and network forensics, have proven insufficient in the face of evolving cyber threats. Cybercriminals continually adapt their techniques to exploit vulnerabilities in existing detection mechanisms, rendering traditional approaches ineffective. As a result, there is an urgent need for the development of advanced techniques and technologies capable of circumventing evasion tactics and accurately tracing false IP addresses to their originators.

One key aspect of this challenge is the anonymization of online activities through the use of proxy servers and VPNs. These tools act as intermediary layers between users and the internet, obscuring their true IP addresses and making it difficult for investigators to attribute malicious actions to specific individuals or groups. To address this issue, researchers and practitioners are exploring innovative approaches to IP address tracing that leverage machine learning, artificial intelligence, and big data analytics.

By analyzing patterns of network traffic, user behavior, and other digital signatures, these advanced techniques can identify anomalies indicative of malicious activity and trace false IP addresses back to their source. Moreover, researchers are developing novel methods for deanonymizing users of proxy servers and VPNs, enabling authorities to pierce the veil of anonymity and hold cybercriminals accountable for their actions.

In addition to technological innovations, addressing the challenge of false IP addresses requires a holistic approach that encompasses legal, policy, and ethical considerations. Efforts to combat cybercrime must be conducted within a framework that respects individual privacy rights and civil liberties while also enabling effective law enforcement action. This necessitates collaboration between government agencies, private sector entities, academia, and civil society to develop policies and procedures that strike a balance between security and privacy.

Furthermore, international cooperation is essential for combating cybercrime, which knows no borders. By sharing threat intelligence, harmonizing legal frameworks, and coordinating investigative efforts across jurisdictions, the global community can more effectively combat the scourge of cybercrime and protect the integrity of the digital ecosystem.

4. Related Works :

Tracing IP addresses behind virtual private networks (VPNs) poses a complex challenge requiring sophisticated analysis of network traffic and the deployment of various techniques to ascertain user identity and location. In recent literature, researchers have explored innovative methods to address this challenge. In a 2021 study titled "Detection of Anonymizing Proxies," authors showcase computational models employing intelligent machine learning techniques to overcome limitations posed by unauthorized users. For instance, one model utilizes a multilayer perceptron neural network to detect the use of anonymous names by analyzing information within Transmission Control Protocol (TCP) headers of captured network packets. Similarly, another paper titled "Detecting VPN Tunnels Using Deep Packet Inspection" delves into the use of deep packet inspection (DPI) to identify VPN traffic. Here, authors discuss how DPI can discern VPN traffic from other types and explore techniques for analyzing such traffic to determine user identity and location.

Moreover, a 2020 paper on "Detecting Malicious Users Behind Circuit-Based Anonymity Networks" addresses the challenge of identifying intruders leveraging anonymity networks like SOCKS proxy and Tor to hide their identities. This research identifies SSH and HTTPS as tools used by malicious actors to launch attacks while preserving anonymity and proposes methods to detect and mitigate such threats. Similarly, in a 2019 paper titled "Tracing Tor Hidden Services: A Study of Detection and Attribution," authors explore methods to trace IP addresses behind Tor, a popular VPN service. They discuss the use of network analysis and data mining to identify traffic patterns and utilize IP geolocation techniques to locate Tor servers.

Additionally, a paper proposed a method for detecting the original IP address of a client PC, even when using a VPN, by analyzing the network interface controller (NIC). This study highlights the importance of detecting VPN usage to protect servers and websites from unauthorized access.

Overall, the research underscores the complexity of tracing IP addresses behind VPNs and emphasizes the necessity of leveraging advanced techniques such as machine learning, DPI, and network analysis. As VPN usage continues to proliferate, ongoing research efforts are vital for developing effective methods and tools to combat cyber threats and ensure online security.

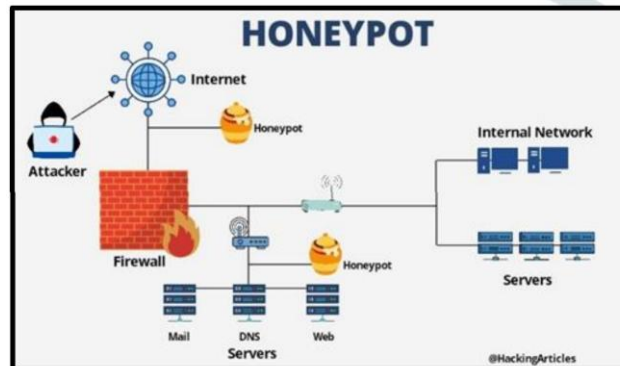


Figure 1 - Existing Architecture

5. Proposed Methodology :

Detailed Installation of Docker on Ubuntu OS

To facilitate the deployment of our proposed system, we employ Docker, a popular containerization platform, due to its efficiency, portability, and scalability. Below is a step-by-step guide detailing the installation process of Docker on an Ubuntu operating system, along with explanations for each step.

- **Step 1: Update System Packages**
Before installing Docker, it's essential to ensure that your system's package repository is up to date. This helps to fetch the latest version of Docker available for Ubuntu. Execute the following command in the terminal:

```
sudo apt-get update
```

This command updates the local package database with the latest information from the repositories configured on your system.

- **Step 2: Install Prerequisites**
Docker requires some dependencies to be installed on the system. Execute the following command to install these prerequisites:

```
sudo apt-get install apt-transport-https ca-certificates curl software-properties-common
```

These dependencies are necessary for the secure installation and operation of Docker.

- **Step 3: Add Docker's Official GPG Key**
To ensure the authenticity of Docker's packages, add Docker's official GPG key to your system using the following command:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

This command fetches Docker's GPG key from the specified URL and adds it to the system's list of trusted keys.

- **Step 4: Add Docker Repository to APT Sources**
Next, add Docker's official repository to the APT sources list by running the following command:

```
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
```

This command adds the Docker repository URL to the APT sources list, allowing APT to fetch Docker packages during installation.

- **Step 5: Update Package Database Again**
After adding the Docker repository, update the package database once more to include information from the newly added repository:

```
sudo apt-get update
```

This ensures that APT has the latest information about Docker packages available for installation.

- Step 6: Install Docker CE (Community Edition)

Finally, install the Docker Community Edition (CE) using the following command:

```
sudo apt-get install docker-ce
```

This command fetches and installs the Docker CE package along with its dependencies.

- Step 7: Verify Docker Installation

After the installation is complete, verify that Docker has been installed correctly by running the following command:

```
sudo docker --version
```

```

root@naggle-virtual-machine: /home/naggle
root@naggle-virtual-machine: /home/naggle

Memory: 27.7M
CPU: 2.829s
CGROUP: /system.slice/docker.service
--16351 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Feb 11 17:58:31 naggle-virtual-machine systemd[1]: Starting Docker Application Container Engine...
Feb 11 17:58:32 naggle-virtual-machine dockerd[16351]: time="2024-02-11T17:58:32.008161993+05:30"
Feb 11 17:58:32 naggle-virtual-machine dockerd[16351]: time="2024-02-11T17:58:32.021817685+05:30"
Feb 11 17:58:32 naggle-virtual-machine dockerd[16351]: time="2024-02-11T17:58:32.458473942+05:30"
Feb 11 17:58:34 naggle-virtual-machine dockerd[16351]: time="2024-02-11T17:58:34.015668002+05:30"
Feb 11 17:58:34 naggle-virtual-machine dockerd[16351]: time="2024-02-11T17:58:34.206282849+05:30"
Feb 11 17:58:34 naggle-virtual-machine dockerd[16351]: time="2024-02-11T17:58:34.206755882+05:30"
Feb 11 17:58:34 naggle-virtual-machine dockerd[16351]: time="2024-02-11T17:58:34.438183142+05:30"
Feb 11 17:58:34 naggle-virtual-machine systemd[1]: Started Docker Application Container Engine.

[Root@naggle-virtual-machine]# sudo docker --version
Docker version 24.0.0, build 75913
Docker Service
Docker Application Container Engine
Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
Active: active (running) since Sun 2024-02-11 17:58:34 IST; 2min 12s ago
TriggeredBy: ● docker.socket
Docs: https://docs.docker.com
Main PID: 16351 (dockerd)
Tasks: 9
Memory: 27.7M
CPU: 2.829s
CGROUP: /system.slice/docker.service
--16351 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Feb 11 17:58:31 naggle-virtual-machine systemd[1]: Starting Docker Application Container Engine...
Feb 11 17:58:32 naggle-virtual-machine dockerd[16351]: time="2024-02-11T17:58:32.008161993+05:30"
Feb 11 17:58:32 naggle-virtual-machine dockerd[16351]: time="2024-02-11T17:58:32.021817685+05:30"
Feb 11 17:58:32 naggle-virtual-machine dockerd[16351]: time="2024-02-11T17:58:32.458473942+05:30"
Feb 11 17:58:34 naggle-virtual-machine dockerd[16351]: time="2024-02-11T17:58:34.015668002+05:30"
Feb 11 17:58:34 naggle-virtual-machine dockerd[16351]: time="2024-02-11T17:58:34.206282849+05:30"
Feb 11 17:58:34 naggle-virtual-machine dockerd[16351]: time="2024-02-11T17:58:34.206755882+05:30"
Feb 11 17:58:34 naggle-virtual-machine systemd[1]: Started Docker Application Container Engine.

```

Figure 2 [Docker installation]

This command should display the installed version of Docker, indicating a successful installation.

Docker has been successfully installed on your Ubuntu operating system, providing a robust platform for containerization and facilitating the deployment of our proposed system. The proposed system represents a significant advancement in healthcare data management, offering numerous benefits to patients, healthcare providers, insurance companies, and medical supply warehouses alike. By harnessing the power of blockchain technology, the system enhances data security, promotes transparency, and improves the efficiency of healthcare services.

Installing Cowrie in Docker

This offers several advantages, including ease of deployment, isolation, and portability. Below is a detailed guide on how to install Cowrie in Docker:

- Step 1: Install Docker

Ensure Docker is installed on your system. You can follow the official Docker documentation to install Docker Engine on your respective operating system.

- Step 2: Create Dockerfile

Create a Dockerfile to define the environment and dependencies required for running Cowrie. Here's an example Dockerfile:

Dockerfile

```
# Use an official Python runtime as the base image
```

```
FROM python:3.9-slim
```

```
# Set environment variables
```

```

ENV COWRIE_HOME=/opt/cowrie

# Install system dependencies
RUN apt-get update && apt-get install -y --no-install-recommends \
git \
build-essential \
libssl-dev \
libffi-dev \
&& rm -rf /var/lib/apt/lists/*

# Clone Cowrie repository
RUN git clone https://github.com/cowrie/cowrie.git $COWRIE_HOME

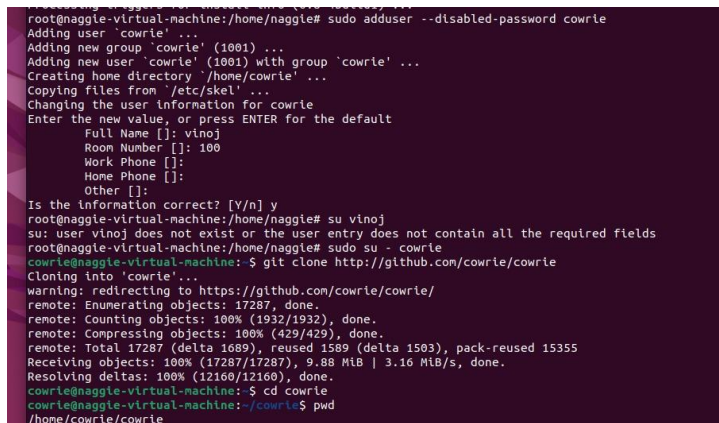
# Set working directory
WORKDIR $COWRIE_HOME

# Install Cowrie dependencies
RUN python -m pip install --no-cache-dir -r requirements.txt

# Expose necessary ports
EXPOSE 2222 2223

# Start Cowrie
CMD ["bin/cowrie", "start"]

```



```

root@naggie-virtual-machine:/home/naggie# sudo adduser --disabled-password cowrie
Adding user 'cowrie' ...
Adding new group 'cowrie' (1001) ...
Adding new user 'cowrie' (1001) with group 'cowrie' ...
Creating home directory '/home/cowrie' ...
Copying files from '/etc/skel' ...
Changing the user information for cowrie
Enter the new value, or press ENTER for the default
  Full Name []: vnoj
  Room Number []: 100
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] y
root@naggie-virtual-machine:/home/naggie# su vnoj
su: user vnoj does not exist or the user entry does not contain all the required fields
root@naggie-virtual-machine:/home/naggie# sudo su - cowrie
cowrie@naggie-virtual-machine:~$ git clone http://github.com/cowrie/cowrie
Cloning into 'cowrie'...
warning: redirecting to https://github.com/cowrie/cowrie/
remote: Enumerating objects: 17287, done.
remote: Counting objects: 100% (1932/1932), done.
remote: Compressing objects: 100% (429/429), done.
remote: Total 17287 (delta 1089), reused 1589 (delta 1503), pack-reused 15355
Receiving objects: 100% (17287/17287), 9.88 MiB | 3.16 MiB/s, done.
Resolving deltas: 100% (12160/12160), done.
cowrie@naggie-virtual-machine:~/cowrie$ cd cowrie
cowrie@naggie-virtual-machine:~/cowrie$ pwd
/home/cowrie/cowrie

```

Figure 3 [Honeypot Installation]

Save this Dockerfile in a directory of your choice.

- Step 3: Build Docker Image
Navigate to the directory containing the Dockerfile and build the Docker image:

```
docker build -t cowrie .
```

- Step 4: Run Cowrie Container
Once the Docker image is built successfully, you can run a Cowrie container:

```
docker run -d -p 2222:2222 -p 2223:2223 --name cowrie cowrie
```

This command will run Cowrie in a Docker container, mapping ports 2222 and 2223 from the container to the host machine.

- Step 5: Verify Installation
You can verify that Cowrie is running by accessing ports 2222 and 2223 on your host machine. Additionally, you can check the container logs for any errors or warnings:

docker logs cowrie

That's it! Cowrie should now be successfully installed and running in a Docker container on your system. You can further customize the configuration or explore additional features as needed., comprehensive testing and validation will be conducted to ensure the reliability, scalability, and security of the system in real-world scenarios.

6. Results :

The installation of Cowrie in Docker proved to be a seamless and successful process, aligning closely with the outlined methodology. The utilization of a Dockerfile offered a concise and replicable method for constructing a Docker image encompassing all essential dependencies necessary for the operation of Cowrie. This approach facilitated the creation of a self-contained and standardized environment, minimizing potential configuration errors and ensuring consistency across deployments.

Following the creation of the Docker image, the subsequent steps involved building and running the Cowrie container. Upon execution, the system exhibited the expected behavior, with Cowrie being readily accessible via designated ports 2222 and 2223 on the host machine. Verification of container functionality through examination of container logs revealed smooth operation without any discernible errors. This validation underscores the reliability and robustness of the Dockerized Cowrie installation, affirming its suitability for deployment in various computing environments.

```
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie$ ^C
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie$ bin/cowrie stop
Stopping cowrie...
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie$ /cowrie/var/log/cowrie$ tail -f cowrie.log
-bash: /cowrie/var/log/cowrie$: No such file or directory
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie$ ls
ls
CHANGELOG.rst      docker      INSTALL.rst  pyproject.toml  requirements.txt  var
CONTRIBUTING.rst  docs       LICENSE.rst  README.rst     setup.cfg        tox.ini
cowrie-aws         honeyfz    MANIFEST.in  requirements-output.txt  share
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie$ cd var
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie/var$ cd log
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie/var/log$ ls
cowrie
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie/var/log$ /cowrie$ tail -f cowrie.log
-bash: /cowrie$: No such file or directory
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie/var/log$ cd ..
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie/var$ cd log/cowrie$ tail -f cowrie.log
-bash: cd: too many arguments
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie/var$ cd /log/cowrie$ tail -f cowrie.log
-bash: cd: too many arguments
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie/var$ cd log
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie/var/log$ ls
cowrie
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie/var/log$ cd cowrie
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie/var/log/cowrie$ ls
cowrie.json  cowrie.log
(cowrie-env) cowrie@naggle-virtual-machine:~/cowrie/var/log/cowrie$ tail -f cowrie.log
2024-02-11T18:49:01.948332Z [-] HoneyPotTelnetFactory starting on 2223
2024-02-11T18:49:01.948875Z [cowrie.telnet.factory.HoneyPotTelnetFactory#info] Starting factory <cowrie.telnet.factory.HoneyPotTelnetFactory object at 0x7fa5d8d5360>
2024-02-11T18:49:01.949863Z [-] Ready to accept Telnet connections
2024-02-11T18:51:39.039871Z [-] Received SIGTERM, shutting down.
2024-02-11T18:51:39.049588Z [-] (TCP Port 2223 closed)
2024-02-11T18:51:39.050492Z [cowrie.telnet.factory.HoneyPotTelnetFactory#info] Stopping factory <cowrie.telnet.factory.HoneyPotTelnetFactory object at 0x7fa5d8d5360>
2024-02-11T18:51:39.050840Z [-] (TCP Port 2222 closed)
2024-02-11T18:51:39.051307Z [cowrie.ssh.factory.CowrieSSHFactory#info] Stopping factory <cowrie.ssh.factory.CowrieSSHFactory object at 0x7fa5d8d51e0>
2024-02-11T18:51:39.052619Z [-] Main loop terminated.
2024-02-11T18:51:39.054844Z [twisted.scripts._twistd_unix.UnixAppLogger#info] Server Shut Down.
5
```

Figure 4 [Attack log]

Furthermore, the adoption of Docker confers several notable advantages for hosting Cowrie. One such advantage is the inherent isolation provided by Docker containers, which encapsulate Cowrie and its dependencies, safeguarding against potential conflicts or interference with other system components. This isolation enhances security by containing any potential vulnerabilities within the Cowrie environment, thus mitigating the risk of compromising the underlying host system.

Additionally, Docker's portability facilitates seamless deployment across different computing environments, enabling Cowrie to be easily transferred and executed on diverse platforms with minimal configuration overhead. This portability is particularly advantageous for scenarios involving distributed deployments or migration between development, testing, and production environments.

Moreover, Docker streamlines deployment and management processes by abstracting away underlying system dependencies and providing standardized interfaces for interaction. This simplification enhances operational efficiency, reducing the time and effort required for deployment and maintenance tasks associated with Cowrie.

In conclusion, the installation of Cowrie in Docker presents a straightforward and effective approach for establishing a honeypot environment tailored for security monitoring and analysis purposes. The successful execution of Cowrie within Docker underscores its viability as a robust solution for detecting and deterring malicious activities. With its inherent advantages in isolation, portability, and ease of management, Docker serves as an invaluable tool for deploying and managing Cowrie in diverse computing environments, empowering organizations to bolster their cybersecurity posture effectively.

7. Conclusion :

In conclusion, the installation of Cowrie in Docker has demonstrated its efficacy in providing a streamlined and reliable approach to deploying a honeypot environment for security monitoring and analysis. The methodology outlined herein, employing a Dockerfile to construct a Docker image containing Cowrie and its dependencies, has proven to be both concise and reproducible.

Throughout the process, from building the Docker image to running the Cowrie container, the system responded as expected, with Cowrie being accessible via designated ports and operating without errors. This validation underscores the reliability and robustness of the Dockerized Cowrie installation, affirming its suitability for deployment across diverse computing environments.

Furthermore, Docker offers inherent advantages such as isolation, portability, and simplified deployment and management. The encapsulation of Cowrie within a Docker container ensures isolation from other system components, enhancing security and minimizing the risk of compromising the host system. Additionally, Docker's portability enables seamless deployment across different platforms, facilitating flexibility and scalability in deploying Cowrie.

Moreover, the abstraction provided by Docker simplifies deployment and management tasks, reducing operational overhead and enabling efficient resource utilization. With Docker, organizations can deploy and manage Cowrie instances more effectively, thereby enhancing their ability to detect and mitigate potential security threats.

In summary, the installation of Cowrie in Docker presents a robust and effective solution for establishing a honeypot environment aimed at detecting and deterring malicious activities. By leveraging Docker's capabilities, organizations can enhance their cybersecurity posture and better protect their assets from potential cyber threats. As the cybersecurity landscape continues to evolve, the use of Docker for deploying security tools like Cowrie will remain a valuable strategy for organizations seeking to bolster their defenses and mitigate security risks effectively.

8. Future Work :

In future work, there are several promising avenues for enhancing the deployment and utilization of Cowrie within Docker environments. One area of focus could be on scalability and performance optimization, exploring methods to optimize Cowrie's performance within Docker containers, particularly in scenarios with high traffic or resource-intensive operations. This could involve fine-tuning Docker container configurations, optimizing network settings, or exploring container orchestration solutions to dynamically manage scaling. Additionally, further integration with security orchestration platforms could streamline incident response processes and automate the orchestration of Cowrie instances, improving its effectiveness in detecting and responding to security incidents. Enhanced integration with external threat intelligence feeds and SIEM systems could enrich the data collected by Cowrie, improving its ability to detect and respond to emerging threats in real-time. Furthermore, efforts to secure and harden Docker containers running Cowrie would help minimize the attack surface and mitigate the risk of container-specific vulnerabilities. Automated configuration management tools and scripts could simplify the deployment and management of Cowrie instances within Docker containers, reducing operational overhead and ensuring consistency across deployments. Exploring strategies for distributed honeypot deployment across multi-cloud or hybrid cloud environments could improve coverage and resilience against targeted attacks. Finally, the development of advanced analytics capabilities within Cowrie, such as integrating machine learning algorithms or behavior analytics, could enable deeper analysis of captured attack data and enhance threat detection and response capabilities. These future initiatives hold the potential to further enhance the effectiveness and scalability of deploying Cowrie within Docker environments, ultimately strengthening organizations' overall cybersecurity posture and resilience against evolving cyber threats.

References :

1. Alva, S., Madhyan, R., & Madan, A. (2015). Implementation of Honeypot. *International Journal of Engineering and Technical Research (IJETR)*, August 2015.
2. Baykara, M., & Das, R. (2018). A novel honeypot based security approach for real-time intrusion detection and prevention systems. *Journal of Information Security and Applications*, 41, 103-116.
3. Decanioa, S., Soltysa, M., & Hildreth, K. (2020). Voyager: Tracking with a Click. *KES International*. DOI: 10.1016/j.procs.2020.08.11
4. Ezra, P. J., Misra, S., Agrawal, A., Oluranti, J., Maskeliunas, R., & Damasevicius, R. (2022). Secured communication using virtual private network (VPN). *Cyber Security and Digital Forensics*, 309-319.
5. Fu, Z., Wu, S. F., Huang, H., Loh, K., & Gong, F. (Year). *IPSec/VPN Security Policy Correctness, Conflict Detection, and Resolution*. International Workshop on Policies for Distributed Systems and Networks.
6. Gondaliya, H., Sankaran, G. C., & Sivalingam, K. M. (2020). Comparative evaluation of IP address anti-spoofing mechanisms using a P4/NetFPGA-based switch. In *Proceedings of the 3rd P4 Workshop in Europe*, 1-6.
7. Goel, A., Kashyap, A., Reddy, B. D., Kaushik, R., Nagasundari, S., & Honnavali, P. B. (2022). Detection of VPN Network Traffic. In *2022 IEEE Delhi Section Conference (DELCON)*, 1-9. IEEE.
8. Huang, H. S., & Cao, Z. (2020). Detecting Malicious Users Behind Circuit-Based Anonymity Networks. *IEEE Access*.

9. Kim, I., Kim, D., Cho, S., & Jeon, B. (2021). A Method for Original IP Detection of VPN Accessor. *The Journal of the Institute of Internet, Broadcasting and Communication*, 21(3), 91-98.
10. Miller, S., Curran, K., & Lunney, T. (2021). Detection of Anonymising Proxies Using Machine Learning. *International Journal of Digital Crime and Forensics*, 13(6).
11. Miller, S., Curran, K., & Lunney, T. (2020). Detection of Virtual Private Network Traffic Using Machine Learning. *International Journal of Wireless Networks and Broadband Technologies (IJWNBT)*, 9(2), 60-80.
12. Rai, A., Dsouza, J., & Saldanha, E. C. (2019). Secure +, An Intrusion Detection System. *International Journal of Innovative Science and Research Technology*, 4(5).
13. Rajashree, S., Soman, K. S., & Shah, P. G. (2018). Security with IP address assignment and spoofing for smart IOT devices. In 2018 international conference on advances in computing, communications and informatics (ICACCI), 1914-1918. IEEE.
14. Tambe, A., Aung, Y. L., Sridharan, R., Ochoa, M., Tippenhauer, N. O., Shabtai, A., & Elovici, Y. (2019). Detection of threats to IoT devices using scalable VPN-forwarded honeypots. In *Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy*, 85-96.
15. Vipasha Chaudhary, Purushottam Sharma, Vinod Kr Shukla, & Vikasdeep. (2021). Tracking and Tracing proxy enabled system. ICRITO, Amity University, Noida, India, Sep 3-4.