EXPLOTACIÓN DE VULNERABILIDAD DE FRITILEAKS

EDWARD CAMILO CAICEDO BENITEZ

UNIVERSIDAD TECNOLÓGICA DEL CHOCÓ DIEGO LUIS CÓRDOBA
FACULTAD DE INGENIERÍA
PROGRAMA DE INGENIERÍA DE TELECOMUNICACIÓNES E INFORMÁTICA

SEGURIDAD Y AUDITORIAS EN REDES

DOCENTE

RAFAEL SANDOVAL MORALES

QUIBDÓ-CHOCÓ-COLOMBIA

2025

TABLA DE CONTENIDO

TABLA DE ILUSTRACIONES

INTRODUCCIÓN

En este laboratorio de seguridad informática se realizó un recorrido por un entorno controlado con el fin de comprender cómo funcionan ciertos mecanismos de protección y, al mismo tiempo, descubrir posibles debilidades en un servidor web. La práctica comenzó con la exploración de directorios y archivos disponibles públicamente, lo cual permitió identificar recursos expuestos y rutas ocultas dentro de la aplicación. A medida que se avanzaba, se presentaron diferentes mensajes, imágenes y formularios que fueron dando pistas sobre el camino a seguir.

OBJETIVO

Analizar y documentar el proceso de exploración y acceso a un servidor web en un entorno controlado, con el propósito de comprender cómo se descubren rutas ocultas, formularios de login y posibles vulnerabilidades que permitan mejorar las medidas de seguridad en aplicaciones similares.

METODOLOGIA

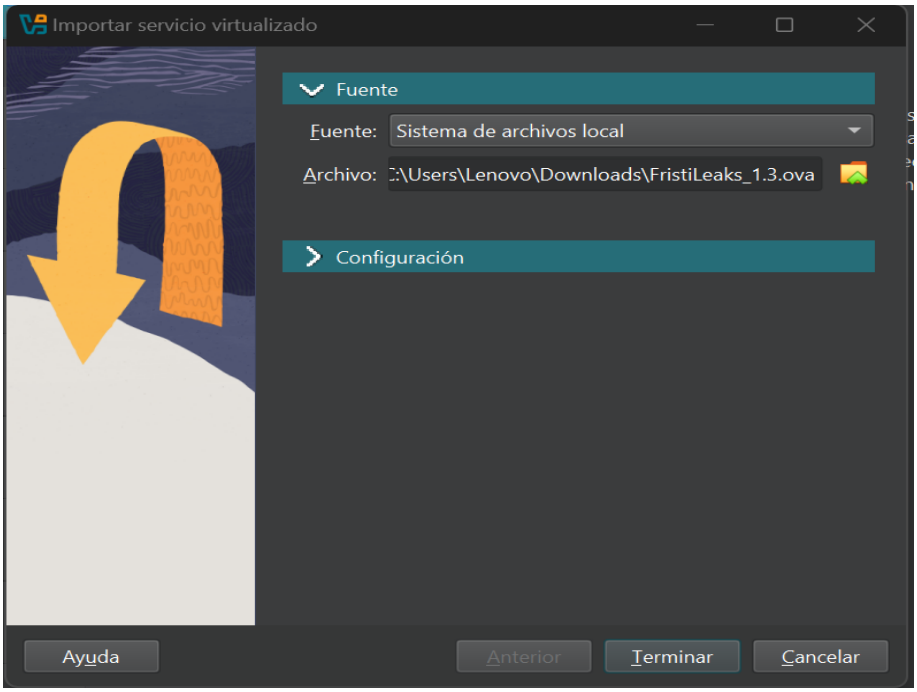
La práctica se llevó a cabo siguiendo una serie de pasos ordenados que simulan el proceso de un pentest:

1. **Exploración inicial**: se listaron los directorios accesibles públicamente, identificando archivos de imágenes y recursos disponibles.
2. **Revisión de robots.txt**: se analizaron las rutas restringidas indicadas por el archivo de configuración, lo que permitió descubrir directorios adicionales.

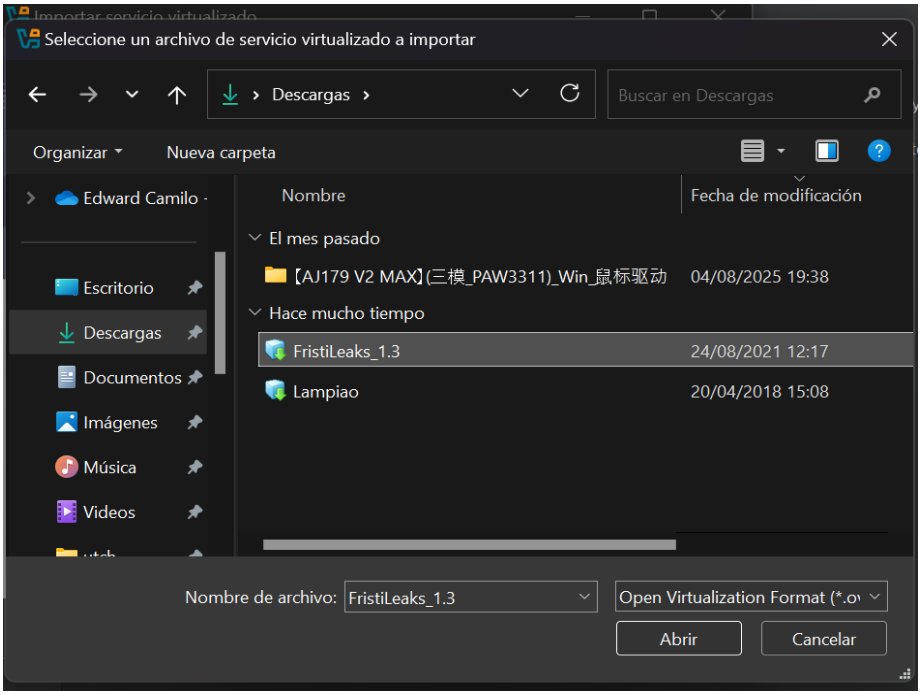
3. **Acceso a rutas ocultas**: se visitaron las URL indicadas en robots.txt, observando mensajes personalizados que servían como pistas dentro del entorno.
4. **Interacción con formularios**: se probaron credenciales en el login presentado por el sistema, lo que permitió acceder con distintos usuarios y observar el flujo de autenticación.
5. **Validación del acceso**: se comprobó que, tras un login exitoso, el sistema ofrecía la posibilidad de subir archivos, lo que abre la puerta a un análisis más profundo sobre el manejo de permisos y seguridad en el servidor.

INSTALACIÓN Y CONFIGURACIÓN DE FRITILEAKS

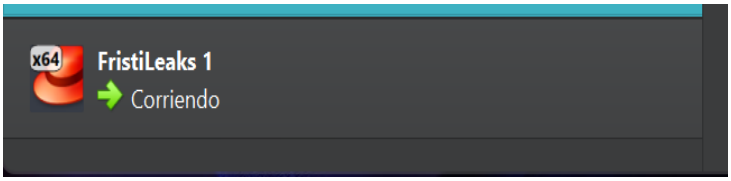
Paso 1: importamos fristileaks e instalación de friktileaks



1. importación de fristileaks en VirtualBox



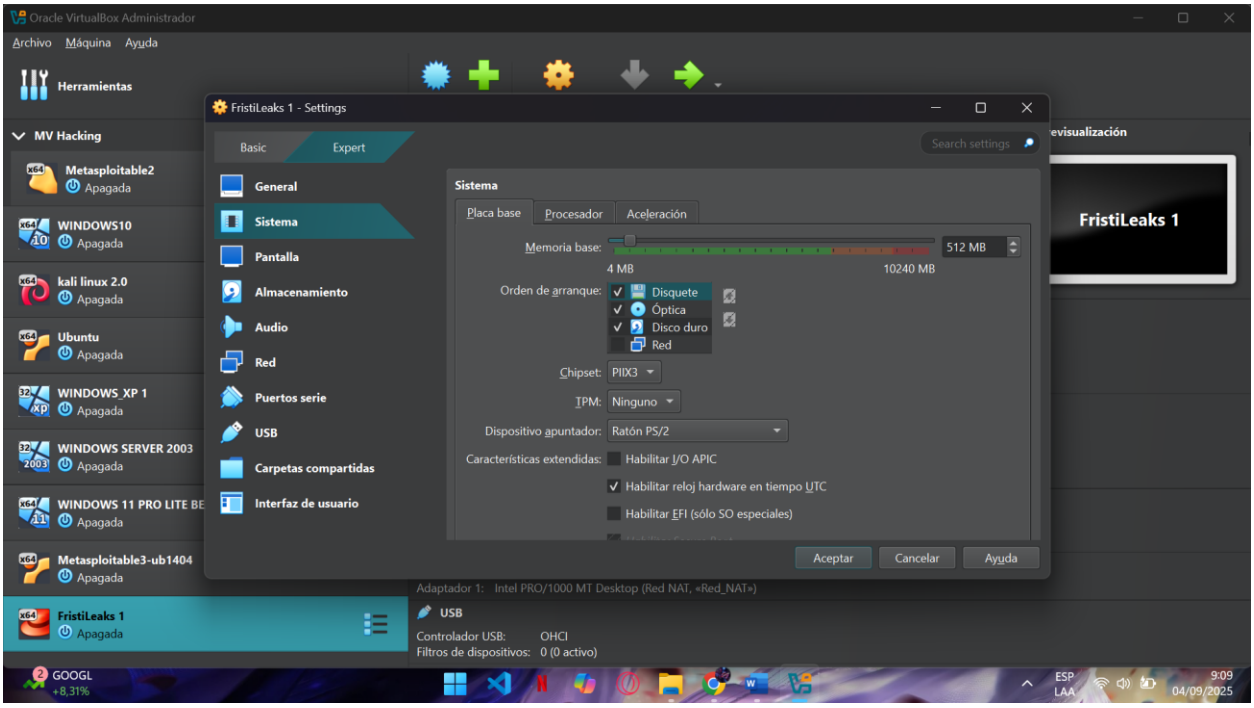
2. Seleccionamos el archivo de Fristileaks



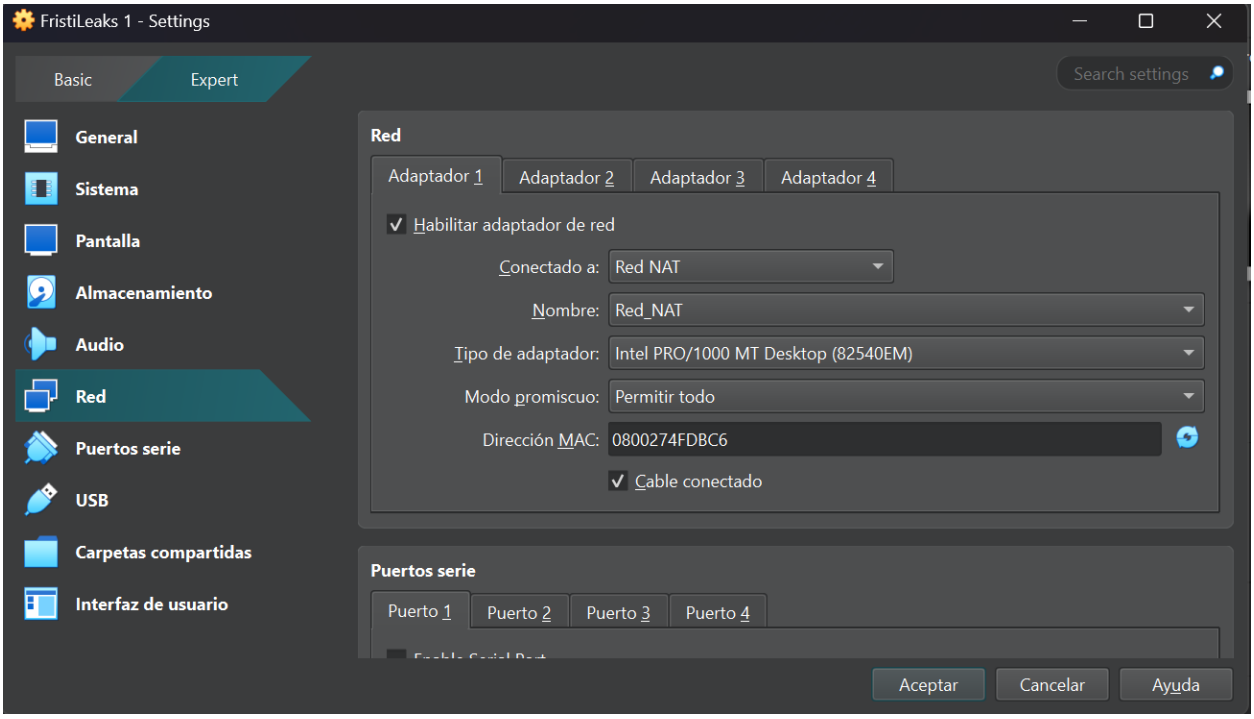
3. Instalación del exitosa del fristileaks

VULNERACIÓN DE FRITILEAKS POR MEDIO DE INYECCION SQL

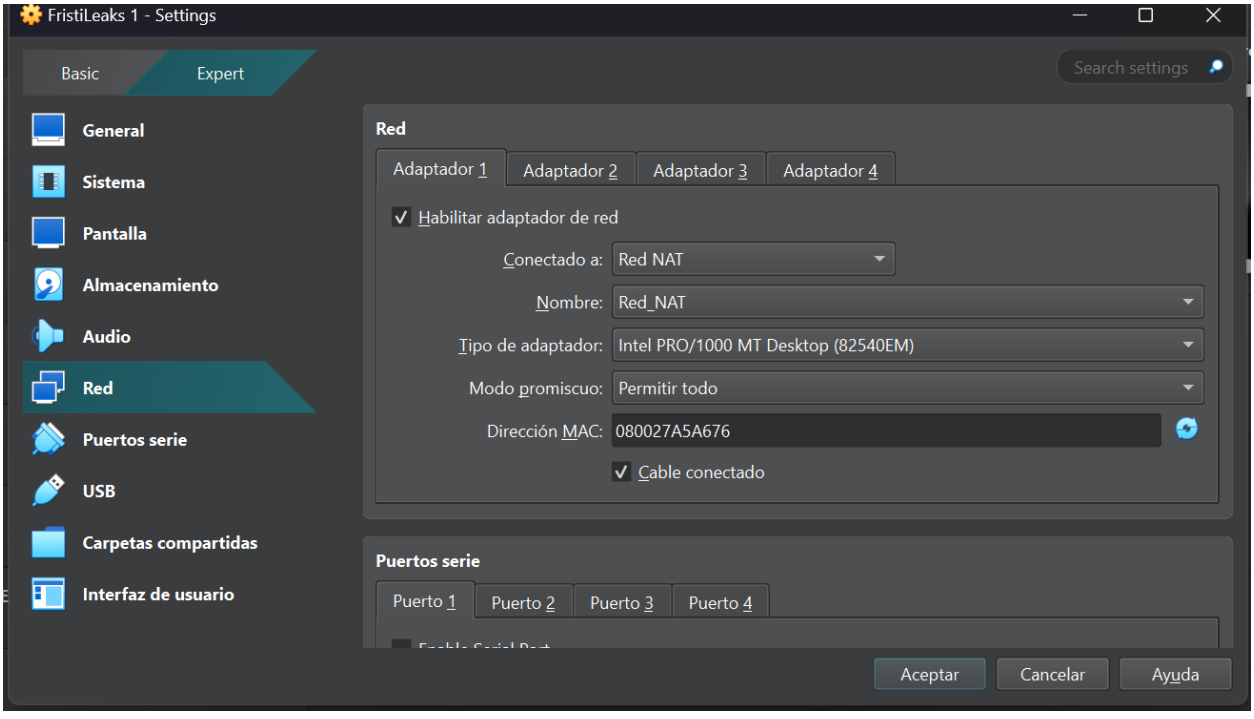
Paso1: configuramos y ejecutamos fritilleks



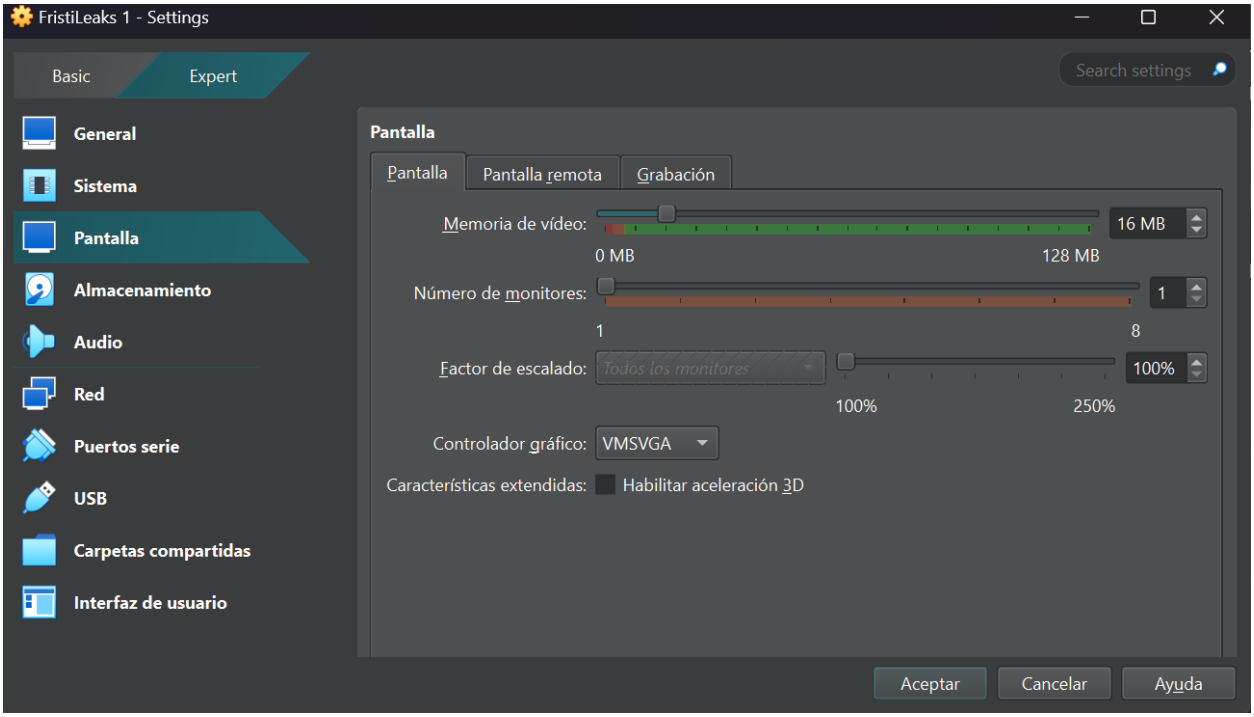
4. Configuración del sistema de fritileaks en VirtualBox



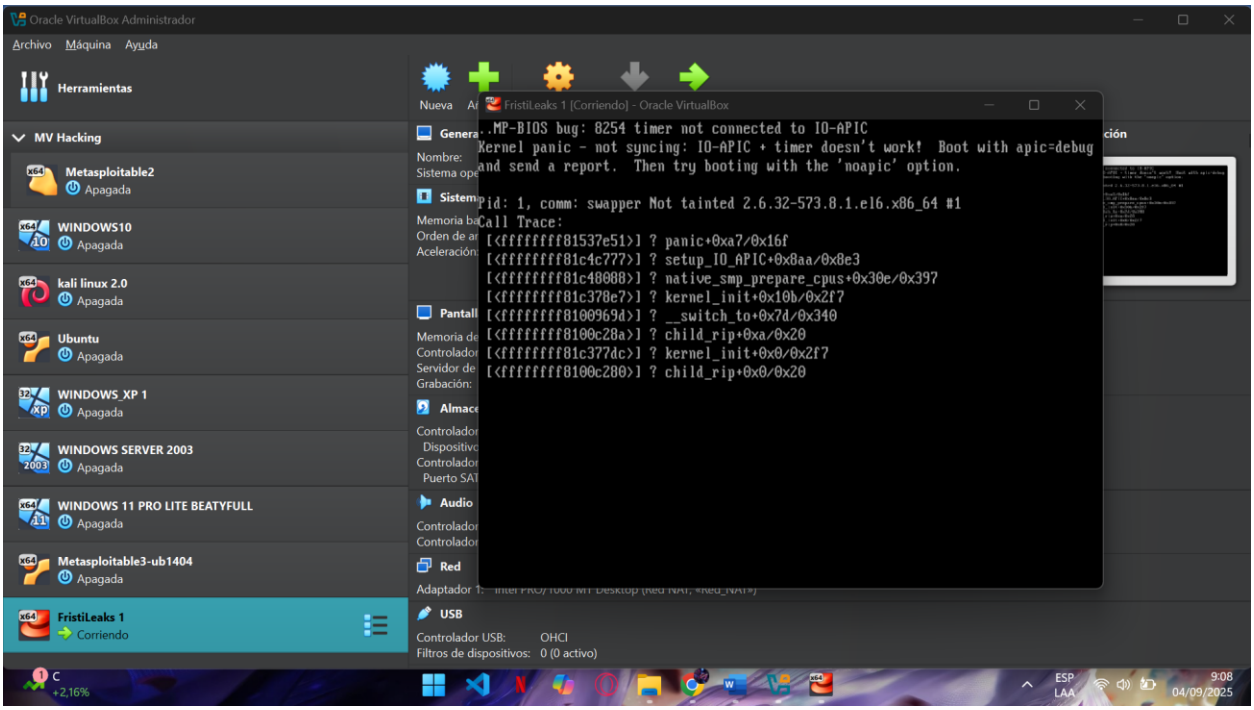
5. Configuración de red de fritileaks en VirtualBox



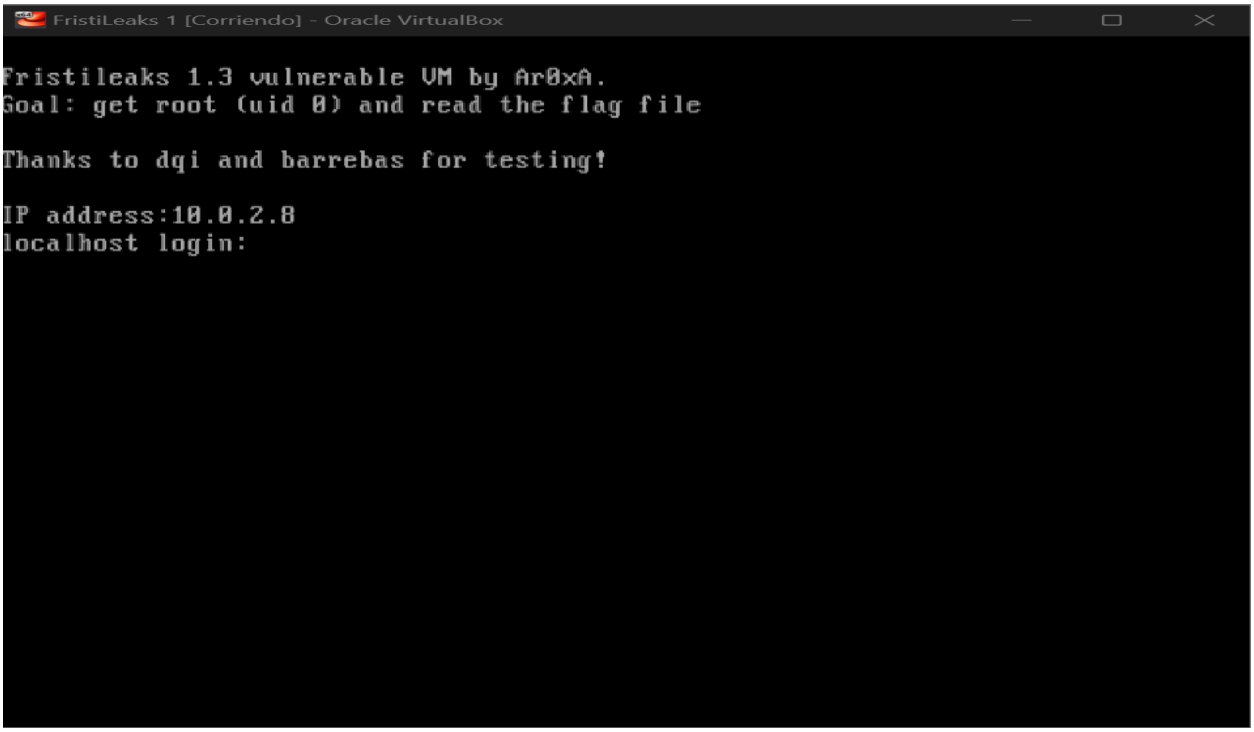
6.Configuración de red de fritileaks en VirtualBox



7. Configuración de pantalla de fritileaks en VirtualBox

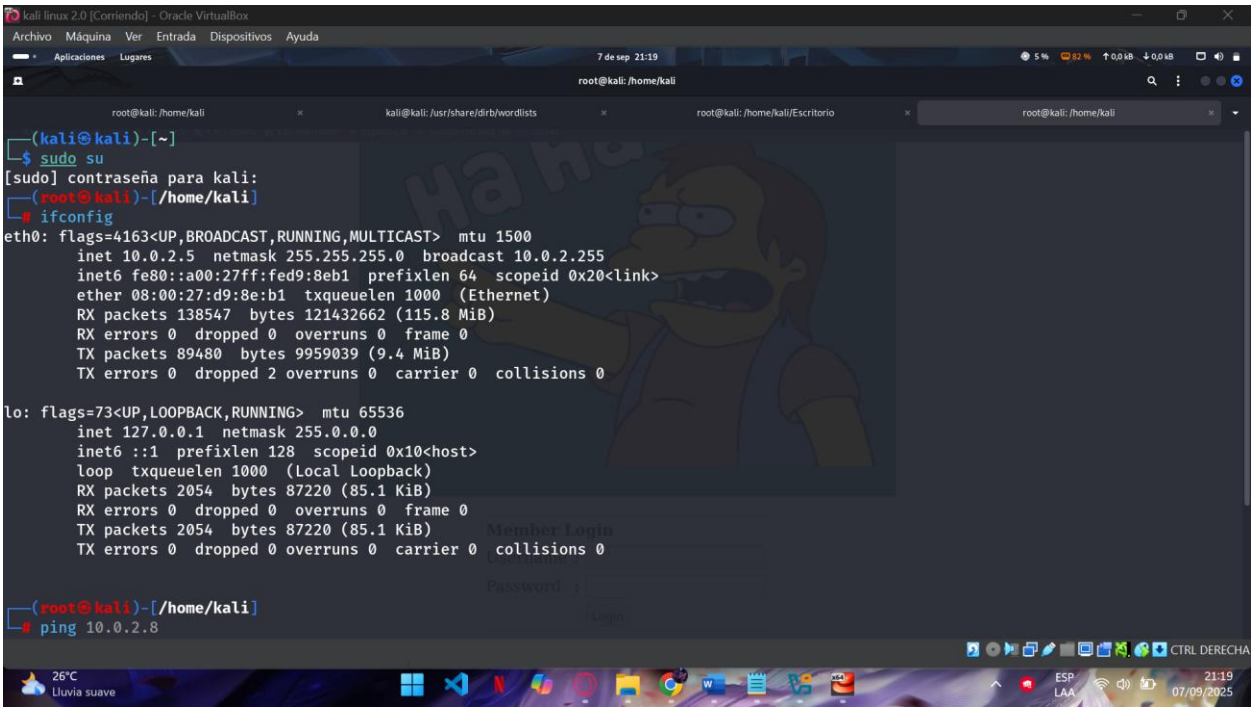


8. Ejecución de fritileaks en VirtualBox



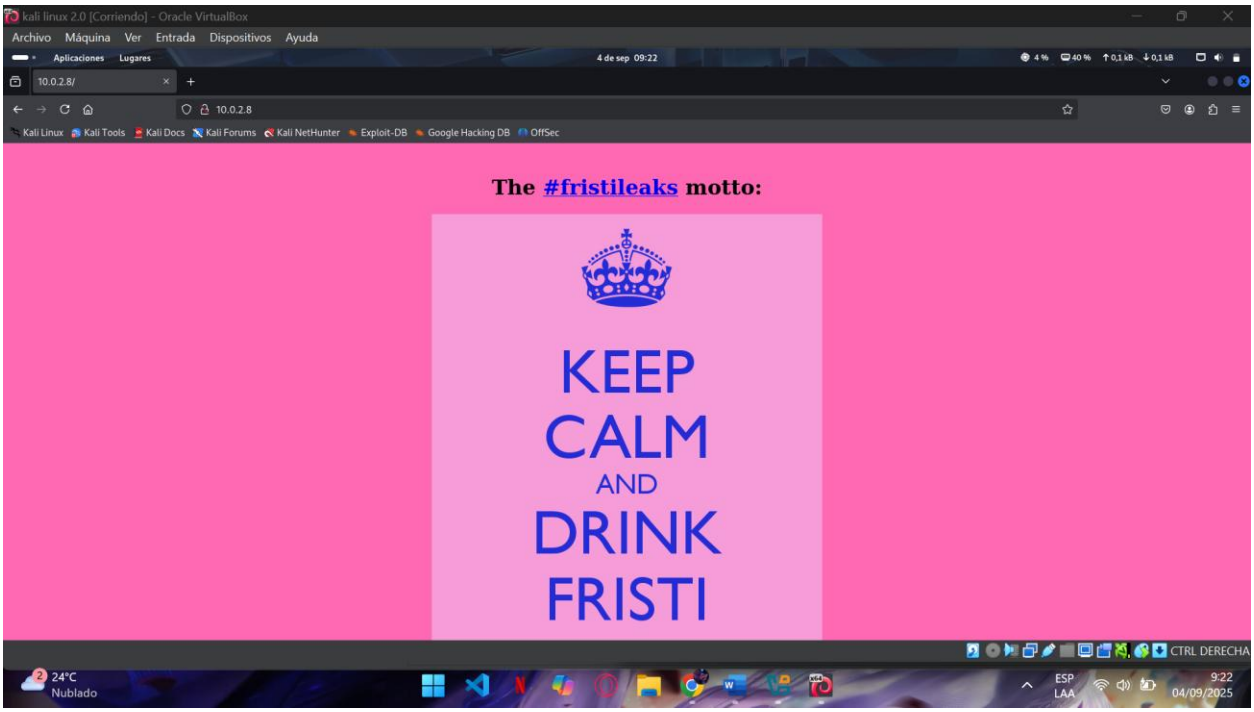
9. fritileaks abierto

Paso 2: Iniciamos a Kali y damos ping a la ip de vagrant para confirmar la conexión entre estos



```
┌──(root㉿kali)-[/home/kali]
└─# ping 10.0.2.8
PING 10.0.2.8 (10.0.2.8) 56(84) bytes of data.
64 bytes from 10.0.2.8: icmp_seq=1 ttl=64 time=7.38 ms
64 bytes from 10.0.2.8: icmp_seq=2 ttl=64 time=2.28 ms
64 bytes from 10.0.2.8: icmp_seq=3 ttl=64 time=1.50 ms
^Z
zsh: suspended  ping 10.0.2.8

┌──(root㉿kali)-[/home/kali]
└─#
```

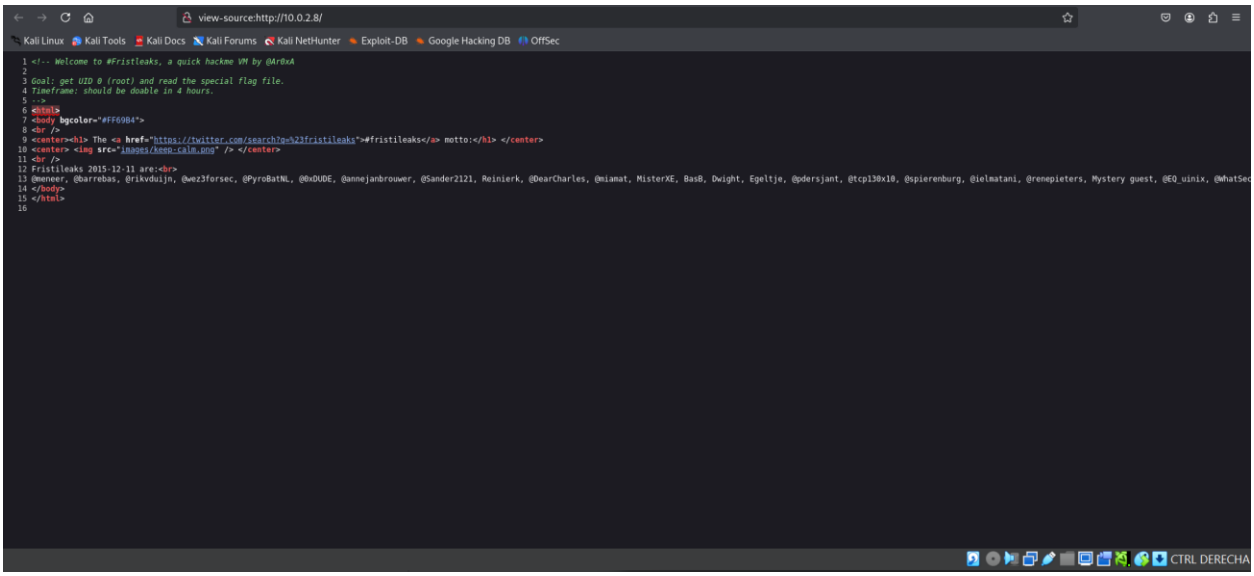
Paso 3: una vez con conexión entre ambas maquinas, se observa el acceso exitoso al servicio web de **Fristileaks** mediante la dirección IP 10.0.2.8, dentro del entorno de práctica configurado en **Kali Linux ejecutándose en VirtualBox**. La página muestra el lema característico “Keep Calm and Drink Fristi”, lo que confirma que el servidor se encuentra activo y accesible desde el navegador.



10. Acceso a la página Fristileaks desde el navegador en Kali Linux

Paso 4: a el código fuente del sitio web **Fristileaks**, accedido a través del navegador en la dirección 10.0.2.8. El contenido HTML revela información relevante, incluyendo enlaces, estructura de la página y referencias a archivos e imágenes (logo-fristi.png). Además, se muestra un bloque de

texto con posibles nombres de usuario, lo cual resulta clave para las fases de recolección de información y análisis de vulnerabilidades en la práctica.



11. Visualización del código fuente de la página Fristileaks

Paso 5: En este paso se utilizó la herramienta **Dirb** desde Kali Linux para realizar una búsqueda de directorios y archivos ocultos en el servidor web **Fristileaks** (http://10.0.2.8/). La exploración identificó recursos importantes, entre ellos:

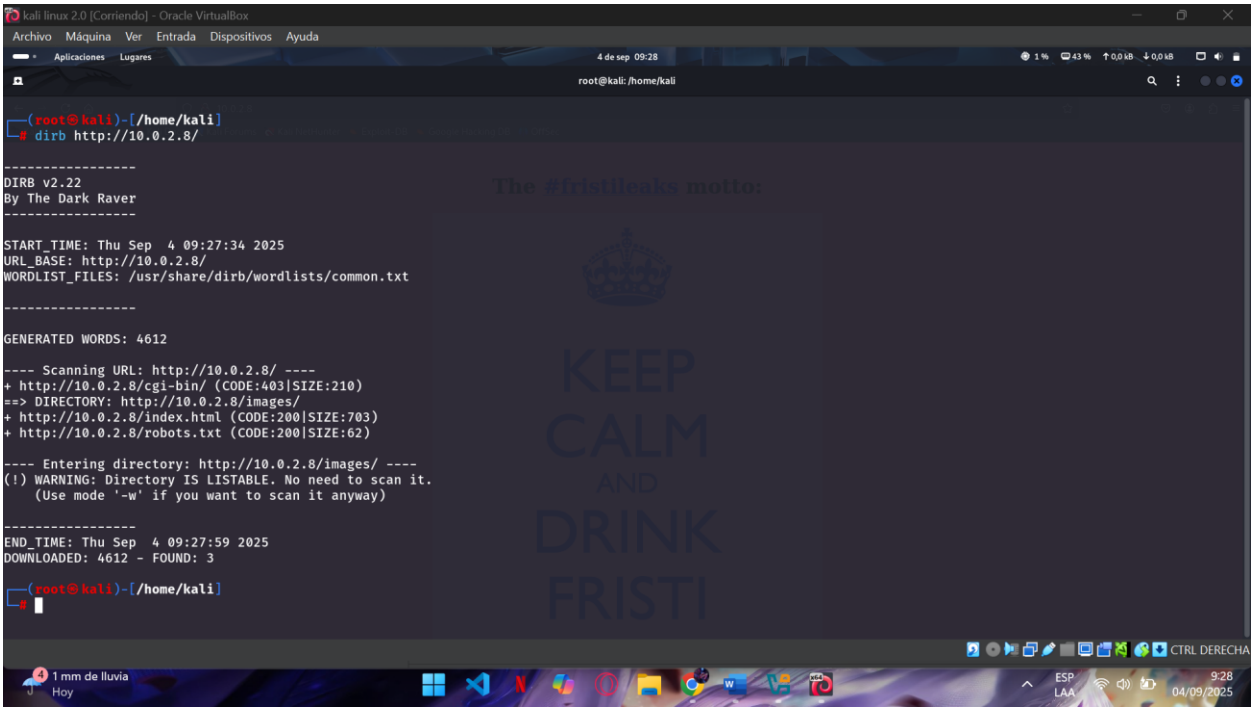
/cgi-bin/ (con código de respuesta 403 – acceso prohibido).

/images/ (directorio accesible y listable).

/index.html (página principal del sitio).

/robots.txt (archivo accesible con código 200).

Estos hallazgos son relevantes porque permiten descubrir posibles rutas y archivos sensibles dentro del servidor, que pueden ser aprovechados para avanzar en la práctica de análisis de seguridad.



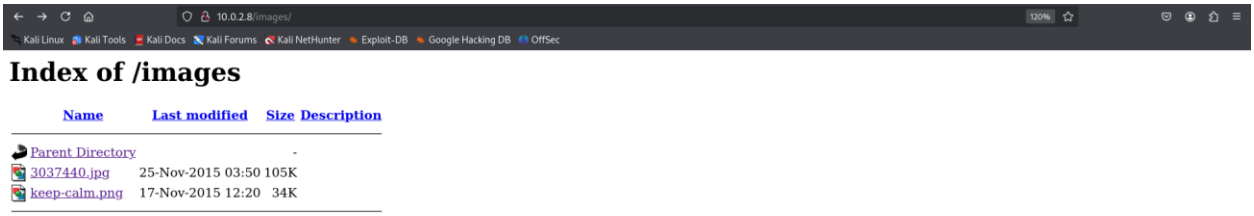
12. Ejecución de dirb para enumeración de directorios

Paso 6: En este paso se accedió a los recursos identificados dentro del archivo robots.txt del servidor **Fristileaks**. Dicho archivo contenía tres rutas restringidas:

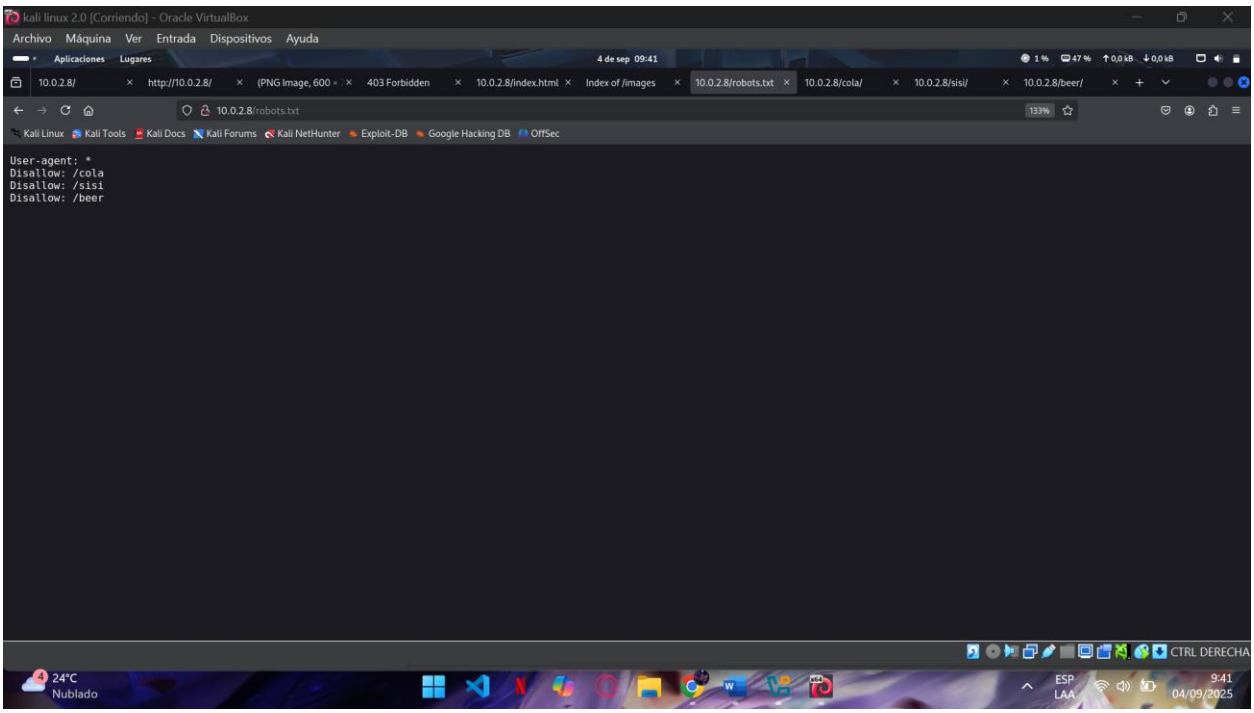
- /cola
- /sisi
- /beer

Al ingresar en cada una de ellas, el servidor desplegó una misma imagen con un mensaje en formato de meme: **“This is not the URL you were looking for”** (Este no es el URL que estabas buscando).

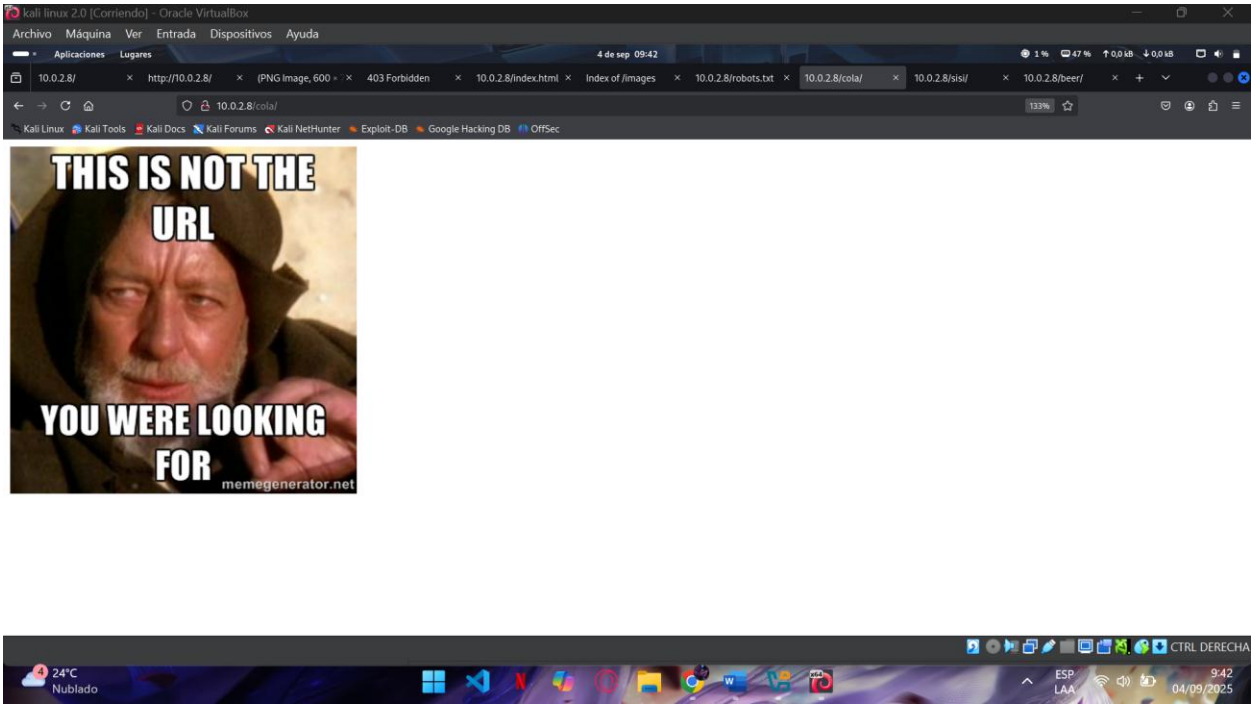
Esto evidencia que las rutas listadas en robots.txt funcionan como señuelos o trampas, y no contienen información sensible directa. Sin embargo, el hallazgo confirma la necesidad de seguir analizando directorios y archivos para encontrar datos ocultos que sí resulten útiles en la práctica de seguridad



13. "Exploración de directorio web expuesto: /images



14. Revisión del archivo robots.txt en el servidor web

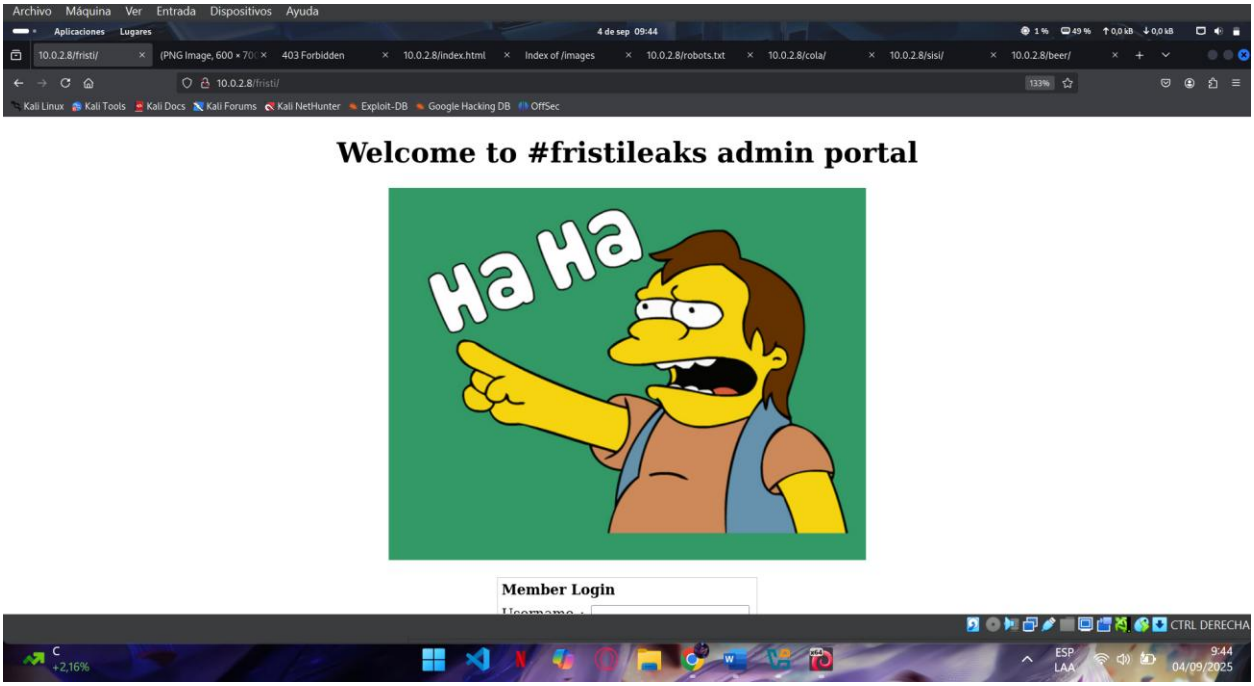


15. Acceso denegado con mensaje personalizado en URL restringida

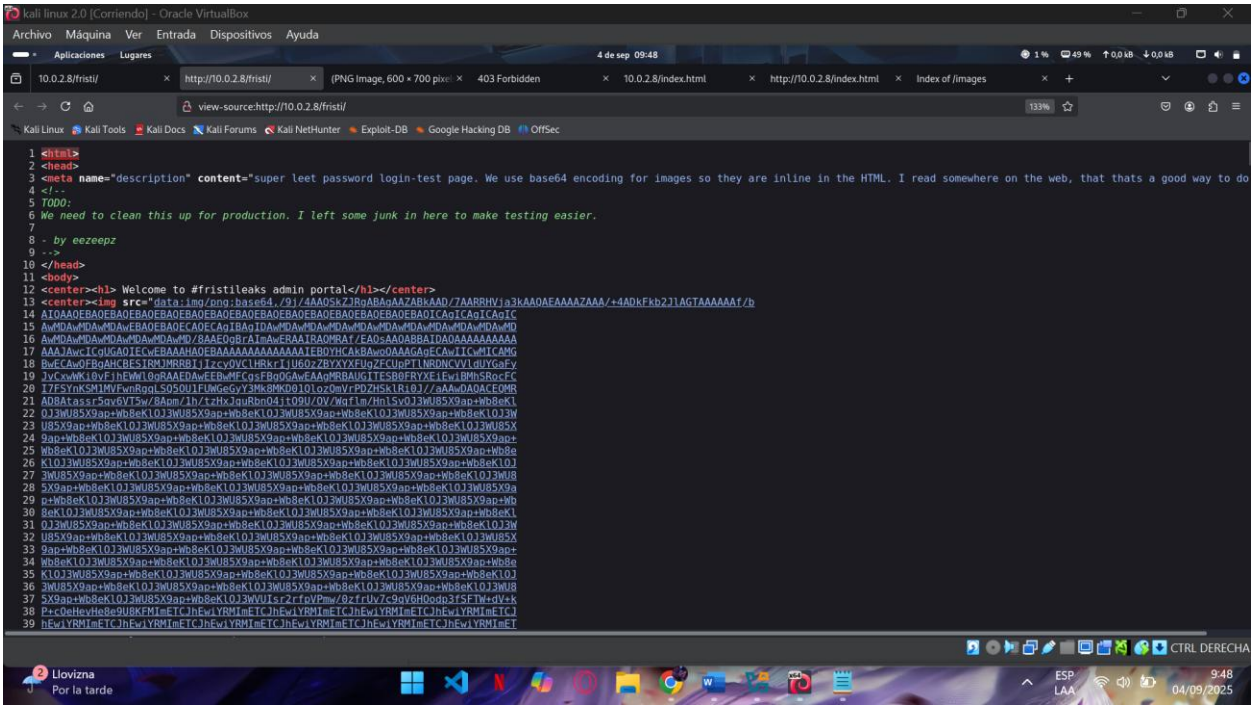
Paso 7 En este paso se accedió al **portal de administración de Fristileaks** ubicado en la ruta /fristi/.

- Inicialmente, la página muestra un formulario de inicio de sesión acompañado de una imagen humorística.
- Al inspeccionar el código fuente, se detectó un bloque en **Base64** que contenía información incrustada.
- Dicho bloque fue decodificado utilizando herramientas en línea, revelando la contraseña: **kekekekekekekkek** y el usuario **eezeepz**.
- Con estas credenciales se completó exitosamente el acceso al portal, mostrando el mensaje **"Login successful"** y habilitando la opción de **subir archivos (upload file)**.

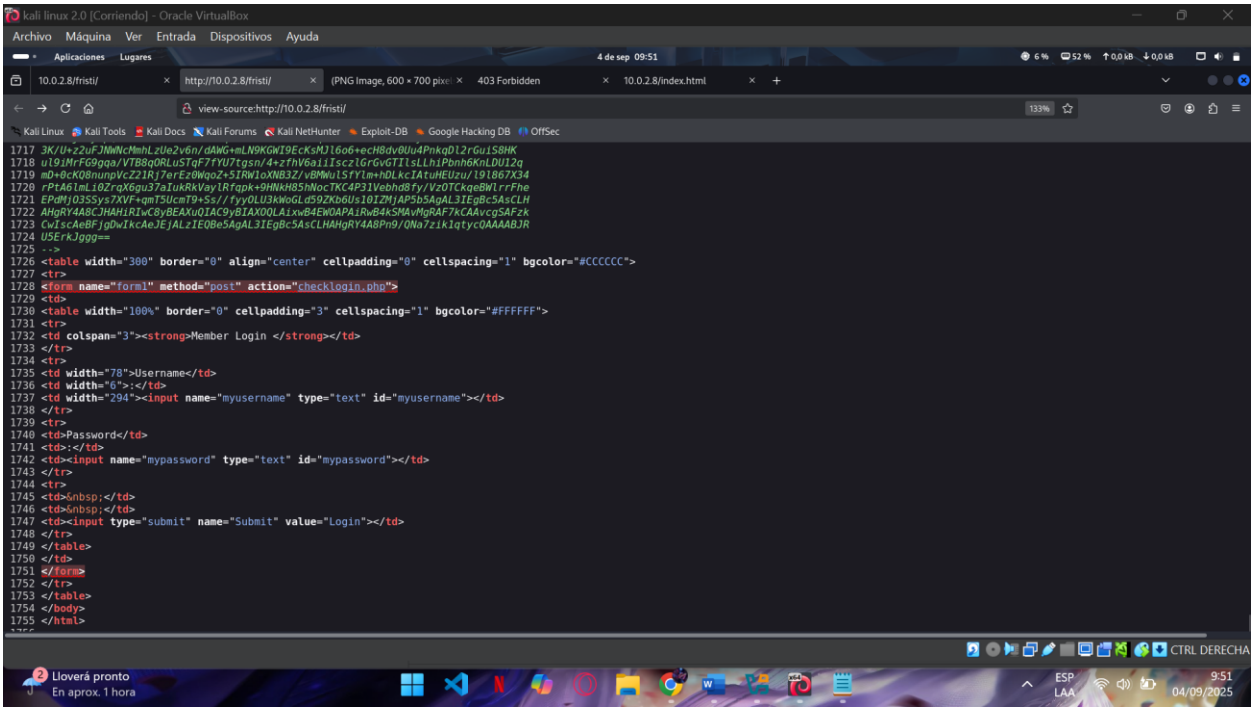
Este hallazgo marca un punto crítico en la práctica, ya que el ingreso al área administrativa permite probar vulnerabilidades adicionales, como la carga de archivos maliciosos para obtener acceso al sistema.



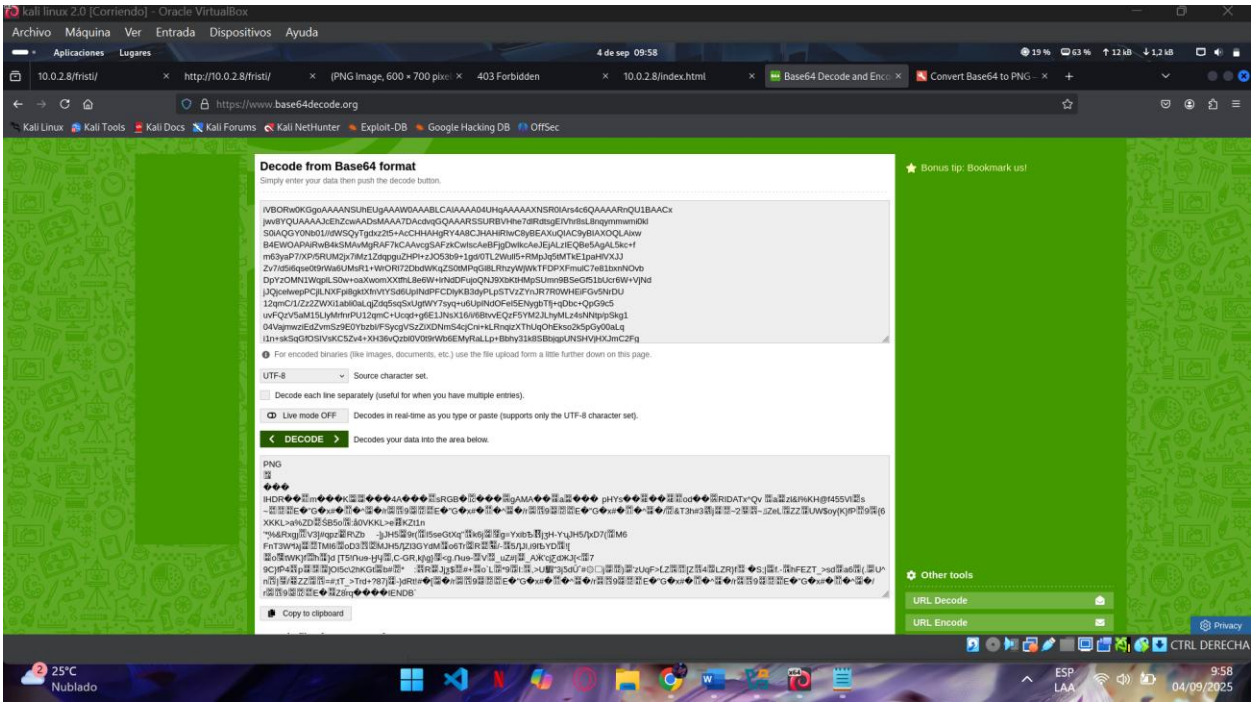
16. Portal de administración de Fristileaks



17. Código fuente con datos en Base64

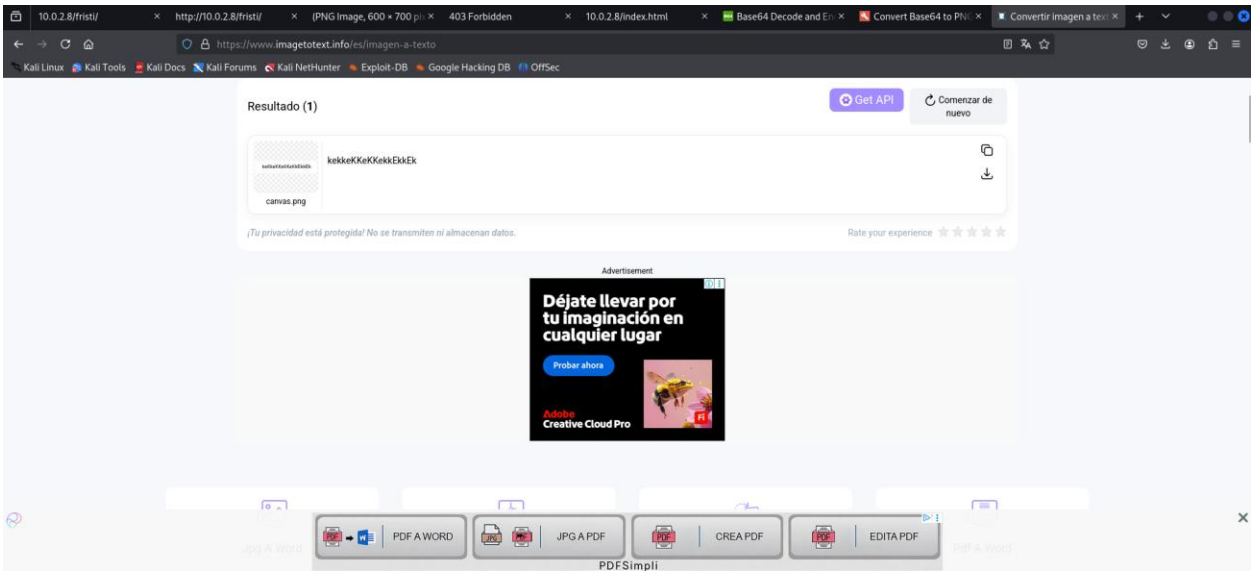


18. Código fuente con formulario de login

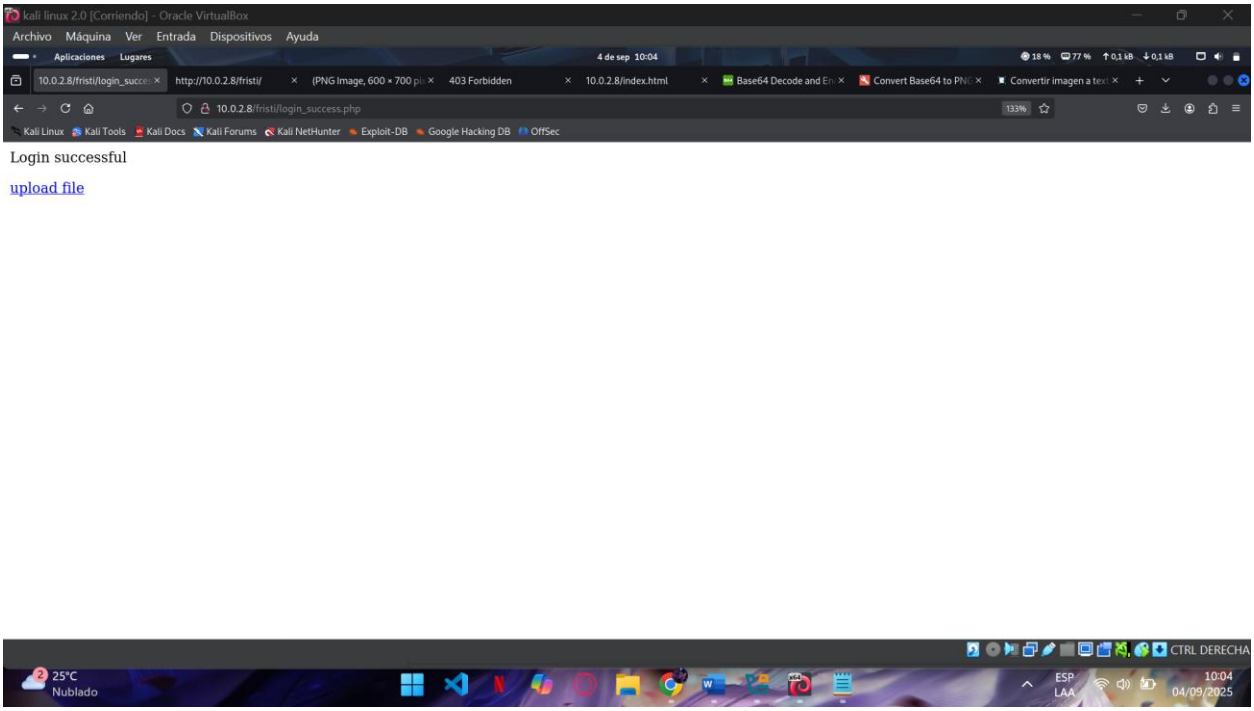


19. Intento de decodificación de Base64

20. Conversión Base64 a imagen PNG



21. Extracción de texto de la imagen



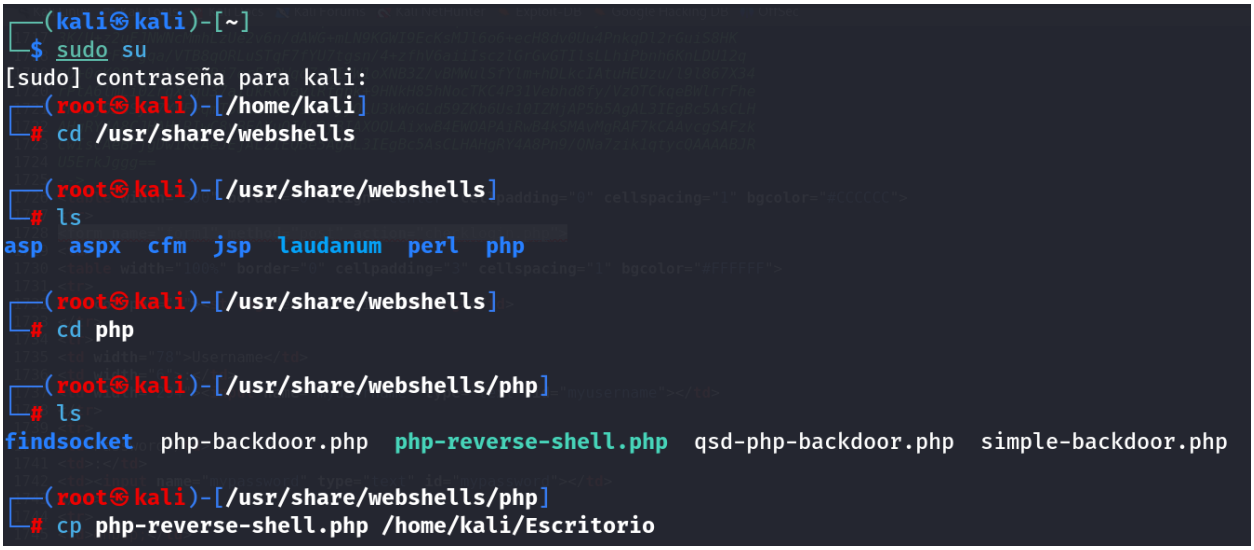
22. Inicio de sesión exitoso

Paso 8: En este paso, desde la terminal de **Kali Linux**, se accedió al directorio de **webshells** ubicado en /usr/share/webshells/php/. Allí se encuentran diferentes *scripts* en PHP diseñados para pruebas de penetración.

El archivo seleccionado fue **php-reverse-shell.php**, el cual permite establecer una conexión inversa hacia la máquina atacante. Este script fue copiado al escritorio del entorno de trabajo con el comando:

cp php-reverse-shell.php /home/kali/Escritorio

Con este procedimiento se prepara el archivo malicioso que posteriormente será subido al portal de administración de **Fristileaks**, aprovechando la funcionalidad de carga de archivos.

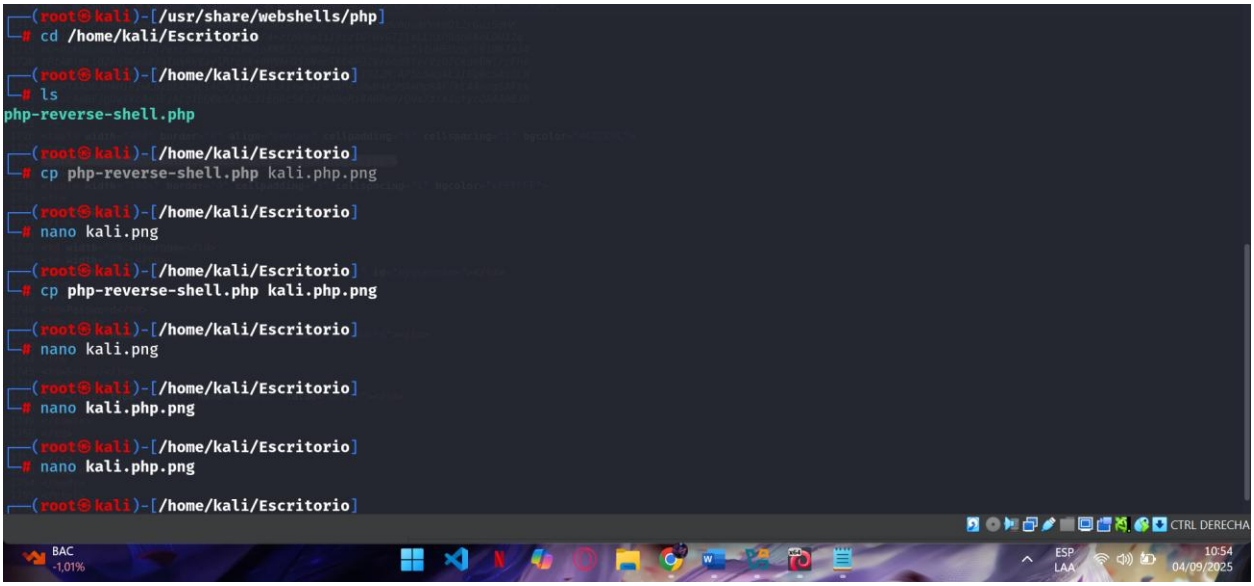


23. Preparación de la reverse shell en PHP

Paso 9: En este paso se trabajó sobre el archivo **php-reverse-shell.php** previamente copiado al escritorio. Para evadir posibles restricciones del portal de subida de archivos de **Fristileaks**, se aplicaron técnicas de **camuflaje de extensión**:

- El archivo se renombró a **kali.php.png**, simulando ser una imagen pero manteniendo código PHP incrustado.
- Se editaron diferentes versiones del archivo utilizando el editor de texto **nano**, asegurando que el *payload* se mantuviera funcional.
- El objetivo de este procedimiento es **subir el archivo disfrazado** a través del formulario del portal de administración, y así ejecutar la *reverse shell* en el servidor.

Este paso es fundamental en el proceso de explotación, ya que permite introducir código malicioso en el sistema de la víctima aprovechando la función de carga de archivos.



24. Modificación y camuflaje de la reverse shell

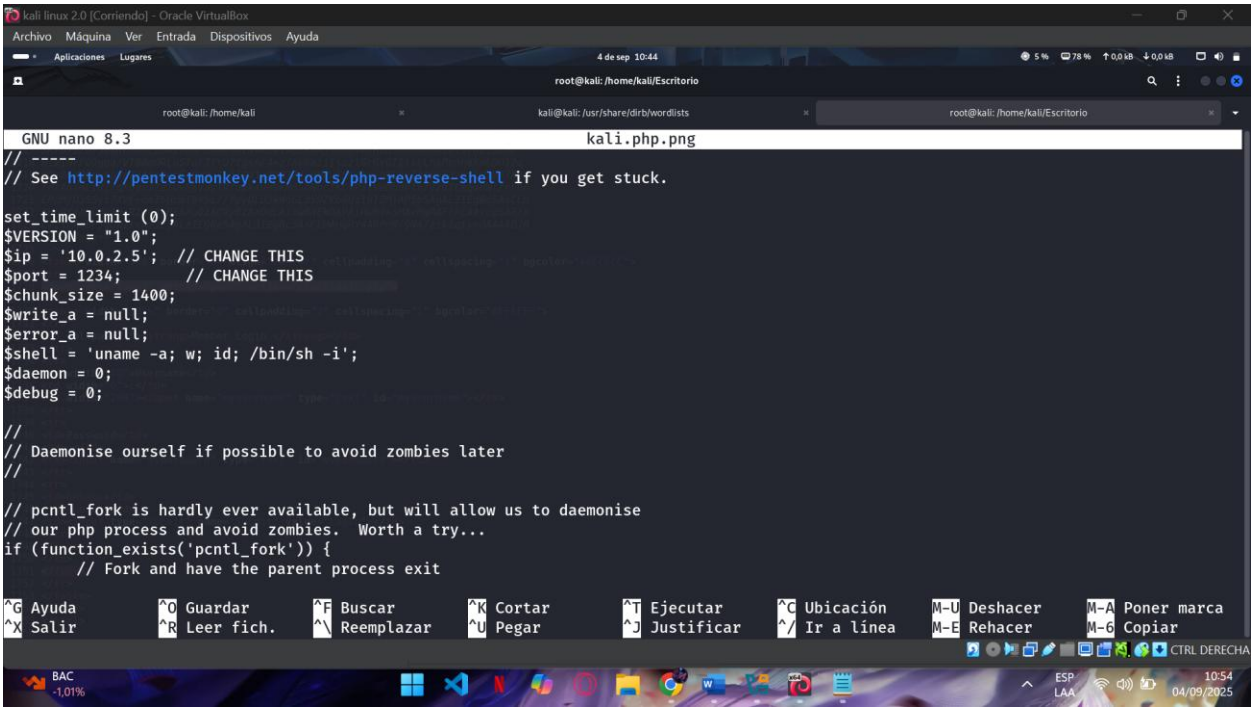
Paso 10: En este paso se editó el archivo **kali.php.png** (la *reverse shell* previamente camuflada) utilizando el editor de texto **nano**.

Dentro del código se realizaron las siguientes configuraciones:

- Se modificó la variable **$ip** para establecer la dirección de la máquina atacante (**10.0.2.5**).
- Se ajustó el valor de **$port** al **1234**, que será utilizado para la conexión inversa.

Estos cambios permiten que, al ejecutar el archivo en el servidor víctima, se establezca una conexión desde el sistema comprometido hacia el equipo atacante, garantizando acceso remoto mediante la shell interactiva.

Este paso es crítico porque prepara el payload para que, al ser subido y ejecutado en el portal de administración, genere una sesión remota que otorgue control sobre el servidor objetivo.



25. Configuración de la reverse shell con dirección IP y puerto

Paso 11**:** En este paso se ejecuta el ataque para establecer la conexión inversa desde el servidor **Fristileaks** hacia la máquina atacante:

1. **Habilitación del listener en Kali Linux:**

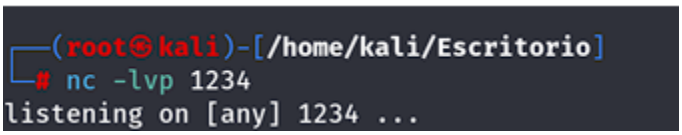
   Se utilizó el comando nc -lvp 1234 para abrir un puerto de escucha en el equipo atacante, esperando la conexión proveniente del servidor comprometido.

2. **Subida y ejecución de la reverse shell:**

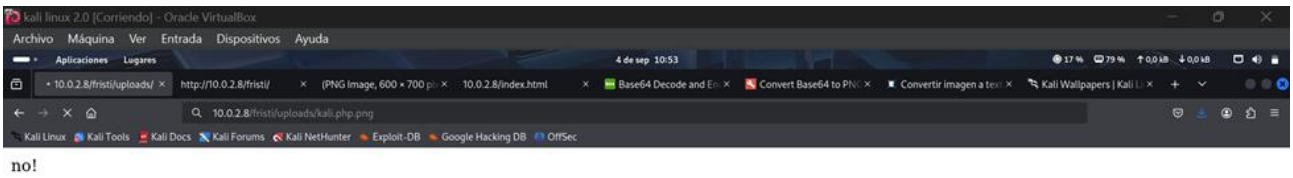
   El archivo camuflado **kali.php.png** fue cargado al portal de administración y ejecutado desde la ruta /fristi/uploads/. ¡En el navegador se muestra el mensaje “no!**”**, lo que indica que el archivo fue procesado.

3. **Conexión establecida con éxito:**

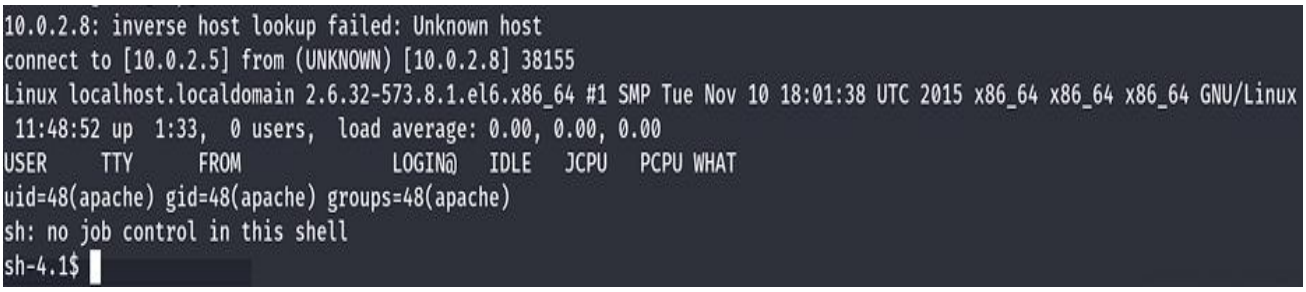
   Finalmente, en la terminal de Kali se confirma la conexión inversa desde el servidor víctima. La sesión obtenida corresponde al usuario del servicio web **apache**, con privilegios limitados.



26. Ejecución de la reverse shel



27. Ejecución del archivo malicioso desde el servidor Fristileaks



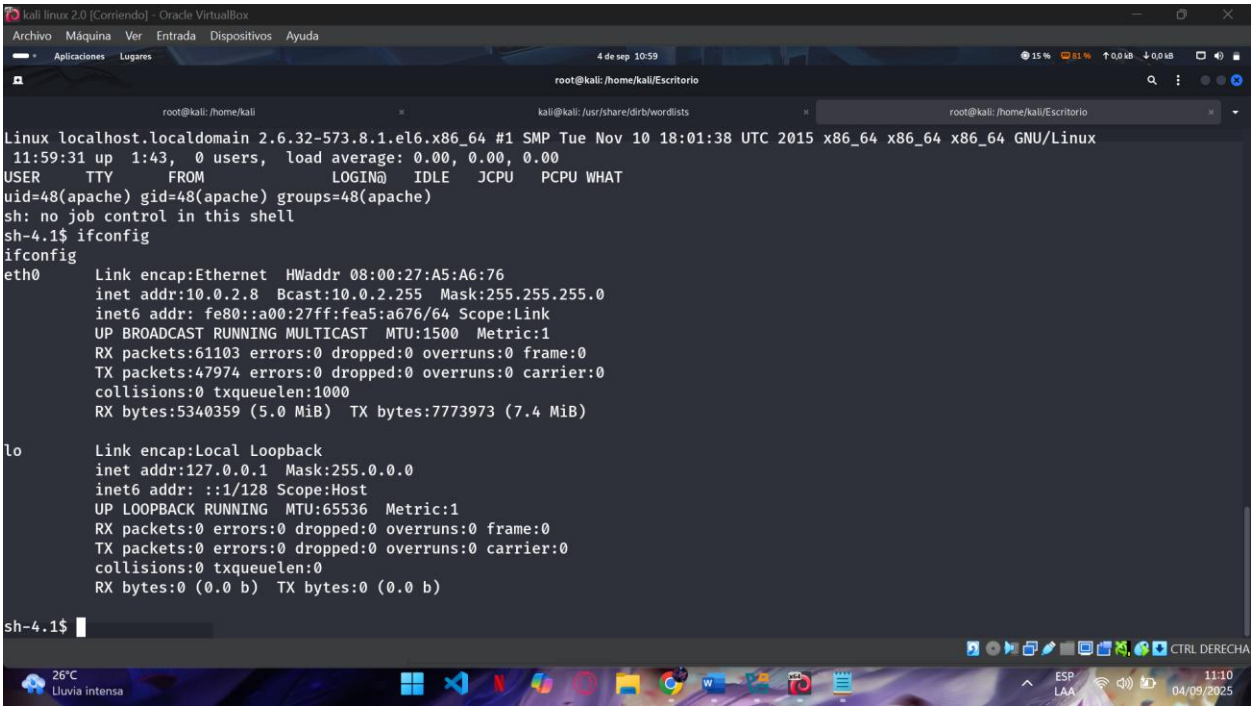
28. Conexión remota establecida con el usuario Apache

Paso 12: En este último paso se realizaron pruebas dentro de la sesión remota obtenida tras ejecutar la *reverse shell*. Se confirma lo siguiente:

- La shell corresponde al usuario **apache**, el cual es el encargado de ejecutar el servicio web.

- Se utilizó el comando ifconfig para visualizar la configuración de red del servidor comprometido, confirmando la dirección **10.0.2.8**.
- La conexión establecida permite ejecutar comandos básicos en el sistema, lo que valida el acceso inicial conseguido mediante la explotación de la vulnerabilidad de carga de archivos.

Con este resultado se da por finalizado el **primer capítulo del laboratorio**, logrando acceso al servidor víctima y quedando preparado el terreno para la siguiente fase: **escalación de privilegios**.



29. Verificación de acceso al sistema comprometido

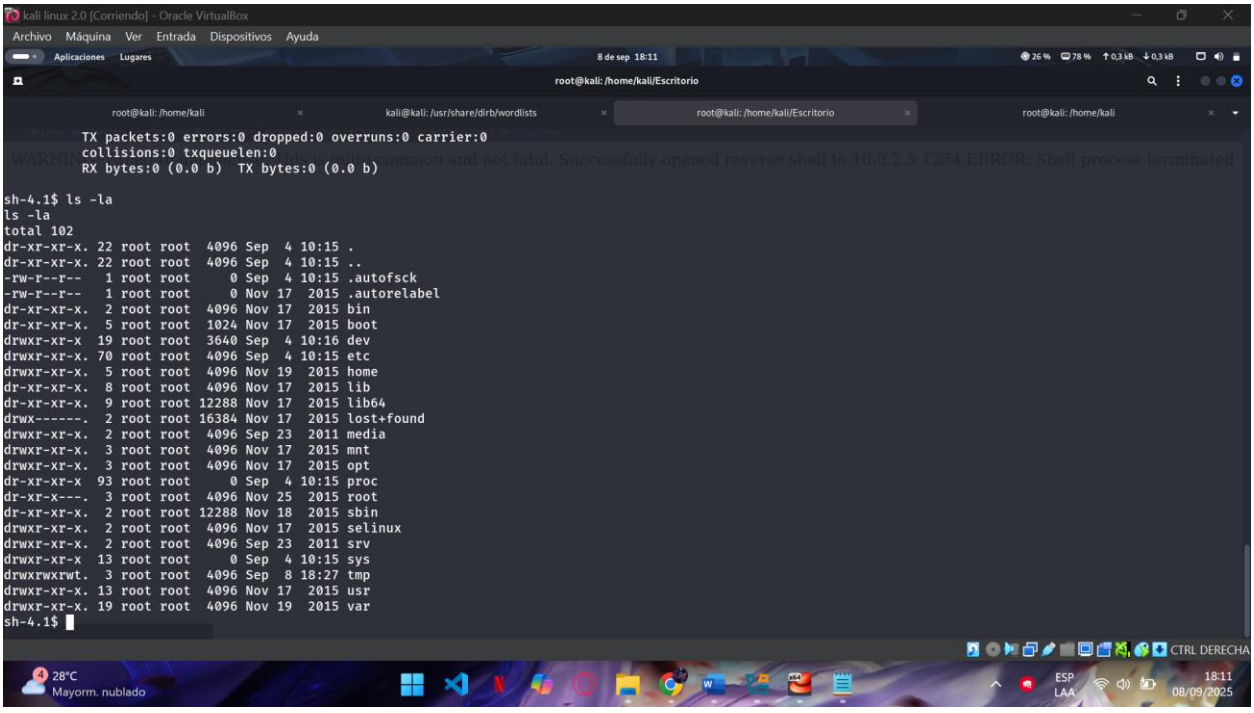
SEGUNDO CAPITULO

Paso 1: En este paso se ejecutó el comando ls -la desde la shell remota obtenida en el capítulo anterior. Esto permitió listar el contenido del directorio raíz (/) del servidor comprometido.

Se observaron directorios clave como:

- **/bin**, **/sbin**, **/usr**, y **/lib**: contienen utilidades y librerías esenciales del sistema.
- **/home**: destinado a los usuarios.
- **/etc**: donde se encuentran configuraciones críticas, incluyendo la gestión de cuentas de usuario.
- **/var**: con archivos de logs y procesos del sistema.

Este reconocimiento inicial es fundamental antes de proceder a la **creación de nuevos usuarios en Fristileaks**, ya que ayuda a identificar rutas y configuraciones que pueden ser aprovechadas para la persistencia dentro del sistema.



30. Exploración inicial del sistema de archivos

Paso 2**:** En este paso se intentó navegar por diferentes directorios del servidor (/var/www, HTML, fristi) con el objetivo de localizar archivos de configuración del sitio web. Aunque varios intentos

no fueron exitosos, se descubrió un archivo clave: **checklogin.php**, asociado al sistema de autenticación de Fristileaks.

Posteriormente, se utilizó un *trick* con Python para mejorar la shell obtenida:

python -c 'import pty; pty.spawn("/bin/bash")'

Con ello se logró una shell más interactiva y funcional.

Luego, se accedió al servicio **MySQL** con el usuario **eezeepz** y se identificó una contraseña válida:

Usuario: eezeepz

Password: 4ll3maal12#

Este hallazgo es crucial, ya que permite el acceso a la base de datos del sistema, donde se pueden gestionar usuarios y extraer información sensible de Fristileaks.

```
sh-4.1$ cd var
cd var
sh-4.1$ cd www
cd www
sh-4.1$ cd HTML
cd HTML
sh: cd: HTML: No such file or directory
sh-4.1$ cd fristi
cd fristi
sh: cd: fristi: No such file or directory
sh-4.1$ cat checklogin.php
cat checklogin.php
cat: checklogin.php: No such file or directory
sh-4.1$ python -c 'import pty;pty.spawn("/bin/bash")'
bash-4.1$ MySQL -u eezeepz -ppython -c 'import pty;pty.spawn("/bin/bash")'
bash-4.1$password: 4ll3maal12#
```

Paso 3: En este paso se utilizó el comando:

mysql -u eezeepz -p

Luego de ingresar la contraseña obtenida anteriormente (**4ll3maal12#**), se accedió de forma exitosa al monitor de **MySQL**.

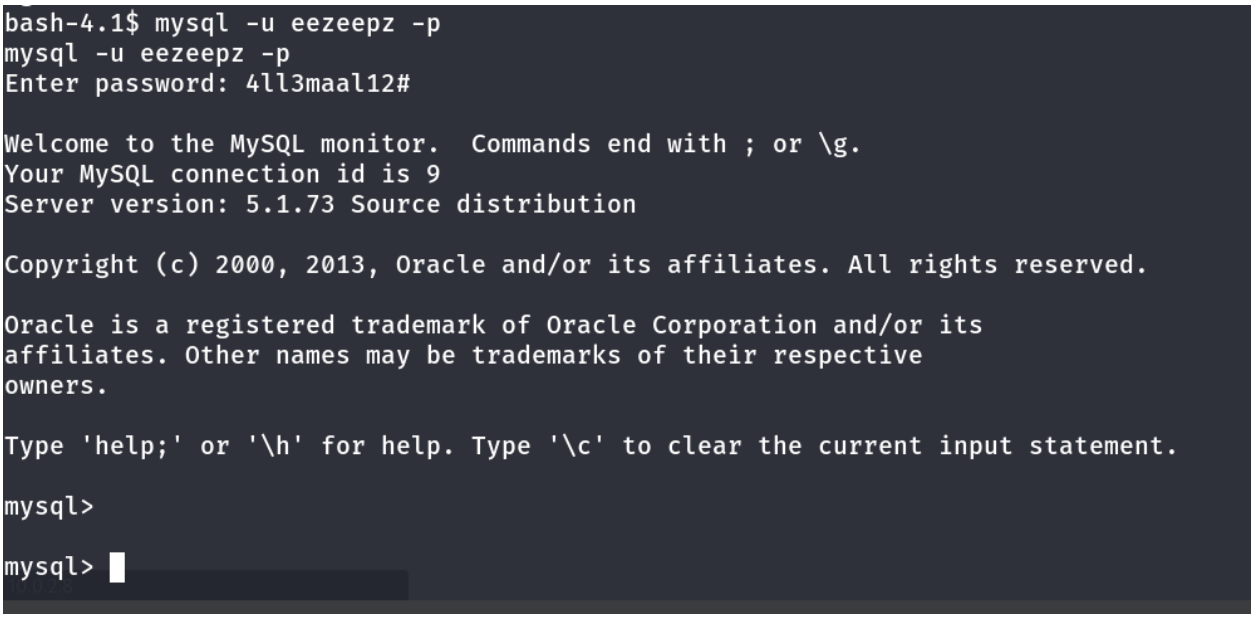
El sistema muestra:

- **Versión del servidor MySQL:** 5.1.73
- **Conexión establecida** con el usuario eezeepz.
- Mensaje de bienvenida a la consola interactiva de MySQL.

Con este acceso ahora es posible:

- Visualizar las bases de datos utilizadas por el sitio **Fristileaks**.
- Revisar las tablas relacionadas con usuarios y credenciales.
- Modificar o crear nuevos usuarios dentro del sistema para obtener persistencia.

Este paso marca la transición hacia la manipulación directa de la base de datos, lo que constituye un punto clave para la **creación de usuarios personalizados** dentro de Fristileaks.



Paso 4: En este paso se trabajó dentro de la base de datos **hackmenow**, en la tabla **members**, donde se gestionan los usuarios del portal **Fristileaks**.

1. Se verificó la estructura de la tabla con el comando:
2. DESCRIBE members;

La tabla contiene tres campos principales: id, username y password.

3. Se consultaron los registros existentes mediante:
4. SELECT * FROM members;

Inicialmente, solo existía el usuario eezeepz con su respectiva contraseña.

5. Se insertaron nuevos usuarios en la base de datos:
6. INSERT INTO members (username, password) VALUES ('edward','user1');
7. INSERT INTO members (username, password) VALUES ('camilo','user2');
8. Finalmente, se verificó la correcta inserción con un nuevo SELECT * FROM members;, confirmando que los usuarios fueron añadidos exitosamente.

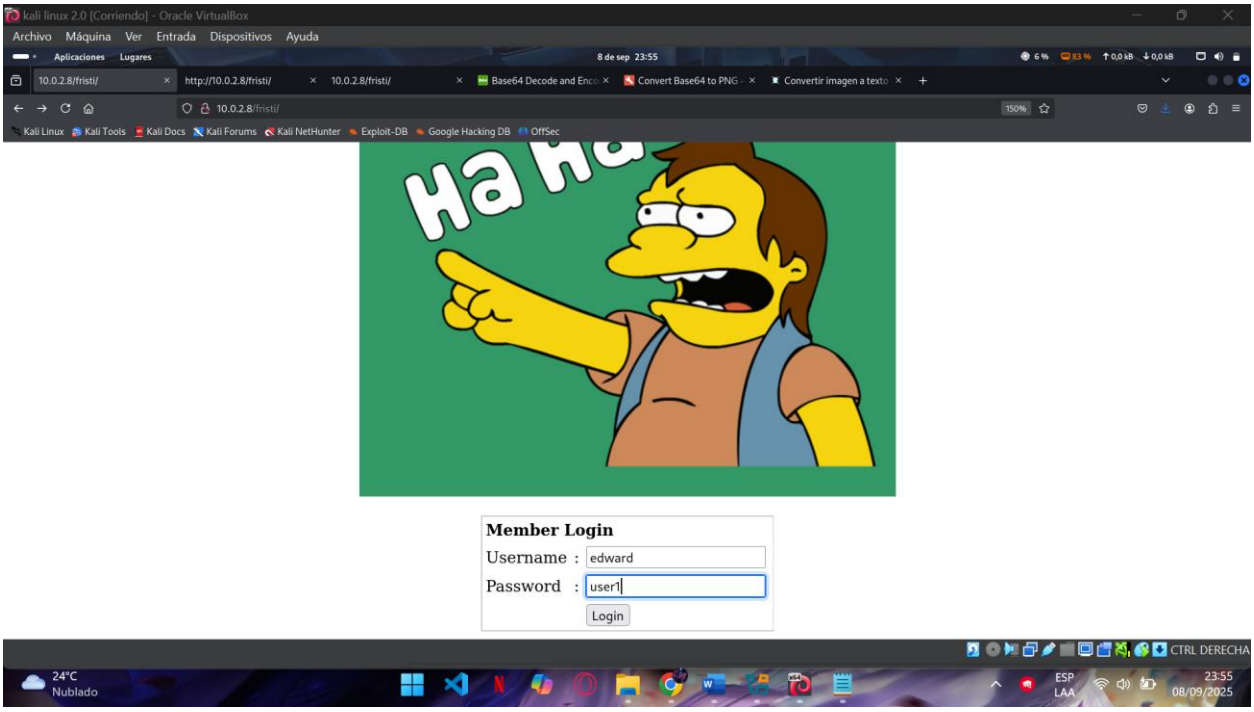
```
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SHOW DATABASES;
SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| hackmenow          |
+--------------------+
2 rows in set (0.01 sec)

mysql> USE hackmenow;
USE hackmenow;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> DESCRIBE members;
DESCRIBE members;
+----------+-------------+------+-----+---------+----------------+
| Field    | Type        | Null | Key | Default | Extra          |
+----------+-------------+------+-----+---------+----------------+
| id       | int(4)      | NO   | PRI | NULL    | auto_increment |
| username | varchar(65) | NO   |     | NULL    |                |
| password | varchar(65) | NO   |     | NULL    |                |
+----------+-------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)
```

31. Exploración de la base de datos hackmenow y estructura de la tabla

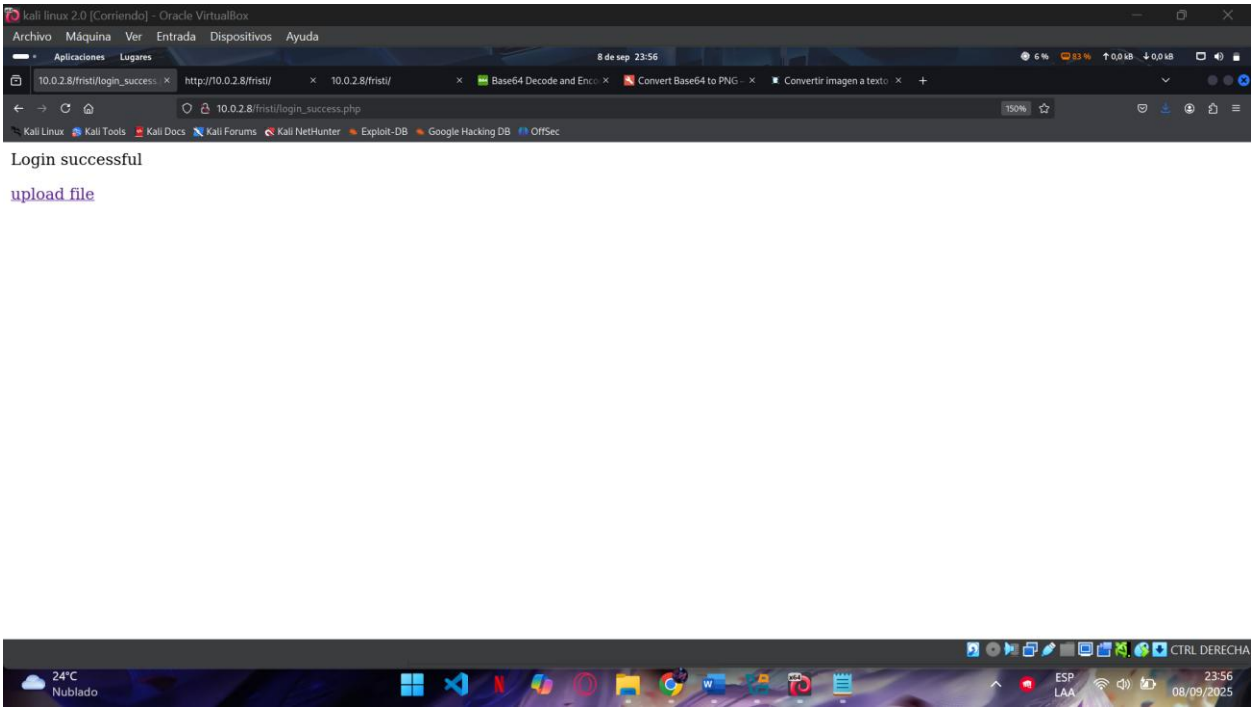
```
mysql> SELECT*FROM members;
SELECT*FROM members;
+----+----------+--------------------+
| id | username | password           |
+----+----------+--------------------+
|  1 | eezeepz  | keKkeKKeKKeKkEkkEk |
+----+----------+--------------------+
1 row in set (0.00 sec)

mysql> INSERT INTO members (username, password) VALUES ('edward','user1');
INSERT INTO members (username, password) VALUES ('edward','user1');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO members (username, password) VALUES ('camilo','user2');
INSERT INTO members (username, password) VALUES ('camilo','user2');
Query OK, 1 row affected (0.00 sec)

mysql> SELECT*FROM members;
SELECT*FROM members;
+----+----------+--------------------+
| id | username | password           |
+----+----------+--------------------+
|  1 | eezeepz  | keKkeKKeKKeKkEkkEk |
|  2 | edward   | user1              |
|  3 | camilo   | user2              |
+----+----------+--------------------+
3 rows in set (0.00 sec)

mysql>
```

32. Inserción y verificación de nuevos usuarios en la tabla

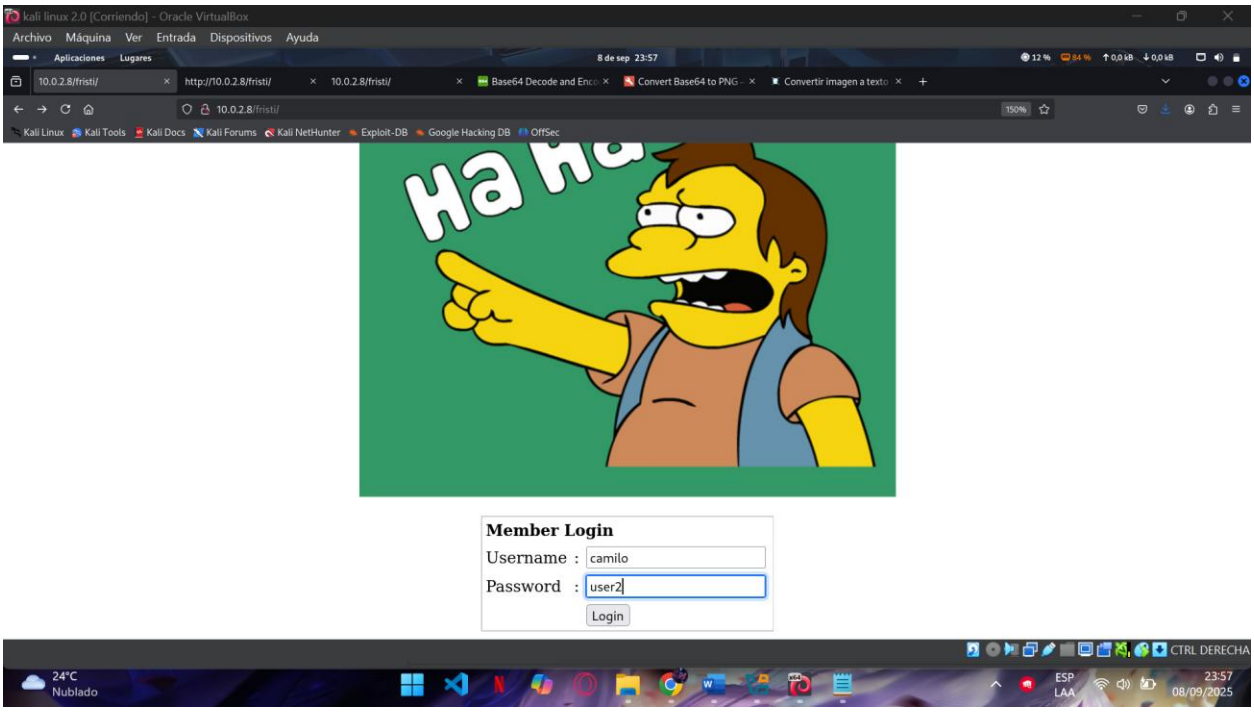
Paso 5: En este paso se procede a **validar el funcionamiento de los usuarios creados en la base de datos**. Primero se accede a la página de inicio de sesión de FristiLeaks, introduciendo las credenciales del usuario **Edward** con la contraseña **user1**, lo que permite un inicio de sesión exitoso y confirma que los datos insertados en la tabla *members* son reconocidos por la aplicación. Posteriormente, se realiza el mismo procedimiento con el usuario **Camilo**, utilizando la contraseña **user2**. Al igual que con Edward, el sistema permite el acceso mostrando el mensaje *Login successful*, lo que evidencia que se han creado y almacenado correctamente los registros en la base de datos.



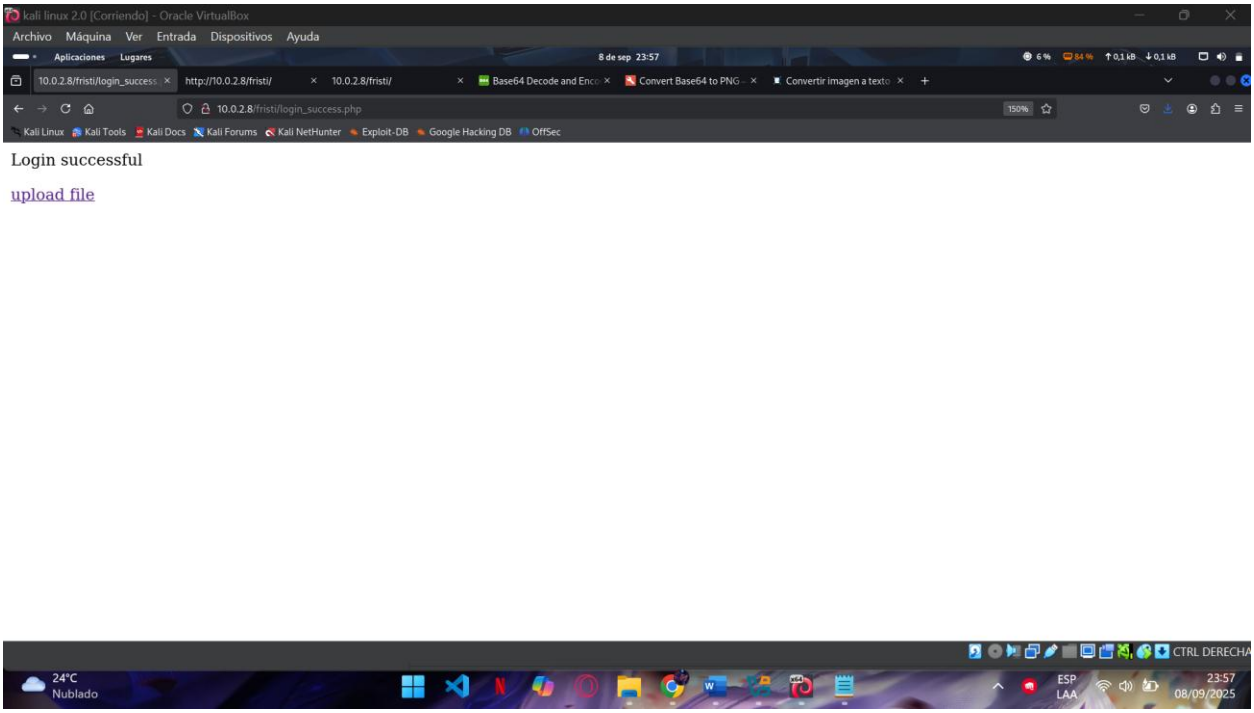
33. Inicio de sesión con el usuario Edward



34. Acceso exitoso con el usuario Edward



35. Inicio de sesión con el usuario Camilo



36. Acceso exitoso con el usuario Camilo