A Report on

SIEM integrated with Honeypot

LabSetup and Project By

Aditya Raj DG

Table of contents

List of Figures

1.) Abstract

This report provides a detailed description of the configuration of the Honeypot in integration with the Security Information and Event Management (SIEM) tool. The Cowrie Honeypot, which includes Telnet and SSH connection services, was set up in Kali Linux, which is installed in VirtualBox where Kali Linux is in virtual Environment, A Windows 11 machine was used to set up the SIEM Tool Called Splunk, which run in a cloud platform , with an Ubuntu machine acting as a Windows Subsystem for Linux (WSL) was used to perform range of attacks, including Hydra attacks, Brute force, Wordlist, SSH and Telnet. The resulting attacked logs were then monitored in the SIEM tool Splunk, over a period of seven days from 29 July to 4 August. The logs were collected through the use of the HTTP Collector and were then analyzed using the SIEM tool.

The objective of this study was to investigate the honeypot concept and analyze the log events from the honeypot in the Security Information and Event Management (SIEM) tool Splunk, in order to gain an understanding of the potential risks associated with the usage of poor login information where it increase the chance of brute force attacks the failed passwords attempts were performed. The log analysis included SSH connection attempts, Hydra attacks and password cracking attempts etc.

The Key findings of this project are by Integrating a honeypot with a SIEM tool, It makes organisations more secure by showing them where the risk occurring. This integration helps to understand new and different attacks. Also gain a knowledge that how organisations respond to incidents by adding a context to security alerts and providing useful information like how attacks are carried out and it also helps to gather information on threats, identify common attack patterns and understand attacker behaviour. By integrating honeypots with SIEM, the SIEM tool provides a complete view of the threats, improves how anomalies are detected and helps to take a proactive approach to security,

2.) Introduction

A honeypot is a fake system which is used to detect and study attacks on real systems. This fake systems or networks attract the attackers by letting the security teams to watch and log the activities of the attacker. Honeypots also help us understand how attackers work. There are two types of honeypot such as low interaction and high interaction Honeypots. Low interaction honeypots are easier to manage while the high interaction honeypots offer a complete operating environment. Honeypots are useful for collecting information about threats, malware, different types of attacks etc. Honeypot also help us in training the security professionals. Honeypot help to detect threats early and take a necessary prevention measures for security purpose. By integrating honeypot with SIEM tool provides a complete view of the threat situation and helps to improve the security. Honeypot is different from other security types the honeypot do not block the attacks directly. The Honeypot purpose is to make the organization IDS(Intrusion Detection system) better and safe.

There are many different types of honeypot around the current popular types of honeypots are listed below:

2.1) Spam honeypot

This Type of honeypot identifies spammers before they reach your inbox. It often has open relays to get attacked and works with RBL(remote block list or Realtime blackhole list) lists in order to block the suspicious or malicious traffic.

2.2) Malware honeypot

This kind of honeypot is designed to simulate the weak systems, applications, and APIs in order to attract malware attacks. By the data gathered from the honeypot will be use for malware pattern recognition, which will help with the development of efficient malware detection tools.

2.3) Database honeypot

Web attackers often target databases, where we can gain knowledge and learn about various attack and tactics of Database such as SQL injection, privilege abuse, SQL services exploitation and much more by setting up a database honeypot.

2.4) Spider Honeypot

This Spider honeypot works by creating fake websites and malicious links that are only reachable by computer programs known as web crawlers. When a crawler gains access to the honeypot it is identified and its headers are saved for further examination, Spider honeypot helps in detecting and blocking the same malicious bots and ad-network crawlers.

Honeypots are a crucial element in the domain of cybersecurity, These Systems are specifically designed to prevent cyber attacks. The data collected from honeypots provides valuable information about emerging threats. In addition honeypots act as early warning systems, detecting potential attacks before they impact the systems. Furthermore, honeypots provide an environment in a way where security researchers can observe and analyse attacker behaviour, tactics, techniques and procedures. Honeypot has a significant importance for the enhancement of defensive measures and the training of security personnel. They provide the analysis of malware by the examination of malware samples and enable to build the effective detection and mitigation strategies also honeypots provide legal evidence in order to trial the cybercriminals.

3.) Methodology

In order to achieve the Assessment objectives, the cowrie as a honeypot and the SIEM tool Splunk. The cowrie honeypot was configured on the Kali machine, which is installed within the virtual environment (Virtual Box). The SIEM tool Splunk was set up on the Windows 11 , and the WSL Ubuntu machine was used to perform attacks on the cowrie honeypot ,each Lab setup configuration are shown below.

4.) Honeypot Configuration Setup

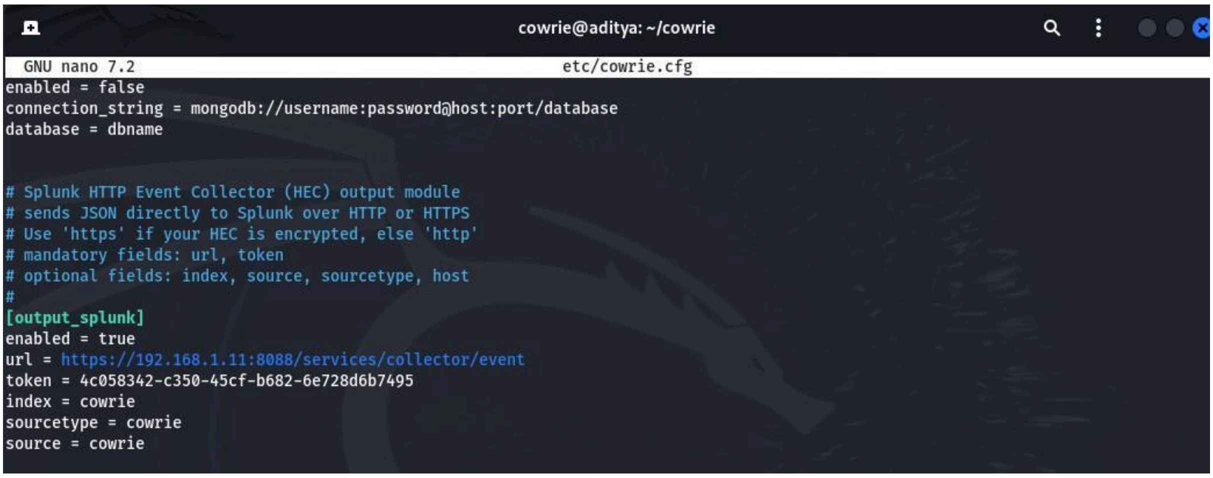
As Mentioned above we have installed Cowrie in the kali Linux Machine and the configurations goes as follows:

1. Installed Kali Linux Machine in Virtual Environment and Selected Bridge Adapter Network setting in order to install cowrie inside the kali, after installing the kali machine Static IP is specified as(192.168.1.12).

2. Installed Cowrie by creating cowrie environment,The below command specifies the Commands to install cowrie with all the dependencies and the pre requisites.

   - sudo apt-get install git python3-virtualenv libssl-dev libffi-dev build-essential libpython3-dev python3-minimal authbind virtualenv
   - sudo adduser --disabled-password cowrie
   - sudo su – cowrie
   - git clone http://github.com/cowrie/cowrie
   - python -m venv cowrie-env
   - source cowrie-env/bin/activate

3. The Cowrie honeypot was integrated with SIEM Splunk by changing the configuration file which is present in the cowrie, The cowrie was Successfully configured to accept the connections from 22,2222,23,2223 ports which enables the SSH and telnet connections. Once Cowrie is started the Honeypot is active and it is ready to accept the SSH and telnet Connections.



```
  GNU nano 7.2                                        etc/cowrie.cfg
enabled = false
connection_string = mongodb://username:password@host:port/database
database = dbname


# Splunk HTTP Event Collector (HEC) output module
# sends JSON directly to Splunk over HTTP or HTTPS
# Use 'https' if your HEC is encrypted, else 'http'
# mandatory fields: url, token
# optional fields: index, source, sourcetype, host
#
[output_splunk]
enabled = true
url = https://192.168.1.11:8088/services/collector/event
token = 4c058342-c350-45cf-b682-6e728d6b7495
index = cowrie
sourcetype = cowrie
source = cowrie
```

Fig 1) Configuration file of Cowrie(cowrie.cfg)

4. The Log are Generated in cowrie.log and json.log ,The Changes made in Configuration file includes, i)make enabled = "true", ii)replace the Splunk Server IP Address(192.168.1.11) in the Url Field, iii)Copy and Paste the HTTP Event Collector(HEC) token which is generated in SIEM Splunk Tool, By this Changes we can Forward the Logs generated in the cowrie to the SIEM Tool Splunk for Log Analysis.

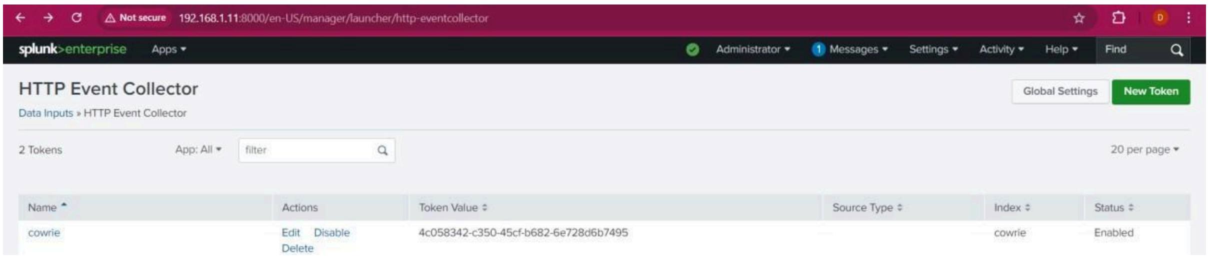
5.) SIEM Splunk Configuration

The SIEM Tool Splunk was Installed from the official website (https://www.splunk.com/en_us/download/splunk-cloud.html)of splunk, we installed splunk Enterprise edition, Once Splunk is installed in the Windows 11 Machine we followed the Below Steps in order to Integrated the Splunk SIEM with the Cowrie Honeypot.

1. Login to Splunk DashBoard with the Host IP(192.168.1.11:8000).

2. Once Logged in We did create a New Index and named it as "cowrie" this Setting is available by navigating to (Settings>Indexes>Add New Index>Save).

3. Once Index is Created We Generated the HTTP EVENT COLLECTOR token which is to be added in the cowrie configuration file (cowrie.cfg) this settings can be found by navigating to (Settings>Data Inputs>HTTP Event Collector>Add New Token>Save).



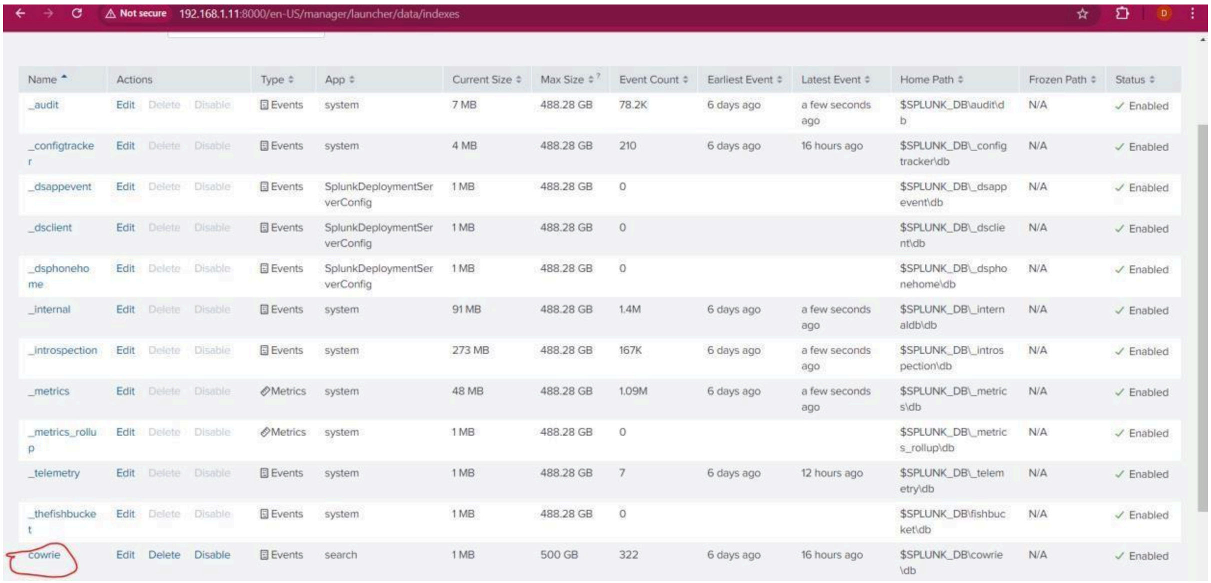
HTTP Event Collector

Data Inputs » HTTP Event Collector

2 Tokens App: All filter 20 per page

Name	Actions	Token Value	Source Type	Index	Status
cowrie	Edit Disable Delete	4c058342-c350-45cf-b682-6e728d6b7495		cowrie	Enabled

Fig 2) HTTP Event Collector where we get the Token Value



192.168.1.11:8000/en-US/manager/launcher/data/indexes

Name	Actions	Type	App	Current Size	Max Size	Event Count	Earliest Event	Latest Event	Home Path	Frozen Path	Status
_audit	Edit Delete Disable	Events	system	7 MB	488.28 GB	78.2K	6 days ago	a few seconds ago	$SPLUNK_DB\audit\db	N/A	✓ Enabled
_configtracker	Edit Delete Disable	Events	system	4 MB	488.28 GB	210	6 days ago	16 hours ago	$SPLUNK_DB_configtracker\db	N/A	✓ Enabled
_dsappevent	Edit Delete Disable	Events	SplunkDeploymentServerConfig	1 MB	488.28 GB	0			$SPLUNK_DB_dsappevent\db	N/A	✓ Enabled
_dsclient	Edit Delete Disable	Events	SplunkDeploymentServerConfig	1 MB	488.28 GB	0			$SPLUNK_DB_dsclient\db	N/A	✓ Enabled
_dsphonehome	Edit Delete Disable	Events	SplunkDeploymentServerConfig	1 MB	488.28 GB	0			$SPLUNK_DB_dsphonehome\db	N/A	✓ Enabled
_internal	Edit Delete Disable	Events	system	91 MB	488.28 GB	1.4M	6 days ago	a few seconds ago	$SPLUNK_DB_internaldb\db	N/A	✓ Enabled
_introspection	Edit Delete Disable	Events	system	273 MB	488.28 GB	167K	6 days ago	a few seconds ago	$SPLUNK_DB_introspection\db	N/A	✓ Enabled
_metrics	Edit Delete Disable	Metrics	system	48 MB	488.28 GB	1.09M	6 days ago	a few seconds ago	$SPLUNK_DB_metrics\db	N/A	✓ Enabled
_metrics_rollup	Edit Delete Disable	Metrics	system	1 MB	488.28 GB	0			$SPLUNK_DB_metrics_rollup\db	N/A	✓ Enabled
_telemetry	Edit Delete Disable	Events	system	1 MB	488.28 GB	7	6 days ago	12 hours ago	$SPLUNK_DB_telemetry\db	N/A	✓ Enabled
_thefishbucket	Edit Delete Disable	Events	system	1 MB	488.28 GB	0			$SPLUNK_DB\fishbucket\db	N/A	✓ Enabled
cowrie	Edit Delete Disable	Events	search	1 MB	500 GB	322	6 days ago	16 hours ago	$SPLUNK_DB\cowrie\db	N/A	✓ Enabled

Fig 3) Splunk SIEM Index dashboard

By the above steps we successfully created index and generated token and configured the Cowrie file in order to get the logs from the honeypot cowrie and visualize the log.

6.) WSL and Putty Installation(Attacker Machines)

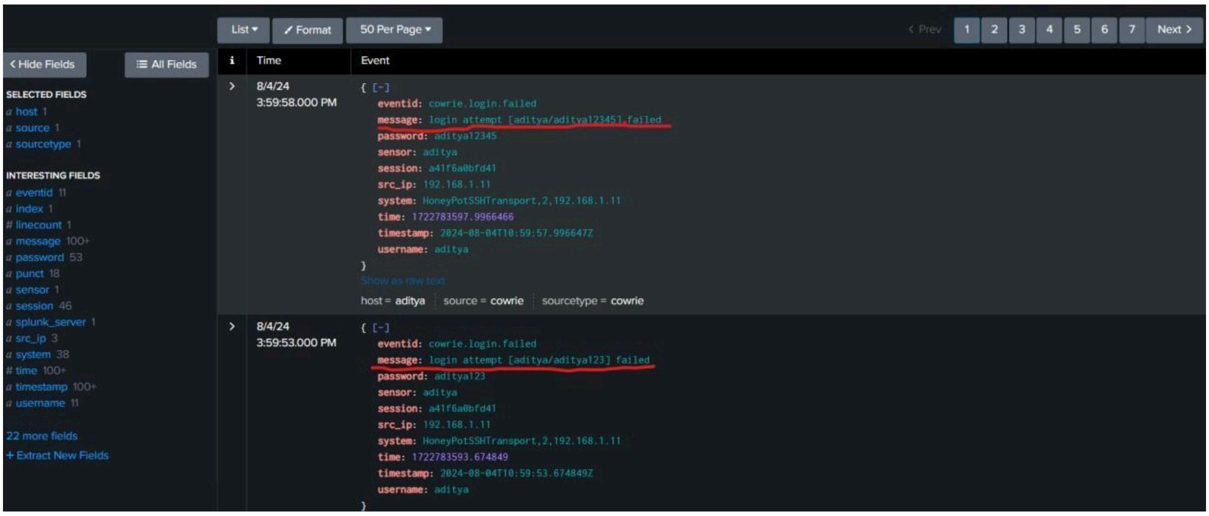
The WSL(Windows Subsystem for Linux)ubuntu Machine was installed in the Windows 11 system and performed various attacks on the Cowrie Honeypot to gather the logs and analyze, visualize the same in the splunk , Also we did install "Putty" in order to perform SSH Attack Manually on cowrie Honeypot. By these machine we generated various different logs which is shown in the Report.

7.) Data Analysis

The analysis of log data is important for the maintenance of cybersecurity and IT operations. Where system enables the real time monitoring and detection of anomalies, thereby providing the information for incident response Team and forensic investigations. Through comprehensive log analysis the organizations can ensure and match the requirements of the auditing standards. The Data analysis of log is very important in honeypots as it offers detailed insights into the activities of malicious actors and their behavioral patterns of the attacks. The analysis of these logs enables security teams to identify and handle the evolving threats.

In this Project we analyzed the Log data and found various type of Attacks as shown below:

- Failed Login Attempts can be seen from the Attacker machine .

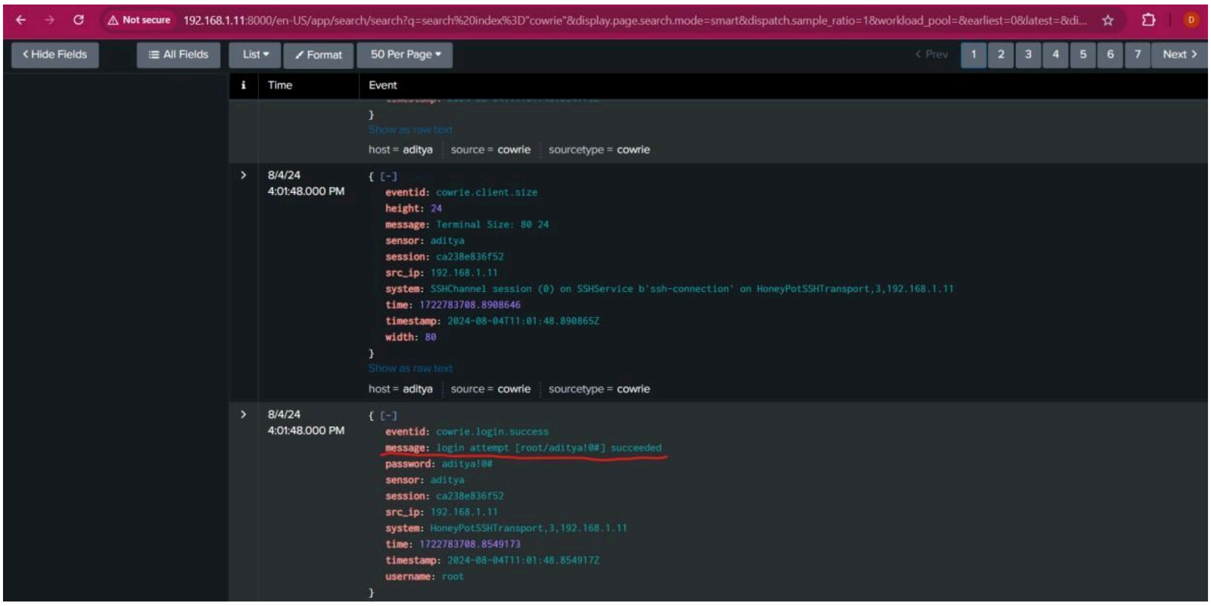


```
SELECTED FIELDS
a host 1
a source 1
a sourcetype 1

INTERESTING FIELDS
a eventid 11
a index 1
# linecount 1
a message 100+
a password 53
a punct 18
a sensor 1
a session 46
a splunk_server 1
a src_ip 3
a system 38
# time 100+
a timestamp 100+
a username 11
```

```
8/4/24 3:59:58.000 PM
{ [-]
   eventid: cowrie.login.failed
   message: login attempt [aditya/aditya12345] failed
   password: aditya12345
   sensor: aditya
   session: a41f6a0bfd41
   src_ip: 192.168.1.11
   system: HoneyPotSSHTransport,2,192.168.1.11
   time: 1722783597.9966466
   timestamp: 2024-08-04T10:59:57.996647Z
   username: aditya
}
host = aditya   source = cowrie   sourcetype = cowrie

8/4/24 3:59:53.000 PM
{ [-]
   eventid: cowrie.login.failed
   message: login attempt [aditya/aditya123] failed
   password: aditya123
   sensor: aditya
   session: a41f6a0bfd41
   src_ip: 192.168.1.11
   system: HoneyPotSSHTransport,2,192.168.1.11
   time: 1722783593.674849
   timestamp: 2024-08-04T10:59:53.674849Z
   username: aditya
}
```

Fig 4) Failed Login Attempts.

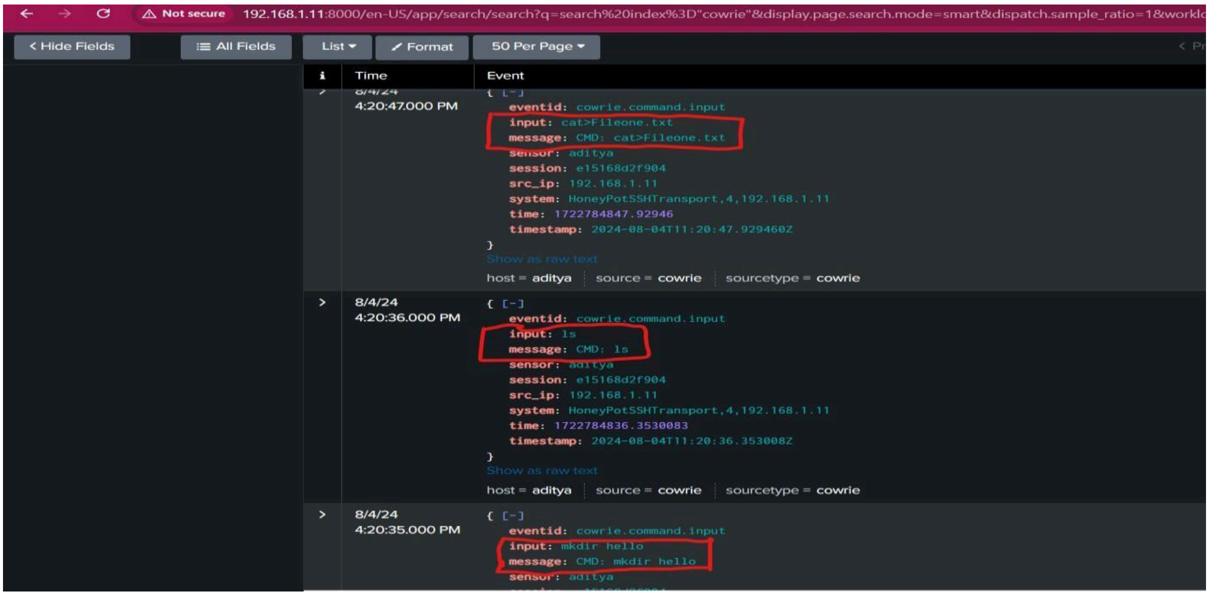
- Successful Logged in Logs, we performed Failed passwords and Successful attempts through SSH attack.


Not secure
192.168.1.11:8000/en-US/app/search/search?q=search%20index%3D"cowrie"&display.page.search.mode=smart&dispatch.sample_ratio=1&workload_pool=&earliest=0&latest=&di...
< Hide Fields
All Fields
List
Format
50 Per Page
< Prev
1
2
3
4
5
6
7
Next >
i
Time
Event
Show as raw text

```
}
host = aditya   source = cowrie   sourcetype = cowrie

8/4/24 4:01:48.000 PM
{ [-]
   eventid: cowrie.client.size
   height: 24
   message: Terminal Size: 80 24
   sensor: aditya
   session: ca238e836f52
   src_ip: 192.168.1.11
   system: SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,3,192.168.1.11
   time: 1722783708.8908646
   timestamp: 2024-08-04T11:01:48.890865Z
   width: 80
}
host = aditya   source = cowrie   sourcetype = cowrie

8/4/24 4:01:48.000 PM
{ [-]
   eventid: cowrie.login.success
   message: login attempt [root/aditya!@#] succeeded
   password: aditya!@#
   sensor: aditya
   session: ca238e836f52
   src_ip: 192.168.1.11
   system: HoneyPotSSHTransport,3,192.168.1.11
   time: 1722783708.8549173
   timestamp: 2024-08-04T11:01:48.854917Z
   username: root
}
```

Fig 5) Successful Login Attempt Logs.

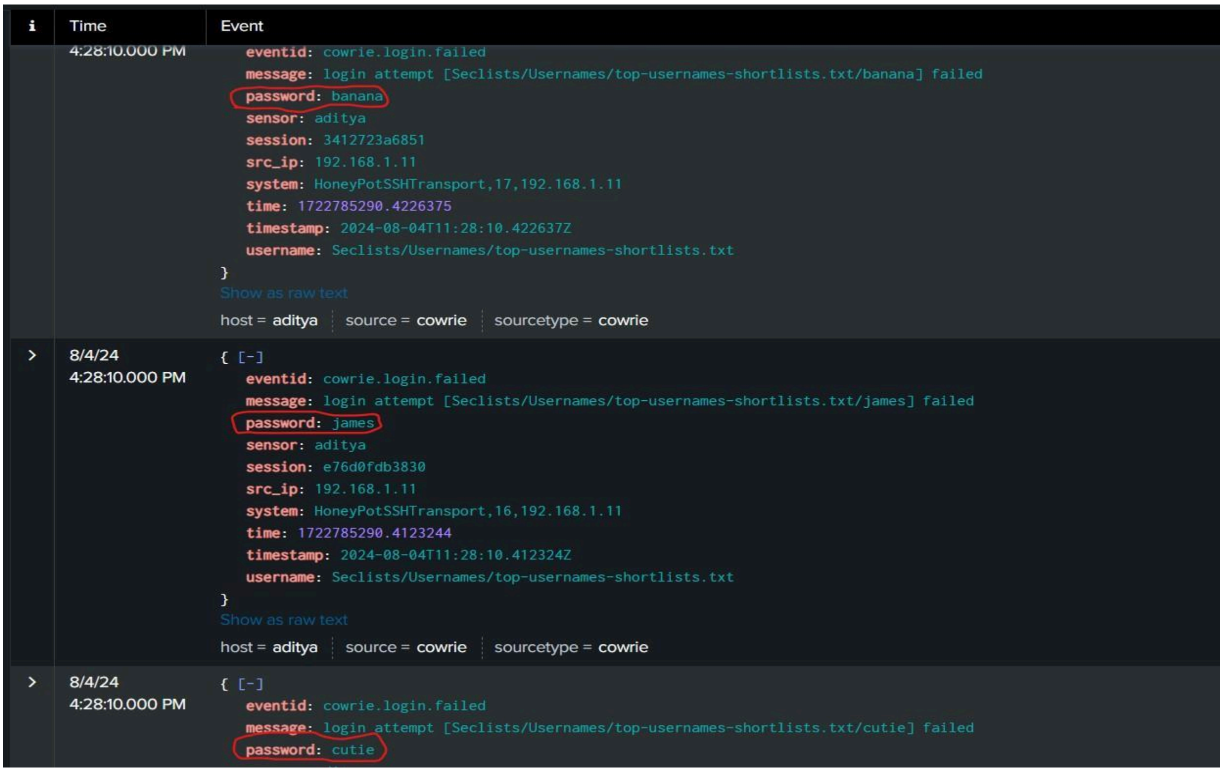
- After Successful Login We performed Creating a file inside the System which explains the Possibility of Command Injection Attacks and possibility of attacker to insert Malicious code.


Not secure
192.168.1.11:8000/en-US/app/search/search?q=search%20index%3D"cowrie"&display.page.search.mode=smart&dispatch.sample_ratio=1&workl
< Hide Fields
All Fields
List
Format
50 Per Page
< Pr

```
i   Time              Event
>   8/4/24            { [-]
    4:20:47.000 PM      eventid: cowrie.command.input
                        input: cat>Fileone.txt
                        message: CMD: cat>Fileone.txt
                        sensor: aditya
                        session: e15168d2f904
                        src_ip: 192.168.1.11
                        system: HoneyPotSSHTransport,4,192.168.1.11
                        time: 1722784847.92946
                        timestamp: 2024-08-04T11:20:47.929460Z
                      }
                      Show as raw text
                      host = aditya   source = cowrie   sourcetype = cowrie
>   8/4/24            { [-]
    4:20:36.000 PM      eventid: cowrie.command.input
                        input: ls
                        message: CMD: ls
                        sensor: aditya
                        session: e15168d2f904
                        src_ip: 192.168.1.11
                        system: HoneyPotSSHTransport,4,192.168.1.11
                        time: 1722784836.3530083
                        timestamp: 2024-08-04T11:20:36.353008Z
                      }
                      Show as raw text
                      host = aditya   source = cowrie   sourcetype = cowrie
>   8/4/24            { [-]
    4:20:35.000 PM      eventid: cowrie.command.input
                        input: mkdir hello
                        message: CMD: mkdir hello
                        sensor: aditya
```

Fig 6)Command Injection Log.

- Brute Force Attacks attempt for Password ,We Performed brute force attack from the WSL Ubuntu Machine that is a Attacker Machine by installing Hydra which is a automatic Password and Username Guessing tool the attempt can be seen below.

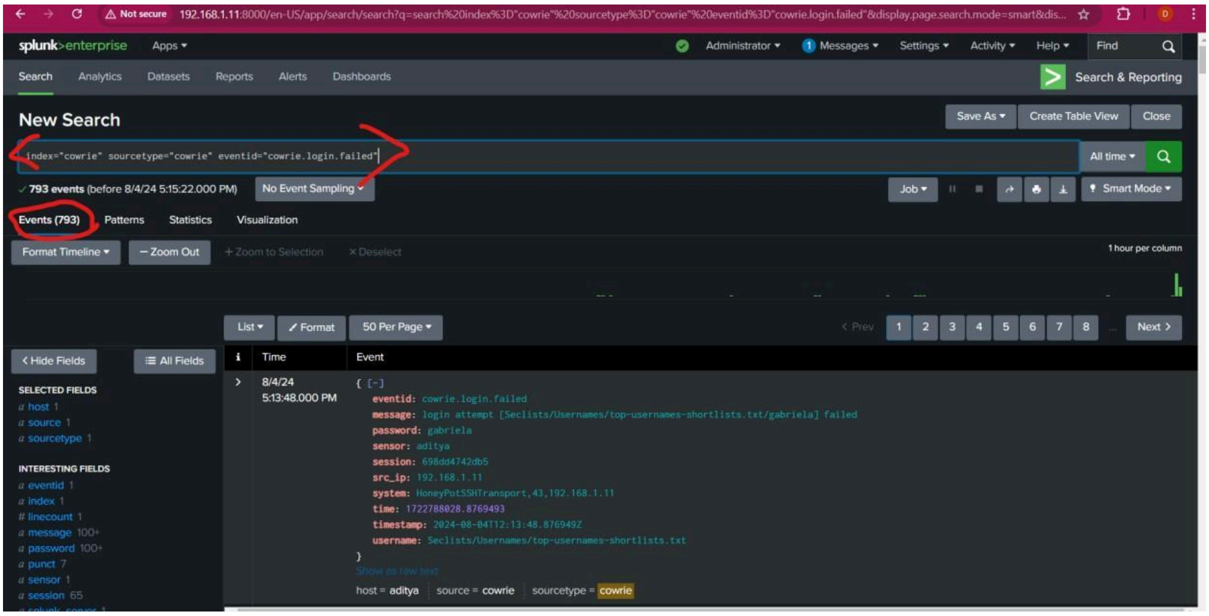


```
i   Time              Event
    4:28:10.000 PM      eventid: cowrie.login.failed
                        message: login attempt [Seclists/Usernames/top-usernames-shortlists.txt/banana] failed
                        password: banana
                        sensor: aditya
                        session: 3412723a6851
                        src_ip: 192.168.1.11
                        system: HoneyPotSSHTransport,17,192.168.1.11
                        time: 1722785290.4226375
                        timestamp: 2024-08-04T11:28:10.422637Z
                        username: Seclists/Usernames/top-usernames-shortlists.txt
                      }
                      Show as raw text
                      host = aditya   source = cowrie   sourcetype = cowrie
>   8/4/24            { [-]
    4:28:10.000 PM      eventid: cowrie.login.failed
                        message: login attempt [Seclists/Usernames/top-usernames-shortlists.txt/james] failed
                        password: james
                        sensor: aditya
                        session: e76d0fdb3830
                        src_ip: 192.168.1.11
                        system: HoneyPotSSHTransport,16,192.168.1.11
                        time: 1722785290.4123244
                        timestamp: 2024-08-04T11:28:10.412324Z
                        username: Seclists/Usernames/top-usernames-shortlists.txt
                      }
                      Show as raw text
                      host = aditya   source = cowrie   sourcetype = cowrie
>   8/4/24            { [-]
    4:28:10.000 PM      eventid: cowrie.login.failed
                        message: login attempt [Seclists/Usernames/top-usernames-shortlists.txt/cutie] failed
                        password: cutie
```

Fig 7)Brute Force attack Log.

8.) Result

After successfully Performing Different types of attacks we got the below Results, Information such as the Number of failed attempts, Wordlist which was performed by using Hydra was gathered. The total number of failed attempts which we performed are 793 .



```
index="cowrie" sourcetype="cowrie" eventid="cowrie.login.failed"
```

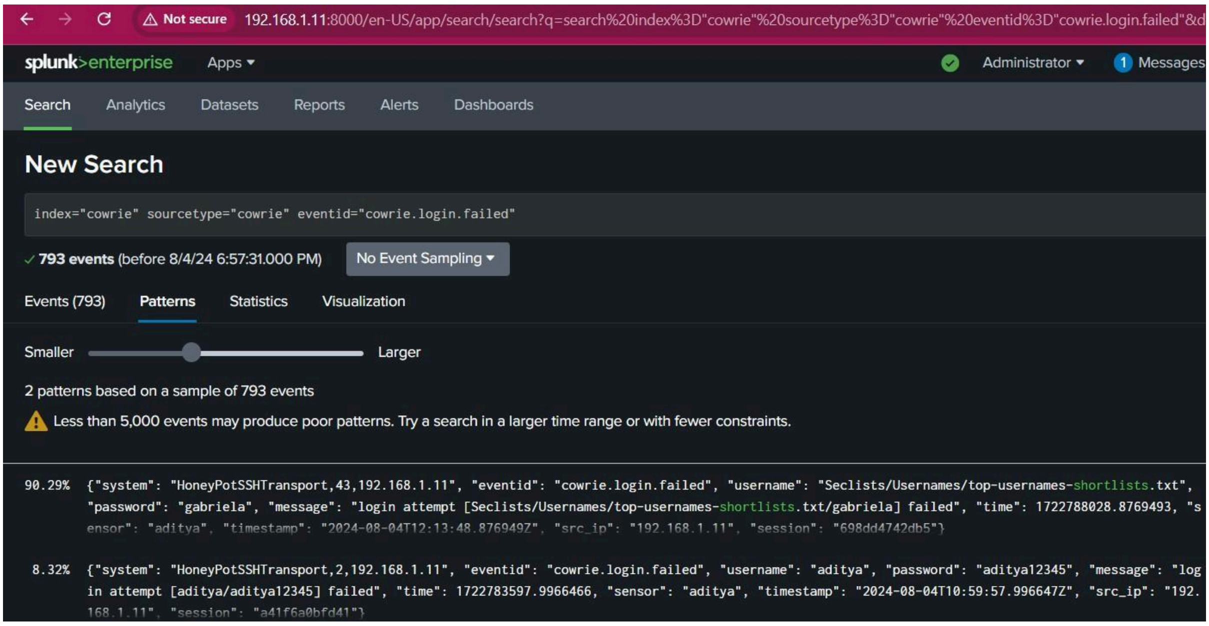
✓ 793 events (before 8/4/24 5:15:22.000 PM)

SELECTED FIELDS
- host 1
- source 1
- sourcetype 1

INTERESTING FIELDS
- eventid 1
- index 1
- linecount 1
- message 100+
- password 100+
- punct 7
- sensor 1
- session 65

```
8/4/24
5:13:48.000 PM
{ [-]
   eventid: cowrie.login.failed
   message: login attempt [Seclists/Usernames/top-usernames-shortlists.txt/gabriela] failed
   password: gabriela
   sensor: aditya
   session: 698dd4742db5
   src_ip: 192.168.1.11
   system: HoneyPotSSHTransport,43,192.168.1.11
   time: 1722788028.8769493
   timestamp: 2024-08-04T12:13:48.876949Z
   username: Seclists/Usernames/top-usernames-shortlists.txt
}
Show as raw text
host = aditya   source = cowrie   sourcetype = cowrie
```

Fig 8) Total Failed Login attempts(793).



```
index="cowrie" sourcetype="cowrie" eventid="cowrie.login.failed"
```

✓ 793 events (before 8/4/24 6:57:31.000 PM)

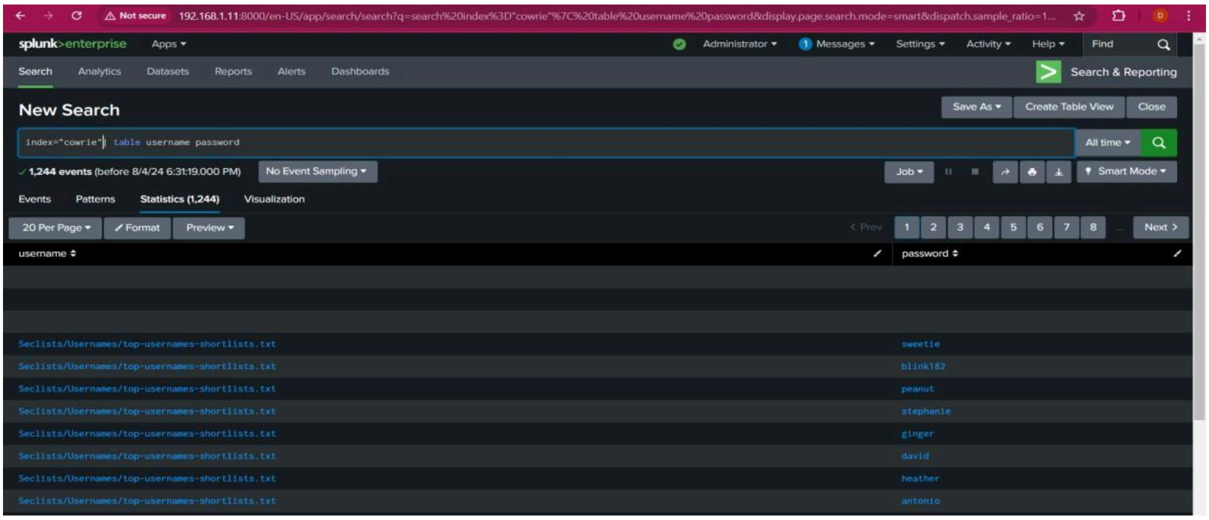
2 patterns based on a sample of 793 events

⚠ Less than 5,000 events may produce poor patterns. Try a search in a larger time range or with fewer constraints.

```
90.29%  {"system": "HoneyPotSSHTransport,43,192.168.1.11", "eventid": "cowrie.login.failed", "username": "Seclists/Usernames/top-usernames-shortlists.txt",
        "password": "gabriela", "message": "login attempt [Seclists/Usernames/top-usernames-shortlists.txt/gabriela] failed", "time": 1722788028.8769493, "s
        ensor": "aditya", "timestamp": "2024-08-04T12:13:48.876949Z", "src_ip": "192.168.1.11", "session": "698dd4742db5"}

 8.32%  {"system": "HoneyPotSSHTransport,2,192.168.1.11", "eventid": "cowrie.login.failed", "username": "aditya", "password": "aditya12345", "message": "log
        in attempt [aditya/aditya12345] failed", "time": 1722783597.9966466, "sensor": "aditya", "timestamp": "2024-08-04T10:59:57.996647Z", "src_ip": "192.
        168.1.11", "session": "a41f6a0bfd41"}
```

Fig 9) Patterns Identified for Login Failed

The Wordlist Result Which was performed using Hydra is shown below where the number of attempts for username and password performed is 1244. By using the query “index=”cowrie”| table username password” we can get the attempted details of username and password as shown below.



New Search

```
index="cowrie"| table username password
```

✓ 1,244 events (before 8/4/24 6:31:19.000 PM)

username	password
Seclists/Usernames/top-usernames-shortlists.txt	sweetie
Seclists/Usernames/top-usernames-shortlists.txt	blink182
Seclists/Usernames/top-usernames-shortlists.txt	peanut
Seclists/Usernames/top-usernames-shortlists.txt	stephanie
Seclists/Usernames/top-usernames-shortlists.txt	ginger
Seclists/Usernames/top-usernames-shortlists.txt	david
Seclists/Usernames/top-usernames-shortlists.txt	heather
Seclists/Usernames/top-usernames-shortlists.txt	antonio

Fig 10) Result of Hydra Attack attempts(1244).

G.) References

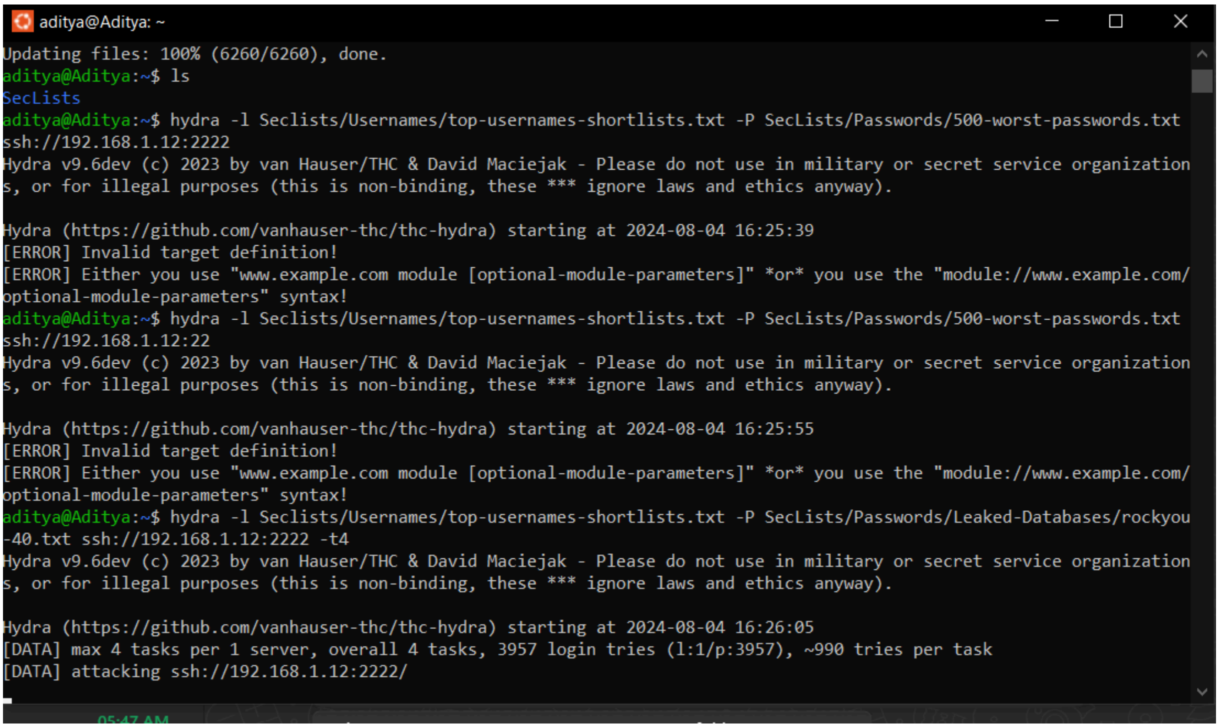
- https://securitytrails.com/blog/top-honeypots
- *dmesser - Repositories*. (n.d.). GitHub. https://github.com/dmesser?tab=repositories
- *Installing Cowrie in seven steps — cowrie 2.5.0 documentation*. (n.d.). https://cowrie.readthedocs.io/en/latest/INSTALL.html
- *Windows Subsystem for Linux (WSL) | Ubuntu*. (n.d.). Ubuntu. https://ubuntu.com/desktop/wsl
- *Free Splunk Trial | Download Splunk Enterprise Free for 60 days | Splunk*. (n.d.). Splunk. https://www.splunk.com/en_us/download/splunk-enterprise.html
- https://www.sophos.com/en-us/cybersecurity-explained/honeypots
- https://cowrie.readthedocs.io/en/latest/splunk/README.html
- *How to send Cowrie output to Splunk — cowrie 2.5.0 documentation*. (n.d.). https://cowrie.readthedocs.io/en/latest/splunk/README.html

10.) Conclusion

This report details the successful installation of the Cowrie honeypot and the integration of Splunk with the SIEM to gather and analyze logs. The logs were analyzed in order to identify the different types of attacks that had been attempted. These included brute force attacks, attempts to guess usernames and passwords, and logs of SSH connections that had been performed using PuTTY. The analysis of the failed attempts indicates that the attacker is attempting to compromise the system. The log analysis allows for the identification of the necessary actions to prevent further attacks. The successful login to the honeypot allows for the capture of the cybercriminal without any data loss or harm to the original server. The original server is safe where the hackers who believe honeypot to be a genuine server and attempt to gain access. By integrating the honeypot with the Splunk, it is possible to analyze the activities occurring on the server and identify any attempted breaches. This integration also facilitates the reporting of incidents, which can help to improve the security of an organization. Thus we can concluded that the integration of a SIEM with a honeypot and log analysis is a crucial aspect of cyber security.

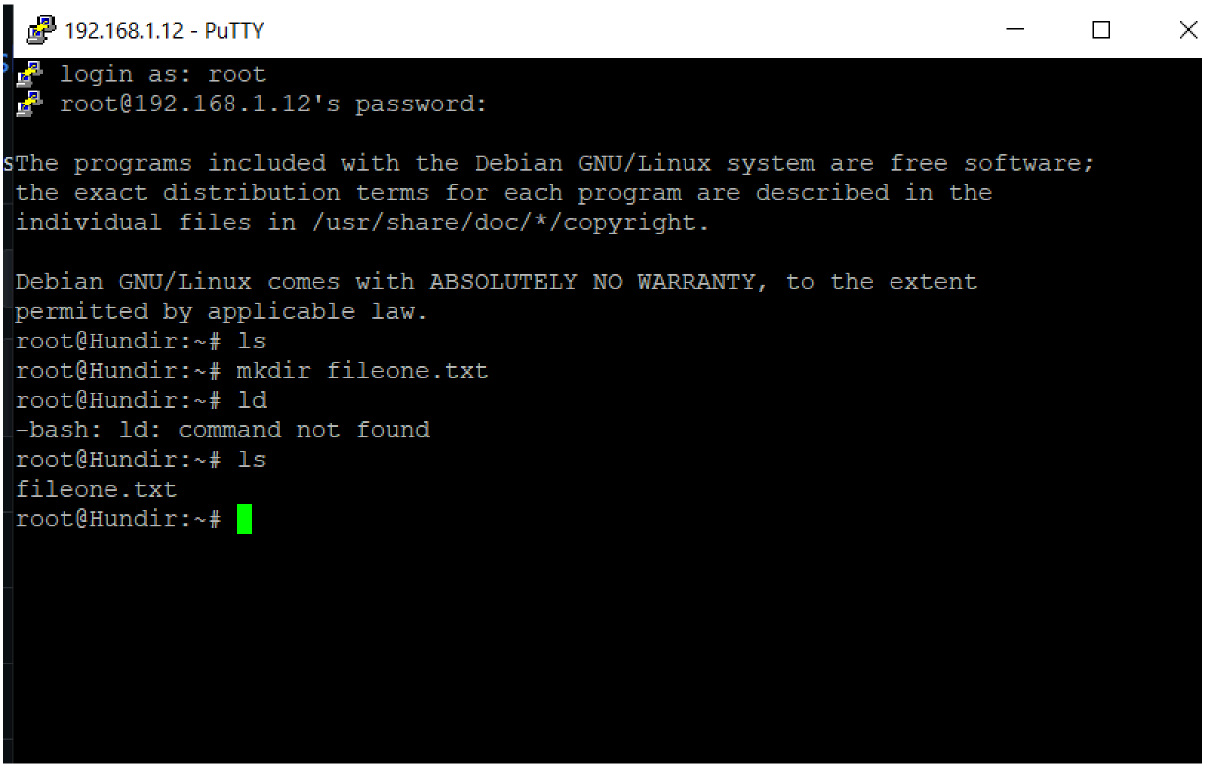
11.) Appendices

- **Screenshots of Some Proof of Software Installed and attack performed**



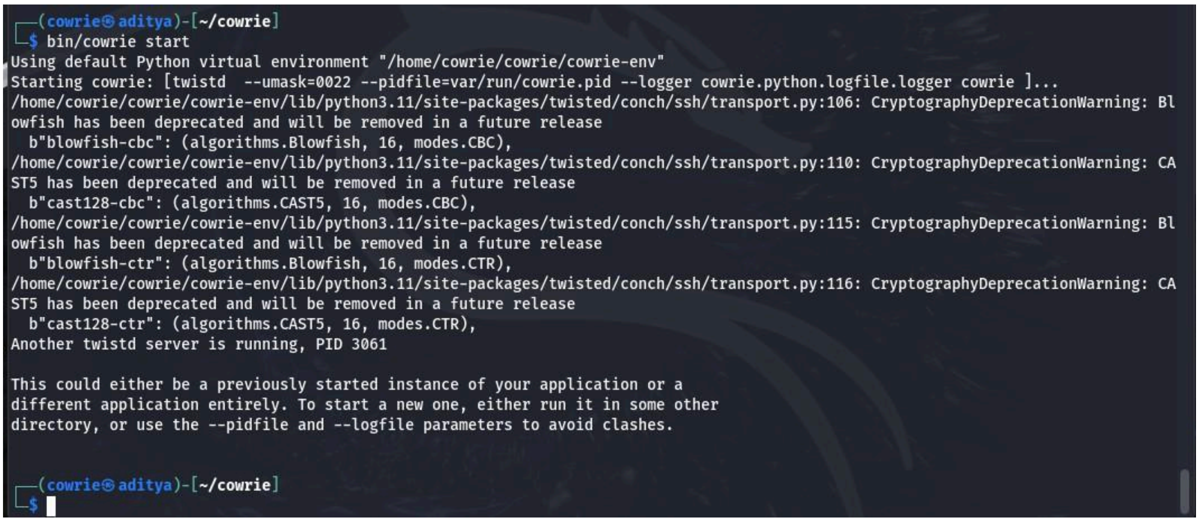
```
Updating files: 100% (6260/6260), done.
aditya@Aditya:~$ ls
SecLists
aditya@Aditya:~$ hydra -l Seclists/Usernames/top-usernames-shortlists.txt -P SecLists/Passwords/500-worst-passwords.txt
ssh://192.168.1.12:2222
Hydra v9.6dev (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organization
s, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-08-04 16:25:39
[ERROR] Invalid target definition!
[ERROR] Either you use "www.example.com module [optional-module-parameters]" *or* you use the "module://www.example.com/
optional-module-parameters" syntax!
aditya@Aditya:~$ hydra -l Seclists/Usernames/top-usernames-shortlists.txt -P SecLists/Passwords/500-worst-passwords.txt
ssh://192.168.1.12:22
Hydra v9.6dev (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organization
s, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-08-04 16:25:55
[ERROR] Invalid target definition!
[ERROR] Either you use "www.example.com module [optional-module-parameters]" *or* you use the "module://www.example.com/
optional-module-parameters" syntax!
aditya@Aditya:~$ hydra -l Seclists/Usernames/top-usernames-shortlists.txt -P SecLists/Passwords/Leaked-Databases/rockyou
-40.txt ssh://192.168.1.12:2222 -t4
Hydra v9.6dev (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organization
s, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-08-04 16:26:05
[DATA] max 4 tasks per 1 server, overall 4 tasks, 3957 login tries (l:1/p:3957), ~990 tries per task
[DATA] attacking ssh://192.168.1.12:2222/
```

Fig 11) Performing Hydra Username and password attack through WSL Ubuntu Machine.



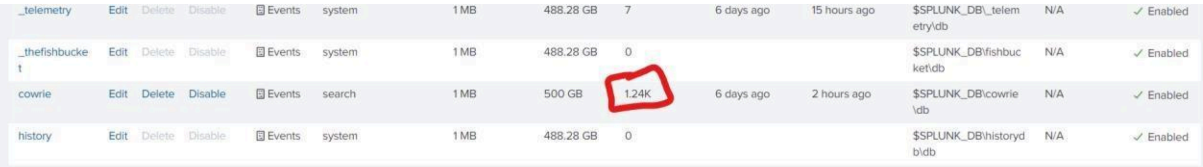
```
login as: root
root@192.168.1.12's password:

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.
root@Hundir:~# ls
root@Hundir:~# mkdir fileone.txt
root@Hundir:~# ld
-bash: ld: command not found
root@Hundir:~# ls
fileone.txt
root@Hundir:~#
```

Fig 12) SSH Login Successful Putty software



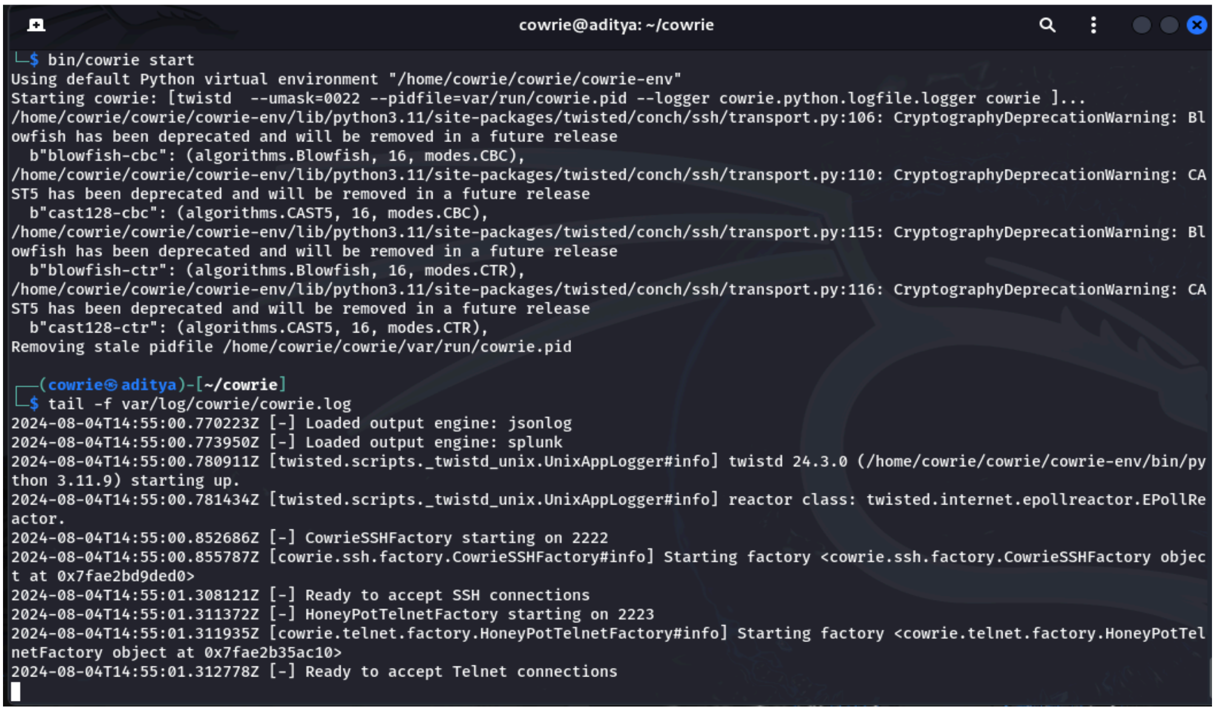
```
┌──(cowrie㉿aditya)-[~/cowrie]
└─$ bin/cowrie start
Using default Python virtual environment "/home/cowrie/cowrie/cowrie-env"
Starting cowrie: [twistd  --umask=0022 --pidfile=var/run/cowrie.pid --logger cowrie.python.logfile.logger cowrie ]...
/home/cowrie/cowrie/cowrie-env/lib/python3.11/site-packages/twisted/conch/ssh/transport.py:106: CryptographyDeprecationWarning: Bl
owfish has been deprecated and will be removed in a future release
  b"blowfish-cbc": (algorithms.Blowfish, 16, modes.CBC),
/home/cowrie/cowrie/cowrie-env/lib/python3.11/site-packages/twisted/conch/ssh/transport.py:110: CryptographyDeprecationWarning: CA
ST5 has been deprecated and will be removed in a future release
  b"cast128-cbc": (algorithms.CAST5, 16, modes.CBC),
/home/cowrie/cowrie/cowrie-env/lib/python3.11/site-packages/twisted/conch/ssh/transport.py:115: CryptographyDeprecationWarning: Bl
owfish has been deprecated and will be removed in a future release
  b"blowfish-ctr": (algorithms.Blowfish, 16, modes.CTR),
/home/cowrie/cowrie/cowrie-env/lib/python3.11/site-packages/twisted/conch/ssh/transport.py:116: CryptographyDeprecationWarning: CA
ST5 has been deprecated and will be removed in a future release
  b"cast128-ctr": (algorithms.CAST5, 16, modes.CTR),
Another twistd server is running, PID 3061

This could either be a previously started instance of your application or a
different application entirely. To start a new one, either run it in some other
directory, or use the --pidfile and --logfile parameters to avoid clashes.

┌──(cowrie㉿aditya)-[~/cowrie]
└─$ 
```

Fig 13) Successful Cowrie Installed and Running.



_telemetry	Edit	Delete	Disable	Events	system	1 MB	488.28 GB	7	6 days ago	15 hours ago	$SPLUNK_DB_telemetry\db	N/A	✓ Enabled
_thefishbucket	Edit	Delete	Disable	Events	system	1 MB	488.28 GB	0			$SPLUNK_DB\fishbucket\db	N/A	✓ Enabled
cowrie	Edit	Delete	Disable	Events	search	1 MB	500 GB	1.24K	6 days ago	2 hours ago	$SPLUNK_DB\cowrie\db	N/A	✓ Enabled
history	Edit	Delete	Disable	Events	system	1 MB	488.28 GB	0			$SPLUNK_DB\historydb\db	N/A	✓ Enabled

Fig 14) Total number of Log Generated.



```
└─$ bin/cowrie start
Using default Python virtual environment "/home/cowrie/cowrie/cowrie-env"
Starting cowrie: [twistd  --umask=0022 --pidfile=var/run/cowrie.pid --logger cowrie.python.logfile.logger cowrie ]...
/home/cowrie/cowrie/cowrie-env/lib/python3.11/site-packages/twisted/conch/ssh/transport.py:106: CryptographyDeprecationWarning: Bl
owfish has been deprecated and will be removed in a future release
  b"blowfish-cbc": (algorithms.Blowfish, 16, modes.CBC),
/home/cowrie/cowrie/cowrie-env/lib/python3.11/site-packages/twisted/conch/ssh/transport.py:110: CryptographyDeprecationWarning: CA
ST5 has been deprecated and will be removed in a future release
  b"cast128-cbc": (algorithms.CAST5, 16, modes.CBC),
/home/cowrie/cowrie/cowrie-env/lib/python3.11/site-packages/twisted/conch/ssh/transport.py:115: CryptographyDeprecationWarning: Bl
owfish has been deprecated and will be removed in a future release
  b"blowfish-ctr": (algorithms.Blowfish, 16, modes.CTR),
/home/cowrie/cowrie/cowrie-env/lib/python3.11/site-packages/twisted/conch/ssh/transport.py:116: CryptographyDeprecationWarning: CA
ST5 has been deprecated and will be removed in a future release
  b"cast128-ctr": (algorithms.CAST5, 16, modes.CTR),
Removing stale pidfile /home/cowrie/cowrie/var/run/cowrie.pid

┌──(cowrie㉿aditya)-[~/cowrie]
└─$ tail -f var/log/cowrie/cowrie.log
2024-08-04T14:55:00.770223Z [-] Loaded output engine: jsonlog
2024-08-04T14:55:00.773950Z [-] Loaded output engine: splunk
2024-08-04T14:55:00.780911Z [twisted.scripts._twistd_unix.UnixAppLogger#info] twistd 24.3.0 (/home/cowrie/cowrie/cowrie-env/bin/py
thon 3.11.9) starting up.
2024-08-04T14:55:00.781434Z [twisted.scripts._twistd_unix.UnixAppLogger#info] reactor class: twisted.internet.epollreactor.EPollRe
actor.
2024-08-04T14:55:00.852686Z [-] CowrieSSHFactory starting on 2222
2024-08-04T14:55:00.855787Z [cowrie.ssh.factory.CowrieSSHFactory#info] Starting factory <cowrie.ssh.factory.CowrieSSHFactory objec
t at 0x7fae2bd9ded0>
2024-08-04T14:55:01.308121Z [-] Ready to accept SSH connections
2024-08-04T14:55:01.311372Z [-] HoneyPotTelnetFactory starting on 2223
2024-08-04T14:55:01.311935Z [cowrie.telnet.factory.HoneyPotTelnetFactory#info] Starting factory <cowrie.telnet.factory.HoneyPotTel
netFactory object at 0x7fae2b35ac10>
2024-08-04T14:55:01.312778Z [-] Ready to accept Telnet connections
```

Fig 15) Cowrie Ready to accept telnet and SSH Connections.