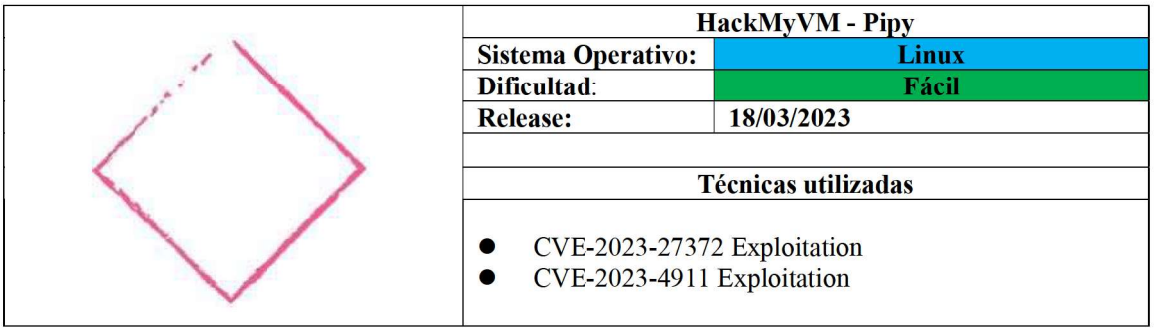


HackMyVM - Pipy	
Sistema Operativo:	**Linux**
Dificultad:	**Fácil**
Release:	**18/03/2023**
Técnicas utilizadas	
● CVE-2023-27372 Exploitation ● CVE-2023-4911 Exploitation	

En este write-up, detallo el proceso de explotación de la máquina Pipy de HackMyMV. La máquina Pipy presenta una serie de desafíos que incluyen la identificación de tecnologías web, la explotación de vulnerabilidades conocidas y la escalada de privilegios. A lo largo de este documento, describo cómo utilicé herramientas como whatweb y Metasploit, para comprometer la máquina objetivo y obtener acceso como usuario root.

Enumeración

Para comenzar la enumeración de la red, utilicé el comando arp-scan -I eth1 --localnet para identificar todos los hosts disponibles en mi red.

```
┌──(root㉿kali)-[/home/administrador/Descargas]
└─# arp-scan -I eth1 --localnet
Interface: eth1, type: EN10MB, MAC: 08:00:27:14:85:97, IPv4: 192.168.1.100
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.1.12    08:00:27:d9:e6:7c       PCS Systemtechnik GmbH

1 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 2.033 seconds (125.92 hosts/sec). 1 responded

┌──(root㉿kali)-[/home/administrador/Descargas]
└─#
```

La dirección MAC que utilizan las máquinas de VirtualBox comienza por "08", así que, filtré los resultados utilizando una combinación del comando grep para filtrar las líneas que contienen "08", sed para seleccionar la segunda línea, y awk para extraer y formatear la dirección IP.

```
┌──(root㉿kali)-[/home/administrador/Descargas]
└─# arp-scan -I eth1 --localnet | grep "08" | sed '2q;d' | awk {'print $1'}
192.168.1.12

┌──(root㉿kali)-[/home/administrador/Descargas]
└─#
```

Una vez que identificada la dirección IP de la máquina objetivo, utilicé el comando **nmap -p- -sS -sC -sV --min-rate 5000 -vvv -Pn 192.168.1.12 -oN scanner_pipy** 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 scripts 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 scripts 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.

```
┌──(administrador㉿kali)-[~/Descargas]
└─$ cat nmap/scanner_pipy
# Nmap 7.94SVN scan initiated Sun Dec 29 01:26:37 2024 as: /usr/lib/nmap/nmap -p- -sS -sC -sV --min-rate 5000 -vvv -Pn -oN nmap/scanner_pipy 192.168.1.12
Increasing send delay for 192.168.1.12 from 0 to 5 due to 19592 out of 65305 dropped probes since last increase.
Nmap scan report for 192.168.1.12
Host is up, received arp-response (0.0015s latency).
Scanned at 2024-12-29 01:26:51 CET for 24s
Not shown: 65533 closed tcp ports (reset)
PORT   STATE SERVICE REASON         VERSION
22/tcp open  ssh     syn-ack ttl 64 OpenSSH 8.9p1 Ubuntu 3ubuntu0.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 c0:f6:a1:6a:53:72:be:8d:c2:34:11:e7:e4:9c:94:75 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBAET5qd1B2Jb+SCBzpF6ifTWw8caKOxMvQkuyEG3M1FHRtyS6a3FyHVBAu4z3ydO6sOlycKooksFn2dDArYbQgc=
|   256 32:1c:f5:df:16:c7:c1:99:2c:d6:26:93:5a:43:57:59 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIIuX5gvUlkO+6tI5fylPXGgBt8+Zc8hJXQ974OtHkGYb
80/tcp open  http    syn-ack ttl 64 Apache httpd 2.4.52 ((Ubuntu))
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-title: Mi sitio SPIP
|_http-server-header: Apache/2.4.52 (Ubuntu)
|_http-generator: SPIP 4.2.0
MAC Address: 08:00:27:D9:E6:7C (Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sun Dec 29 01:27:15 2024 -- 1 IP address (1 host up) scanned in 38.03 seconds
```

Análisis del puerto 80 (HTTP)

Al acceder a la página web disponible en el servidor, encontré una página sencilla que se había creado usando el gestor de contenido SPIP. Sin embargo, no conocía la versión exacta.

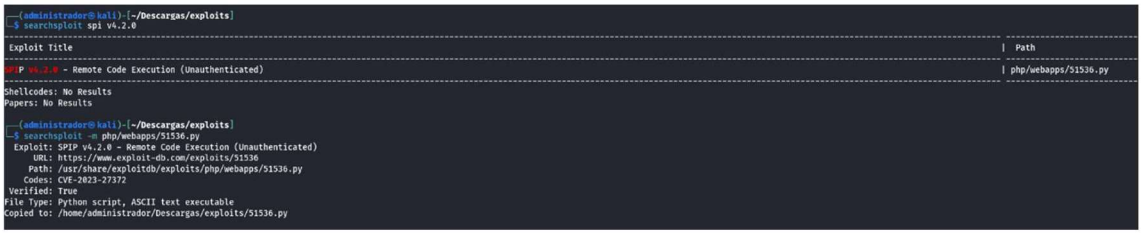
Para averiguar la tecnología usada en esta página web y su versión, usé whatweb. En este caso, se trata del gestor de contenido SPIP v4.2.0. Esta versión es conocida por tener una vulnerabilidad conocida como CVE-2023-27372.

SPIP (Sistema de Publicación para una Internet Participativa) es un software libre de origen francés, diseñado para la producción de sitios web colaborativos, especialmente orientado a revistas en línea.

La vulnerabilidad CVE-2023-27372 permite la ejecución remota de código a través de valores de formularios en el área pública debido a un manejo incorrecto de la serialización. Esta vulnerabilidad explota dos errores para abusar de una función de sanitización de variables demasiado permisiva, permitiendo la inyección de código PHP a través del parámetro 'oubli' en la función de restablecimiento de contraseñas.

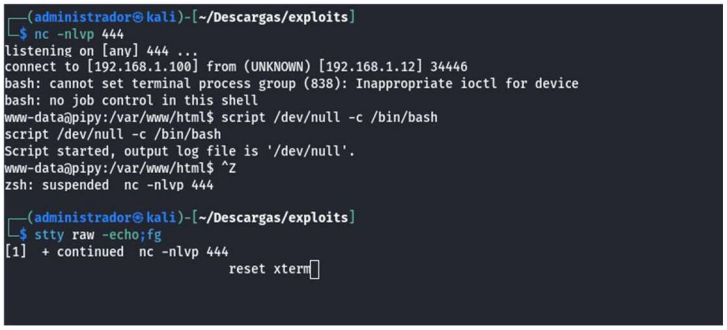
```
┌──(administrador㉿kali)-[~/Descargas]
└─$ whatweb http://192.168.1.12/ -v
WhatWeb report for http://192.168.1.12/
Status    : 200 OK
Title     : Mi sitio SPIP
IP        : 192.168.1.12
Country   : RESERVED, ZZ

Summary   : Apache[2.4.52], HTML5, HTTPServer[Ubuntu Linux][Apache/2.4.52 (Ubuntu)], JQuery, MetaGenerator[SPIP 4.2.0], Script[text/javascript], SPIP[4.2.0][http://192.168.1.12/local/config.txt]

Detected Plugins:
[ Apache ]
        The Apache HTTP Server Project is an effort to develop and
        maintain an open-source HTTP server for modern operating
        systems including UNIX and Windows NT. The goal of this
        project is to provide a secure, efficient and extensible
        server that provides HTTP services in sync with the current
        HTTP standards.

        Version      : 2.4.52 (from HTTP Server Header)
        Google Dorks: (3)
        Website     : http://httpd.apache.org/

[ HTML5 ]
        HTML version 5, detected by the doctype declaration


[ HTTPServer ]
        HTTP server header string. This plugin also attempts to
        identify the operating system from the server header.

        OS           : Ubuntu Linux
        String       : Apache/2.4.52 (Ubuntu) (from server string)

[ JQuery ]
        A fast, concise, JavaScript that simplifies how to traverse
        HTML documents, handle events, perform animations, and add
        AJAX.

        Website     : http://jquery.com/

[ MetaGenerator ]
        This plugin identifies meta generator tags and extracts its
        value.

        String       : SPIP 4.2.0
```

Por tanto, sólo queda encontrar un exploit para la vulnerabilidad encontrada:



Por último, sólo queda explotarla usando un script, que, en este caso, encontré en un repositorio de GitHub:

Si el exploit se ha ejecutado correctamente se obtendría acceso remoto a la máquina objetivo:



Existe una vía alternativa para lograr el mismo resultado. En este caso, es usando Metasploit. Para ello, es necesario crear, en primer lugar, un payload en PHP que permita entablar una conexión inversa con mi máquina de atacante, usando msfvenom:

Ahora sólo queda configurar correctamente el módulo de Metasploit, tal y como se muestra en la imagen siguiente:

Después de ejecutar el exploit correspondiente, se obtendría acceso a la máquina objetivo en una consola de meterpreter.

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.1.100:1234
[*] Meterpreter session 1 opened (192.168.1.100:1234 -> 192.168.1.12:48266) at 2024-12-29 02:46:56 +0100

meterpreter > sysinfo
Computer    : pipy
OS          : Linux pipy 5.15.0-84-generic #93-Ubuntu SMP Tue Sep 5 17:16:10 UTC 2023 x86_64
Meterpreter : php/linux
meterpreter >
```

Más tarde, pude obtener las credenciales de usuario root para realizar una conexión a la base de datos de MySQL en la máquina víctima:

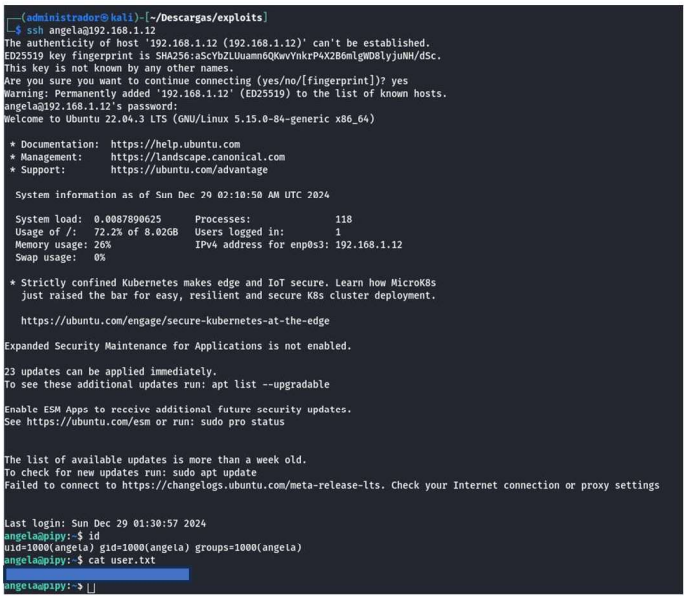
```
www-data@pipy:/var/www/html$ cd config/
www-data@pipy:/var/www/html/config$ ls
chmod.php  cles.php  connect.php  ecran_securite.php  remove.txt
www-data@pipy:/var/www/html/config$ cat connect.php
<?php
if (!defined("_ECRIRE_INC_VERSION")) return;
defined('_MYSQL_SET_SQL_MODE') || define('_MYSQL_SET_SQL_MODE',true);
$GLOBALS['spip_connect_version'] = 0.8;
spip_connect_db('localhost','','root','dbpassword','spip','mysql', 'spip','','');
www-data@pipy:/var/www/html/config$
```

En la tabla spip_auteurs descubrí un posible usuario, Angela, que tenía su contraseña en texto plano y que posiblemente podría utilizar:

```
MariaDB [spip]> desc spip_auteurs
    -> ;
+--------------+--------------+------+-----+---------------------+-----------------------------+
| Field        | Type         | Null | Key | Default             | Extra                       |
+--------------+--------------+------+-----+---------------------+-----------------------------+
| id_auteur    | bigint(21)   | NO   | PRI | NULL                | auto_increment              |
| nom          | text         | NO   |     | ''                  |                             |
| bio          | text         | NO   |     | ''                  |                             |
| email        | tinytext     | NO   |     | ''                  |                             |
| nom_site     | tinytext     | NO   |     | ''                  |                             |
| url_site     | text         | NO   |     | ''                  |                             |
| login        | varchar(255) | YES  | MUL | NULL                |                             |
| pass         | tinytext     | NO   |     | ''                  |                             |
| low_sec      | tinytext     | NO   |     | ''                  |                             |
| statut       | varchar(255) | NO   | MUL | 0                   |                             |
| webmestre    | varchar(3)   | NO   |     | non                 |                             |
| maj          | timestamp    | NO   |     | current_timestamp() | on update current_timestamp() |
| pgp          | text         | NO   |     | ''                  |                             |
| htpass       | tinytext     | NO   |     | ''                  |                             |
| en_ligne     | datetime     | NO   | MUL | 0000-00-00 00:00:00 |                             |
| alea_actuel  | tinytext     | YES  |     | NULL                |                             |
| alea_futur   | tinytext     | YES  |     | NULL                |                             |
| prefs        | text         | YES  |     | NULL                |                             |
| cookie_oubli | tinytext     | YES  |     | NULL                |                             |
| source       | varchar(10)  | NO   |     | spip                |                             |
| lang         | varchar(10)  | NO   |     |                     |                             |
| imessage     | varchar(3)   | NO   |     |                     |                             |
| backup_cles  | mediumtext   | NO   |     | ''                  |                             |
+--------------+--------------+------+-----+---------------------+-----------------------------+
23 rows in set (0.001 sec)

MariaDB [spip]> select nom, pass from spip_auteurs;
+--------+--------------------------------------------------------------+
| nom    | pass                                                         |
+--------+--------------------------------------------------------------+
| Angela | 4ng3l4                                                       |
| admin  | $2y$10$.GR/i2bwnVInUmzdzSi10u66AKUUWGGDBNnA7IuIeZBZVtFMqTsZ2 |
+--------+--------------------------------------------------------------+
2 rows in set (0.000 sec)
```

Análisis del puerto 22 (SSH)

Considerando que posiblemente haya obtenido las credenciales del usuario Angela, decidí usarlas para iniciar sesión en la máquina objetivo mediante SSH:



La vulnerabilidad CVE-2023-4911, también conocida como "Looney Tunables", es un desbordamiento de búfer descubierto en el cargador dinámico ld.so de la biblioteca GNU C mientras se procesaba la variable de entorno GLIBC_TUNABLES. Este problema podría permitir que un atacante local utilice variables de entorno GLIBC_TUNABLES manipuladas con fines malintencionados al iniciar archivos binarios con permiso SUID para ejecutar código con privilegios elevados. La gravedad de esta vulnerabilidad es alta, con una puntuación base CVSS v3.1 de 7.80. Las versiones vulnerables incluyen GNU C Library desde la versión 2.34 hasta la 2.39, excluyendo esta última.

El cargador dinámico es un binario ubicado típicamente en /lib/ld-linux-x86-64.so.2. Este se ejecuta cuando se inicia un programa y es responsable de identificar qué bibliotecas son necesarias para el binario, cargarlas en la memoria del proceso y asegurarse de que cualquier referencia se actualice para apuntar a las direcciones correctas en la memoria.

El cargador dinámico utiliza la variable de entorno GLIBC_TUNABLES para permitir al usuario especificar ciertos valores de configuración "ajustables" cuando se ejecuta el programa. En un sistema con glibc 2.33 o superior, la lista completa de tunables en tu sistema se puede mostrar ejecutando lo siguiente:

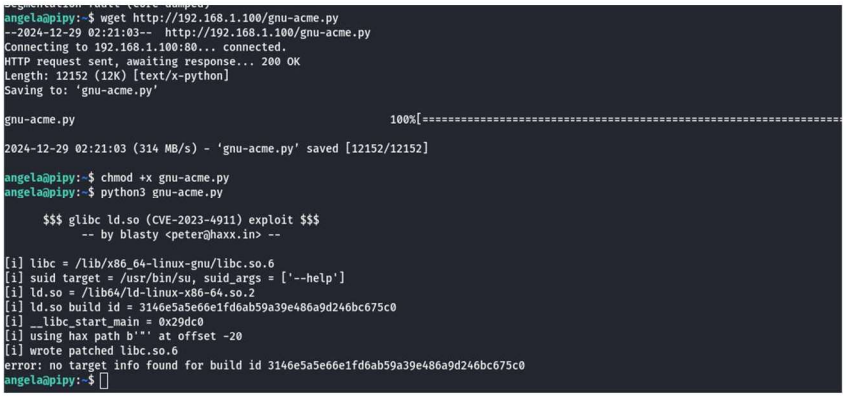
```
/lib64/ld-linux-x86-64.so.2 --list-tunables
```

La vulnerabilidad, en este caso, se explota configurando un entorno maliciosamente diseñado (incluyendo la variable GLIBC_TUNABLES) de manera que cause un desbordamiento de búfer en el cargador dinámico. Este desbordamiento puede ser controlado para obtener ejecución arbitraria y devolver una shell. Si el binario que se está cargando se ejecuta con privilegios de root (como un programa SetUID), entonces la shell resultante también tendrá privilegios de root.

Para comprobar si la máquina objetivo es vulnerable, puede comprobarse ejecutando el comando que puede verse en la siguiente imagen:

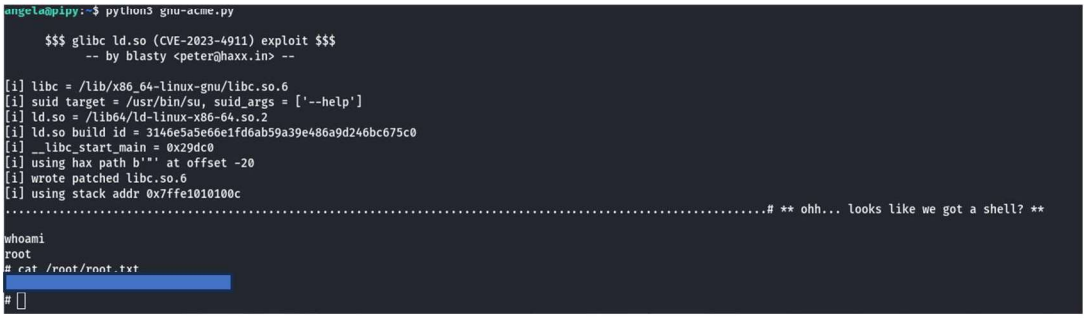
Escalada de privilegios

Teniendo en cuenta la vulnerabilidad encontrada, decidí ejecutar el exploit para elevar mis privilegios en la máquina objetivo. Sin embargo, éste produce un error devolviendo un id:



Esto será necesario incluirlo en el exploit, tal y como puede verse en la imagen siguiente:

Finalmente, accedí al sistema como usuario root, dando por terminado este reto de ciberseguridad:



Bibliografía

https://www.spip.net/es_article78.html
https://es.wikipedia.org/wiki/SPIP
https://github.com/nuts7/CVE-2023-27372
https://nvd.nist.gov/vuln/detail/CVE-2023-27372
https://www.incibe.es/incibe-cert/alerta-temprana/vulnerabilidades/cve-2023-27372
https://www.hackthebox.com/blog/exploiting-the-looney-tunables-vulnerability-cve-2023-4911