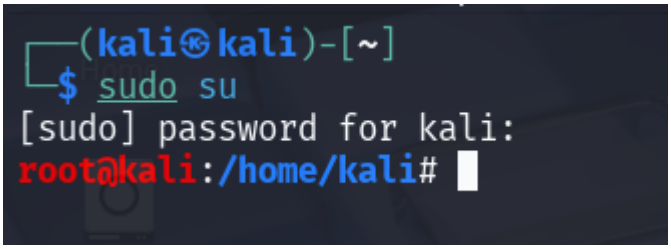
Honeypot Server Setup

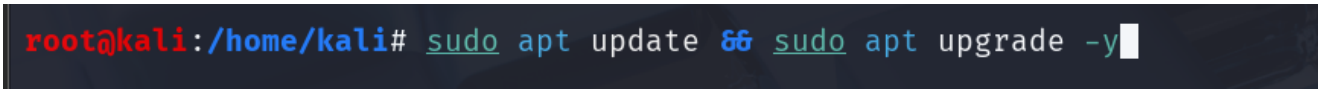
Introduction :
In Cyber Security, honeypots are more powerful tools that attract and capture malicious activity. Honeypots are 'fake' system designed to look like a real target for attackers.
The Purpose of the project was to deploy a honeypot server to simulate Vulnerable services and detect malicious attack patterns.
By using COWRIE, a medium-interaction SSH/TELNET honeypot, attacker activities such as brute-force login attempts and command execution were captured, analyzed, and visualized.

Prerequisites:
- kali Linux installed and updated
- kali purple
- Virtual Machine/Oracle VirtualBox

Implementation Steps:
Step 1: Setup environment.
Before we beginning need to change to the root user.

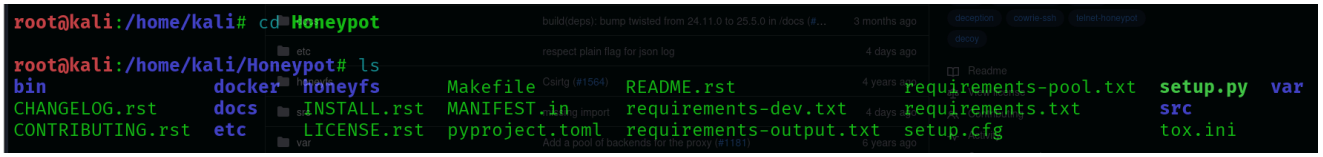


sudo su
Step 2: Updating the System
First we need to update the system



sudo apt update && sudo apt upgrade -y
Step 3: Installing Cowrie
Cowrie isn't in kali's default repository, so we will download it from Github.

git clone https://github.com/cowrie/cowrie.git /home/kali/Honeypot
List Content:



ls
Step 4: Setting up Python Virtual Environment.
Cowrie requires a python 3 virtual environment to run smoothly. Let's set that up
1. Installing the python virtual environment package:

```
root@kali:/home/kali# sudo apt install python3-venv -y
```

sudo apt install python3-venv -y

2. Creating a virtual environment within the Honeypot directory:

```
root@kali:/home/kali/Honeypot# python3 -m venv cowrie-env
```

python3 -m venv cowrie-env

3.Activating the Virtual Environment:

```
root@kali:/home/kali/Honeypot# source cowrie-env/bin/activate
```

source cowrie-env/bin/activate

Step 5: Installing dependencies

Installing all the Python packages that cowrie need to function.

We should see (cowrie-env) in our command prompt.

```
(cowrie-env)root@kali:/home/kali/Honeypot# pip install -r requirements.txt
Requirement already satisfied: attrs==25.3.0 in ./cowrie-env/lib/python3.13/site-packages (from -r requirements.txt (line 1)) (25.3.0)
Requirement already satisfied: bcrypt==4.3.0 in ./cowrie-env/lib/python3.13/site-packages (from -r requirements.txt (line 2)) (4.3.0)
Requirement already satisfied: cryptography==45.0.6 in ./cowrie-env/lib/python3.13/site-packages (from -r requirements.txt (line 3)) (45.0.6
)
```

pip install -r requirements.txt

Step 6: Configure Cowrie:

Once the dependencies are installed, we need to configure cowrie:

1.Copying the sample configuration file:

We need to copy the file because cowrie.cfg.dist to cowrie.cfg because cowrie needs this specific file(cowrie.cfg) to run. This copied file allow us to make any changes we want, such as adjusting settings, without affecting the original template file.

```
(cowrie-env)root@kali:/home/kali/Honeypot# cp etc/cowrie.cfg.dist etc/cowrie.cfg

(cowrie-env)root@kali:/home/kali/Honeypot# ls etc/
cowrie.cfg  cowrie.cfg.dist  userdb.example
```

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

ls etc/

We can open the configuration file and need to change in text in that file setting if needed.

```
(cowrie-env)root@kali:/home/kali/Honeypot# nano etc/cowrie.cfg
```

nano etc/cowrie.cfg

After opening the file We will need to change the `#hostname` and SSH Settings.

The key reason for this step is :

2.**Creating Illusions**:

Setting a realistic hostname, such as **ssh_server022**, makes the server appear genuine, encouraging attackers to engage more seriously, which provides valuable behavioral data.

3. **Enhancing Credibility**:

Modifying default SSH settings, like using a non-standard port instead of the typical port 22, helps disguise the honeypot. This increases the likelihood of attracting serious interactions rather than just automated attacks.

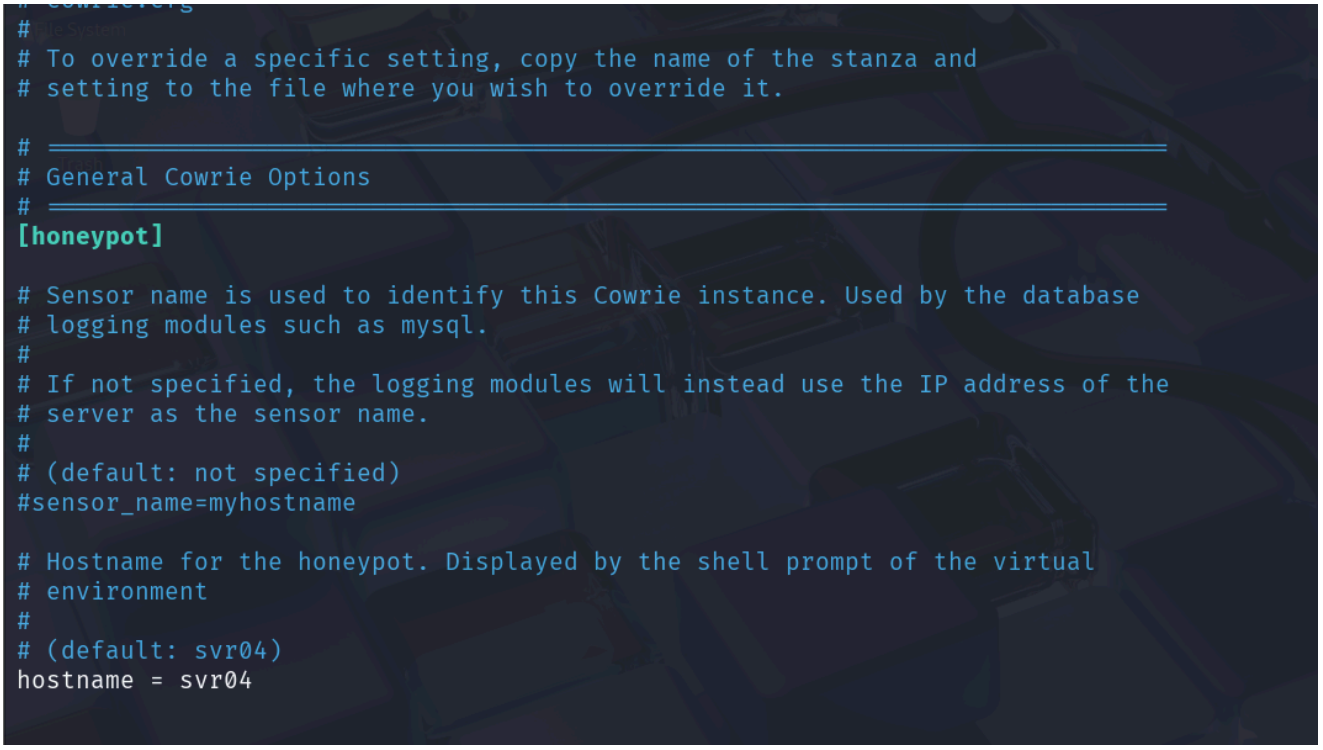
4. **Improving Log Analysis**:

Unique hostnames and altered SSH ports facilitate clearer log differentiation between automated and human attacks, aiding in the identification of targeted threats.

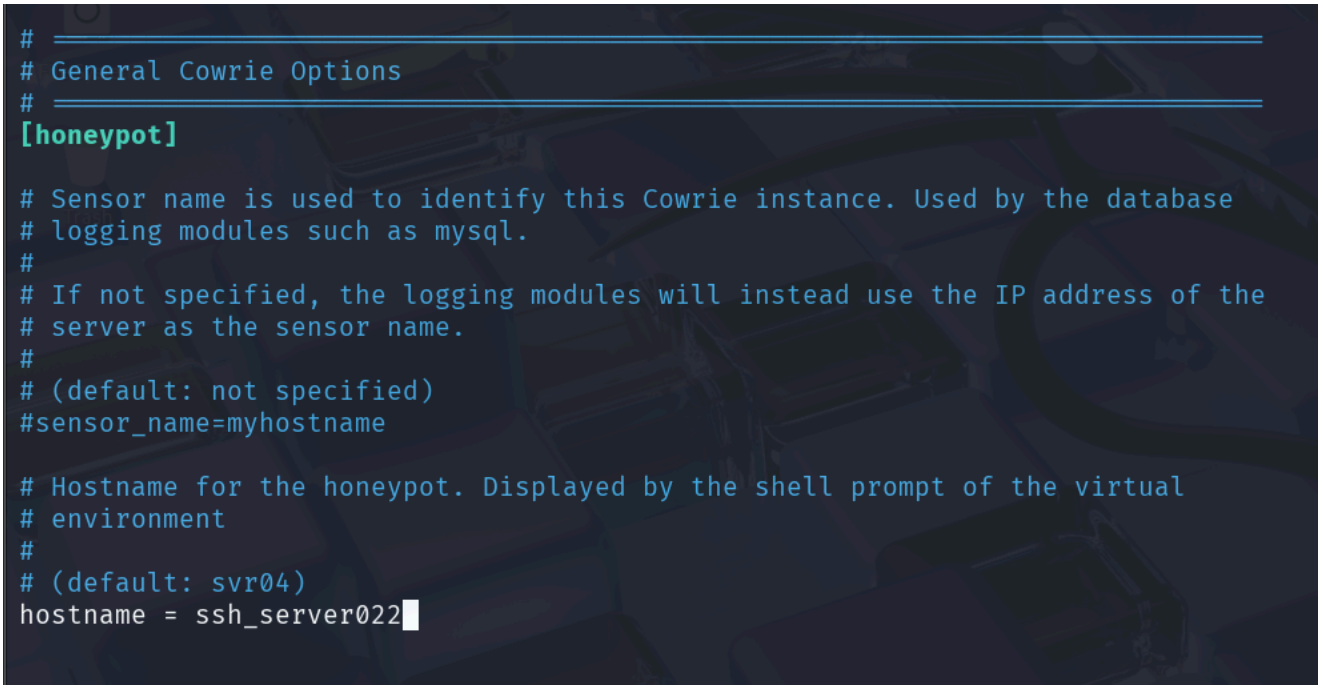
5. **Evasion of Detection**:

Changing standard configurations helps avoid detection by automated tools that identify honeypots, allowing attackers to spend more time interacting with the system and yielding more data for analysis.

Scroll down to change the host name

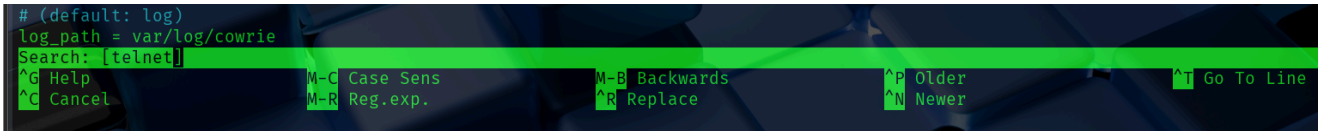


to



hostname = ssh_server022

CTRL + W Look for telnet



[telnet]

Change enabled =false -> change enabled = true

```
# Public key authentication
# This is an all or nothing switch that will allow none or any public key certificate to login
#
# (default: false)
auth_publickey_allow_any = false

# Configure keyboard-interactive login
auth_keyboard_interactive_enabled = false


# ================================================================
# Telnet Specific Options
# ================================================================
[telnet]

# Enable Telnet support, disabled by default
enabled = false
```

to

```
# Public key authentication
# This is an all or nothing switch that will allow none or any public key certificate to login
#
# (default: false)
auth_publickey_allow_any = false

# Configure keyboard-interactive login
auth_keyboard_interactive_enabled = false


# ================================================================
# Telnet Specific Options
# ================================================================
[telnet]

# Enable Telnet support, disabled by default
enabled = true
```

Save (CTRL+O) enter and Exit (CTRL+X)

Step 7: Setting up Cowrie to listen on port 22

A. **Deceptive Listening Port**:

Most legitimate SSH servers utilize port 22, leading attackers to target this port first. By configuring Cowrie to appear as if it is listening on port 22, we create the illusion of a genuine server.

B. **Traffic Redirection**:

A command is employed to redirect incoming traffic from port 22 to port 2222, where Cowrie is actively listening.

C. **Capturing Realistic Attacks**:

This redirection allows attackers who connect to port 22 to unknowingly interact with Cowrie on port 2222, enhancing the likelihood of capturing more authentic attack patterns.

```
(cowrie-env)root@kali:/home/kali/Honeypot# iptables -t nat -A PREROUTING -p tcp --dport 22 -j REDIRECT --to-port 2222
```

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

· **iptables:** This is the tool used to create rules for managing network traffic.

· **-t nat:** Specifies the NAT (Network Address Translation) table, used for traffic redirection.

· **-A PREROUTING:** Adds a rule to alter the packet before it reaches its destination.

· **-p tcp:** Indicates that this rule applies to TCP traffic (which SSH uses).

· — dport 22: The port we want to redirect (22, the default SSH port).

· **-j REDIRECT:** Tells the system to forward the traffic.

· **— to-port 2222:** Specifies the new port to which the traffic will be sent.

Optional for Telnet:

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

Step 8: Settings up permissions:

1.Changing Ownership

/var/run : Temporary, runtime data, usually PID files and other temporary information needed for the application's active session.

chmod -R kali: kali /home/kali/Honeypot/var/run

Changing Ownership

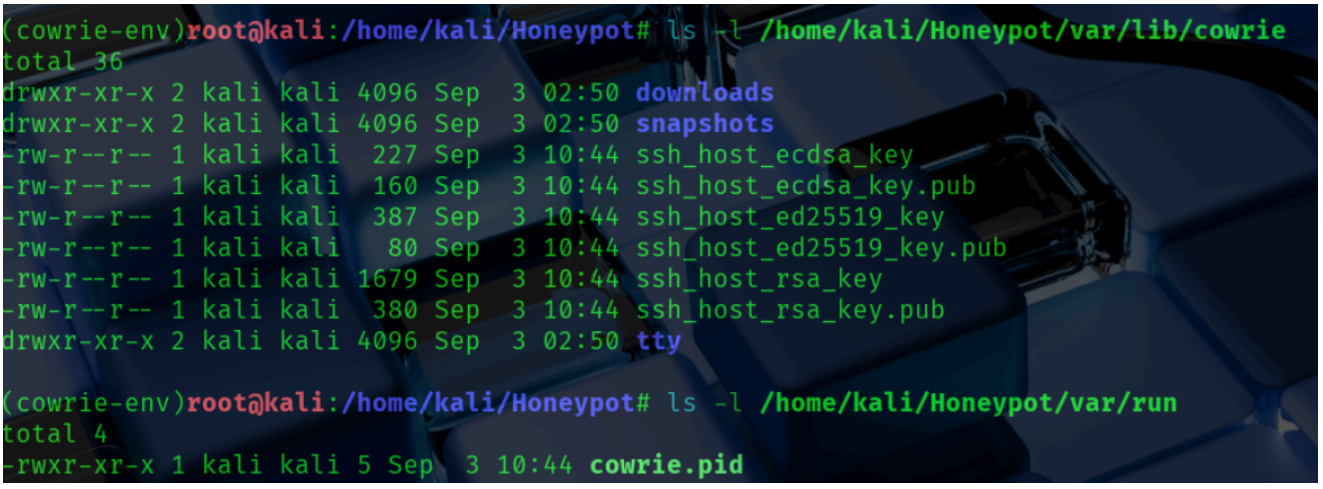
/var/lib/cowrie: Persistent data specific to Cowrie, which may include configuration files, logs, or data created by Cowrie.

chown -R kali: kali /home/kali/Honeypot/var/lib/cowrie

Setting Permissions

chmod -R 755 /home/kali/Honeypot/var/run

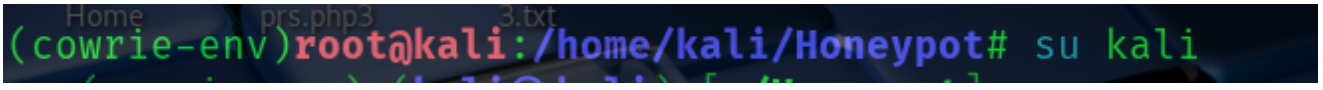
Verifying Ownership



ls -l /home/kali/Honeypot/var/lib/cowrie

ls -l /home/kali/Honeypot/var/run

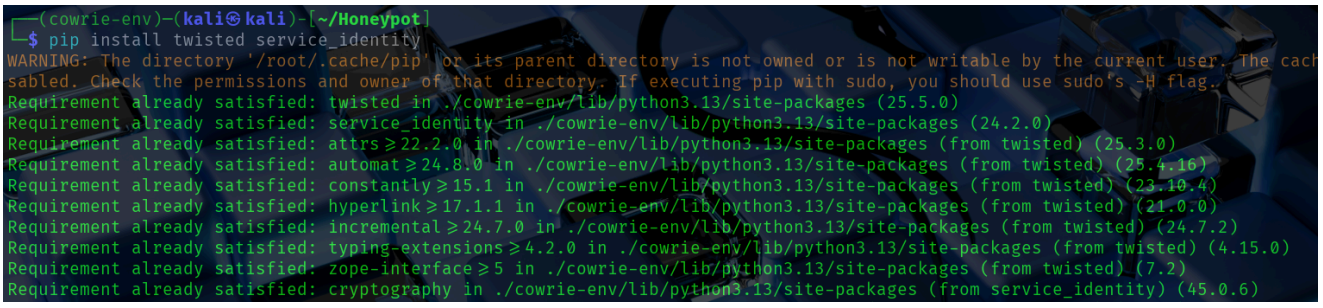
2. Changing to the non root user.



su kali

step 9:Starting the Honeypot:

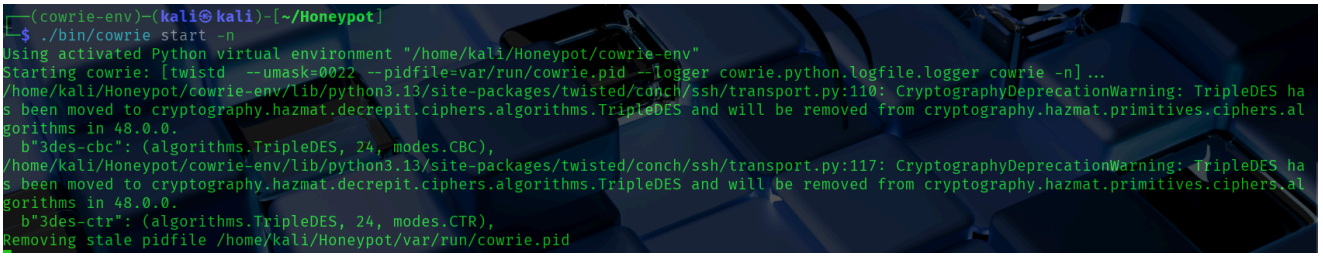
Before starting the Cowrie we need to install twisted service_identity because if there is an error while starting the Honeypot we need to install twistd.



pip install twisted service_identity

Starting the Honeypot:

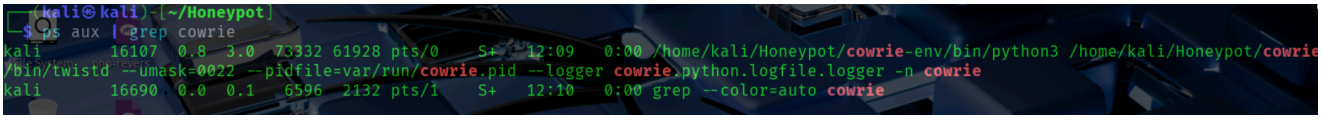
Running Cowrie to start capturing any attempted connections:



./bin/cowrie start -n

Cowrie has started without showing any errors regarding permissions or multiple instances. The deprecation warnings about TripleDES are not critical issues and can be ignored, as they are related to older encryption algorithms that will eventually be removed in future updates.

Check Active Processes:



ps aux | grep cowrie

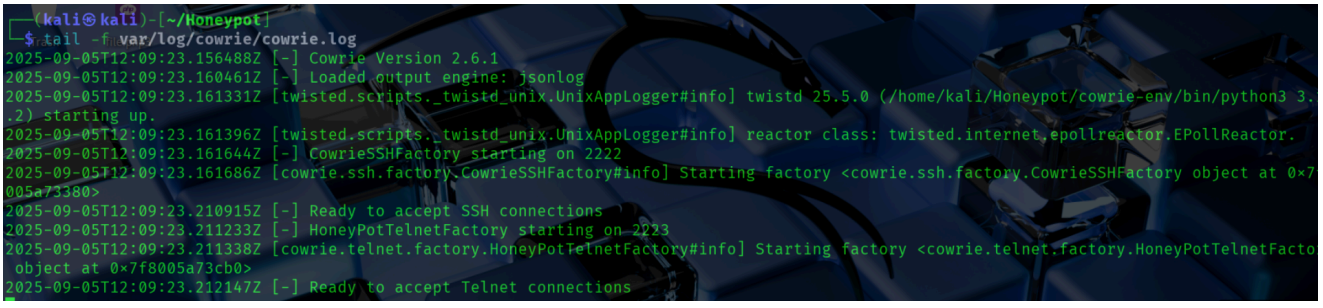
ps aux: This command lists all running processes on your system.

- **a** shows processes for all users, not just the current user.
- **u** displays the user who owns each process.
- **x** includes processes not attached to a terminal (background processes, like services).
- **| (Pipe)**: The pipe takes the output of ps aux and sends it to the next command (grep cowrie).
- **grep cowrie**: This part searches the output for any lines that contain the word “cowrie.” This filters out the list so that only processes with “cowrie” in their description are shown.

The Cowrie honeypot is running. The output shows a Cowrie process with the twistd command, which is used to start Cowrie, along with the relevant path and parameters.

Viewing the Logs

To confirm that Cowrie is capturing activity, we will check the logs for any login attempts or other interactions



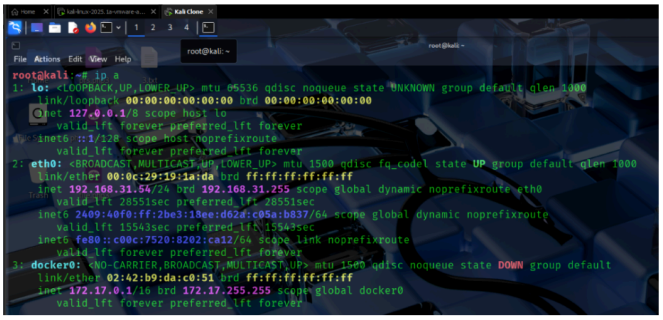
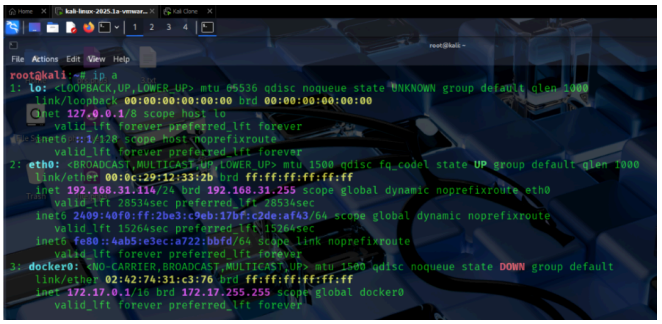
tail -f var/log/cowrie/cowrie.log

The log output:

- **CowrieSSHFactory starting on 2222**: This means Cowrie is ready to accept SSH connections on port 2222.
- **HoneyPotTelnetFactory starting on 2223**: Cowrie is also set up to accept Telnet connections on port 2223.

Step 10: Simulating an Attack :

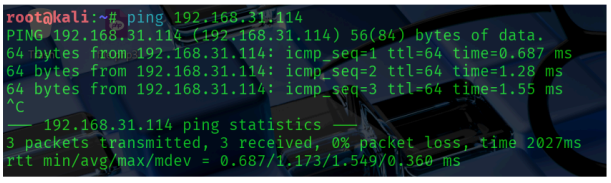
In the 2 machines checking for the IP address

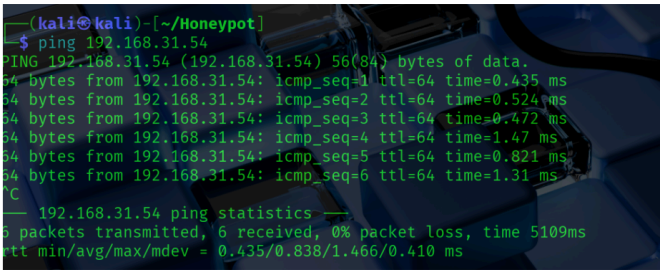



The first one(Left hand side) is Kali clone and

The second one(Right-hand side) is kali Original

Now we need to check concoction using ping:

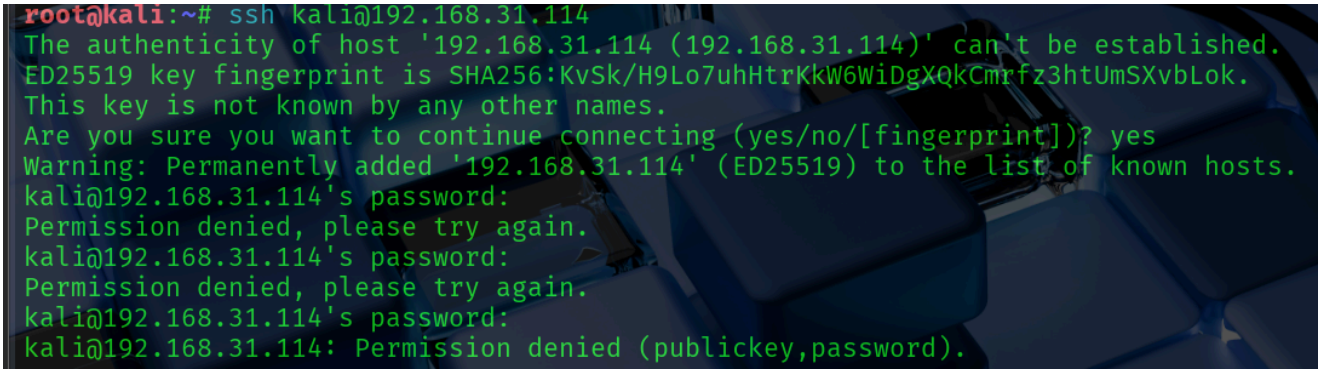




ip a

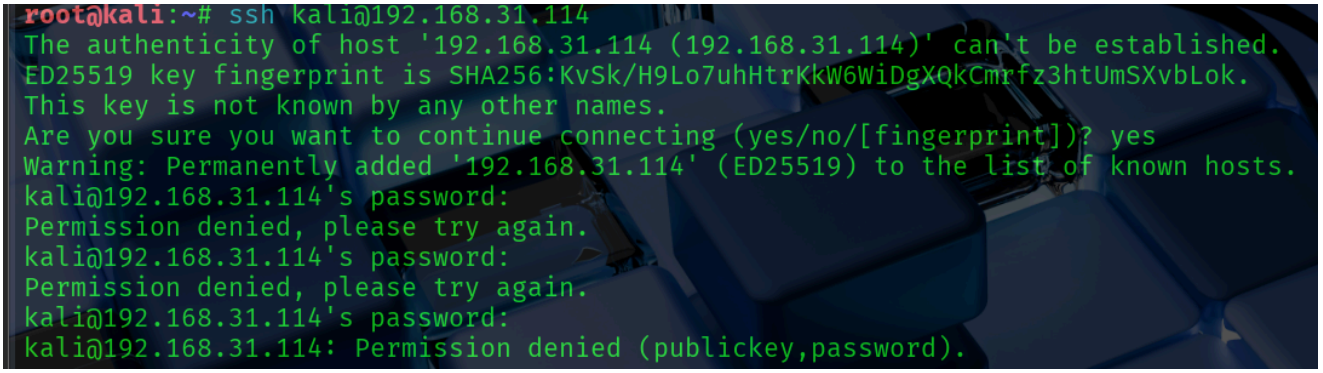
- Once we confirm that both machines can communicate, we can proceed to simulate the attack.
- To simulate attacks in this environment, we will set up a few common penetration testing techniques to see how the honeypot responds
- Open a terminal on the attacker machine (Kali Purple).
- Attempting to connect Cowrie honeypot using an SSH client.

This will allow to simulate an interaction with the target system and observe how it responds to login attempts.



1.Monitor the Logs on cowrie:

The log shows that someone (in this case, from the IP address 192.168.31.114) is trying to log in to the Cowrie honeypot using the username kali.



- The logs show that the person kept trying to enter the password but failed each time. This is useful information because it shows that someone is actively trying to access your server but does not have the correct password.

2.Attempting a Brute-Forcing SSH Logins Attack with Hydra

- **Hydra**:

- Hydra is fast and works well for trying to break into many types of services, like SSH, FTP, and others. It’s a good choice if you want a quick test or if you’re working on a smaller system with fewer resources.

hydra -l kali -P /usr/share/wordlists/rockyou.txt ssh://<Target_IP> -t 4

hydra -l kali -P /usr/share/wordlists/rockyou.txt ssh://192.168.31.114 -t 4

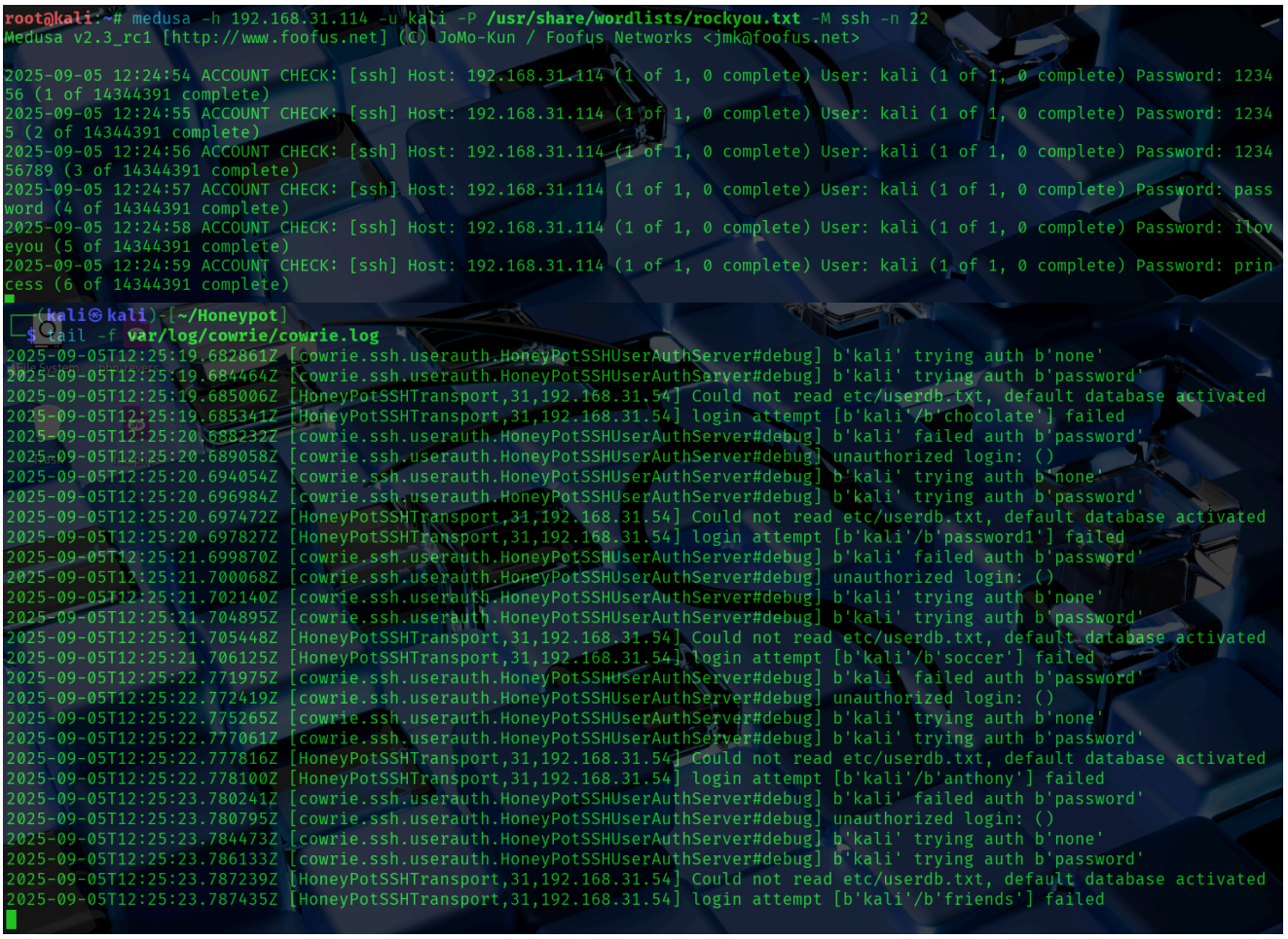
tail -f /home/kali/Honeypot/var/log/cowrie/cowrie.log

3.Attempting a Brute-Forcing Logins Attack with Medusa

- **Medusa**:

Medusa is known for being able to handle larger-scale attacks because it runs multiple

attempts at once, making it great for testing many usernames or passwords at the same time. It's best for cases where you have more data or want to simulate attacks on a bigger system.



medusa -h <Target_IP> -u kali -P /usr/share/wordlists/rockyou.txt -M ssh -n 22
medusa -h 192.168.31.114 -u kali -P /usr/share/wordlists/rockyou.txt -M ssh -n 22
tail -f /home/kali/Honeypot/var/log/cowrie/cowrie.log

-> **Tool Name**: **Medusa** is the tool being used for the brute-force attack.

-> **Target Specification**:

The option **-h <Target_IP>** is used to define the target IP address, which in this context is the IP of the Cowrie honeypot.

->**Username Selection**:

The **-u kali** option indicates the username for the login attempt. The username can be modified to test different accounts.

-> **Password List**:

The **-P /usr/share/wordlists/rockyou.txt** option specifies the path to the password list for the attack. The **rockyou.txt** wordlist is commonly used due to its extensive collection of frequently used passwords.

->**Module Selection**:

The **-M ssh** option indicates that the SSH module is being used for the brute-force login attempt.

->**Port Configuration**:

The **-n 22** option specifies the port number for the connection. While port 22 is the default for SSH, it should be noted that in the Cowrie setup, port **2222** is the actual listening port due to

the redirection configuration.

Step 11 : Stop the cowrie:

./bin/cowrie stop -n