

# Metasploit Tutorial for Beginners (Part 1)

February 18, 2015 By Raj Chandel

The Metasploit project is an open-source penetration testing platform that enables you to find and exploit vulnerabilities. In 2003, H.D. Moore created Metasploit as a portable network tool. On October 21, 2009, the Metasploit project was acquired by Rapid7.

The Metasploit project helps security and IT professionals identify security issues, verify vulnerability mitigations, and manage expert-driven security assessments. The Metasploit project includes sub-project like Metasploit Framework and its commercial counterparts: Metasploit Pro, Express, Community, and Nexpose Ultimate.

## Minimum System Requirements:

- 2 GHz+ Processor
- 4 GB RAM (8 GB recommended)
- 1 GB Disk space (50 GB recommended)

## Supported Operating System:

- Windows Server 2008, Server 2012
- Windows 8.1, Windows 10
- Red Hat Enterprise Linux 5.10, 6.5, 7.1 or later
- Ubuntu Linux 14.04 or 16.04 LTS(recommended)

## Required Browser version:

- Google Chrome(latest)
- Mozilla Firefox(latest)
- Microsoft internet explorer 11

## Basic Terms of Metasploit

**Vulnerability:** A vulnerability is a weakness which can be exploited by an attacker to perform unauthorized actions with a computer system. A vulnerability can be as simple as weak passwords or as complex as buffer overflows or SQL injection vulnerabilities.

**Exploit:** Exploit is a piece of code, or a chunk of data, or a sequence of commands that take the advantage of a vulnerability present in a computer system to cause unintended behavior to occur on a computer system such as giving unauthorized access to a system or allowing privilege escalation.

**Payload:** The payload is the part of the private user text which could also contain malware such as worms or viruses which performs the malicious action; deleting data, sending spam or encrypting data.

**Auxiliary:** Auxiliaries are modules present in Metasploit that are used to perform scanning, sniffing, and fuzzing. Auxiliary modules are not useful to give you a shell, but they are extremely useful to brute force passwords or for scanning vulnerabilities.

**Post:** Post modules are used for post exploitation that is used on a compromised target machine to gather evidence or pivot deep within the network.

**Encoders:** Encoder module is used to ensure the payload makes it to the destination.

**Nops:** Nops are used to keep the size of the payload consistent across exploit attempts.

## A cheat sheet of Basic Commands

To start the Metasploit framework we type msfconsole on the terminal. We are greeted by a banner; it spawns a banner every time we start the msfconsole.

```
msfconsole
```

```
root@kali:~# msfconsole ↵

      .:ok000kdc'          'cdk000ko:.
    .x00000000000000c      c0000000000000x.
   :000000000000000k,    ,k000000000000000:
  '000000000k00000:  :000000000000000000'
 o00000000.MMMM. o0000o0000l.MMMM,00000000o
d00000000.MMMMMM.c00000c.MMMMMM,00000000x
l00000000.MMMMMMMMM;d;MMMMMMMMMM,00000000l
.00000000.MMM.;MMMMMMMMMMMM;MMMM,00000000.
c0000000.MMM.00c.MMMMM'o00.MMM,0000000c
o000000.MMM.0000.MMM:0000.MMM,000000o
l00000.MMM.0000.MMM:0000.MMM,00000l
;0000'MMM.0000.MMM:0000.MMM;0000;
.d00o'WM.0000o0ccx0000.MX'x00d.
,k0l'M.0000000000000.M'd0k,
:kk;.0000000000000.;0k:
;k000000000000000k:
,x000000000000x,
.l0000000l.
,d0d,
.

=[ metasploit v4.17.9-dev ]
+ -- --=[ 1806 exploits - 1027 auxiliary - 312 post ]
+ -- --=[ 539 payloads - 42 encoders - 10 nops ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]
```

After starting the Metasploit framework, we can check for the basic commands by using “help” command.

```
msf > help
```

## Core Commands:

- **? / help:** Display the summary of commands that can be used in msfconsole.
- **banner:** Change and display banner in msfconsole.
- **cd:** Change the current working directory.
- **color:** Enable or disable the color output of Metasploit. It has 3 options “true”, “false” and auto.
- **connect:** netcat like function to connect to a host machine build into msfconsole.
- **exit:** Exit the Metasploit console.
- **get:** Gets the value of a context-specific variable
- **getg:** Gets the value of global variable
- **grep:** It matches a given pattern from the output of another msfconsole command
- **history:** Shows command that are previously used in Metasploit
- **irb:** Opens a live ruby interactive shell
- **load:** Loads a Metasploit plugin
- **quit:** Exit the Metasploit console
- **route:** It allows you to route sockets through a session or ‘comm’, providing basic pivoting capabilities
- **save:** This command allows you to save your current environment and settings
- **sessions:** This command allows you to list, interact, and kill spawned sessions
- **set:** This command allows you to configure Framework options and parameters for the current module that is selected on the console.
- **setg:** This command is used to set global variables within msfconsole
- **sleep:** Do nothing for the specified number of seconds
- **spool:** It allows a user to save the output of Metasploit console to a specified file
- **threads:** View and manipulate background threads
- **unload:** unloads a previously loaded plugin and removes any extended commands
- **unset:** It removes a parameter previously configured with set
- **unsetg:** It removes a global variable inside msfconsole
- **version:** Show the framework and console library version numbers

## Module Commands:

- **advanced:** It is used to further fine-tune a module, ‘show advanced’ displays a more advanced option for a module.
- **back:** Once you have finished working with a particular module, or if you inadvertently select the wrong module, you can issue the back command to move out of the current context.

- **info:** It provides detailed information about a particular module including all options, targets, and other information.
- **loadpath:** It loads a third-party module tree for the path.
- **options:** It shows you the available parameters for an exploit.
- **popm:** It pops the pushed module from the top of the module stack.
- **previous:** It sets the previously loaded module as the current module.
- **pushm:** This command pushes the current module on to the stack.
- **reload\_all:** It reloads all modules from all defined module paths.
- **search:** It searches module names and descriptions
- **show:** This command displays modules of a given type, or display all modules.
- **use:** It is used to select a particular module.

msf > help ↩

## 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
exit	Exit the console
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
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
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
-----	-----
advanced	Displays advanced options for one or more modules
back	Move back from the current context
info	Displays information about one or more modules
loadpath	Searches for and loads modules from a path
options	Displays global options or for one or more modules
popm	Pops the latest module off the stack and makes it active
previous	Sets the previously loaded module as the current module
pushm	Pushes the active or list of modules onto the module stack
reload_all	Reloads all modules from all defined module paths
search	Searches module names and descriptions
show	Displays modules of a given type, or all modules
use	Selects a module by name

## Job Commands:

- **handler:** It starts a payload handler in the background.

- **Jobs:** It is used to list jobs running in the background and terminate them.
- **kill:** It kills any running job.
- **rename\_job:** It is used to rename a job

### Resource Script Commands:

- **makerc:** It saves commands entered to a specified rc file.
- **Resource:** It runs all the command stored in the rc file.

### Developer Commands:

- **edit:** This command is used to edit the currently selected module.
- **log:** It displays framework.log starting from the bottom.
- **reload\_lib:** This command is used to reload one or more library files from specified paths.

### Database Backend Commands:

- **db\_connect:** It is used to connect to an existing database.
- **db\_disconnect:** It is used to disconnect from the current database instance.
- **db\_export:** It is used to export a file containing the contents of the database.
- **db\_import:** It is used to import a scan result file.
- **db\_rebuild\_cache:** It is used to rebuild the database-stored module cache.
- **db\_status:** It shows the name of the currently connected database.
- **hosts:** It lists all hosts in the database.
- **loot:** It lists all loot in the database.
- **notes:** It lists all notes in the database.
- **services:** It lists all services in the database.
- **vulns:** It lists all vulnerabilities in the database.
- **workspace:** It helps to switch between database workspaces.

## 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

## Developer Commands

=====

Command	Description
-----	-----
edit	Edit the current module or a file with the preferred editor
irb	Drop into irb scripting mode
log	Displays framework.log starting at the bottom if possible
pry	Open a Pry session on the current module or Framework
reload_lib	Reload one or more library files from specified paths

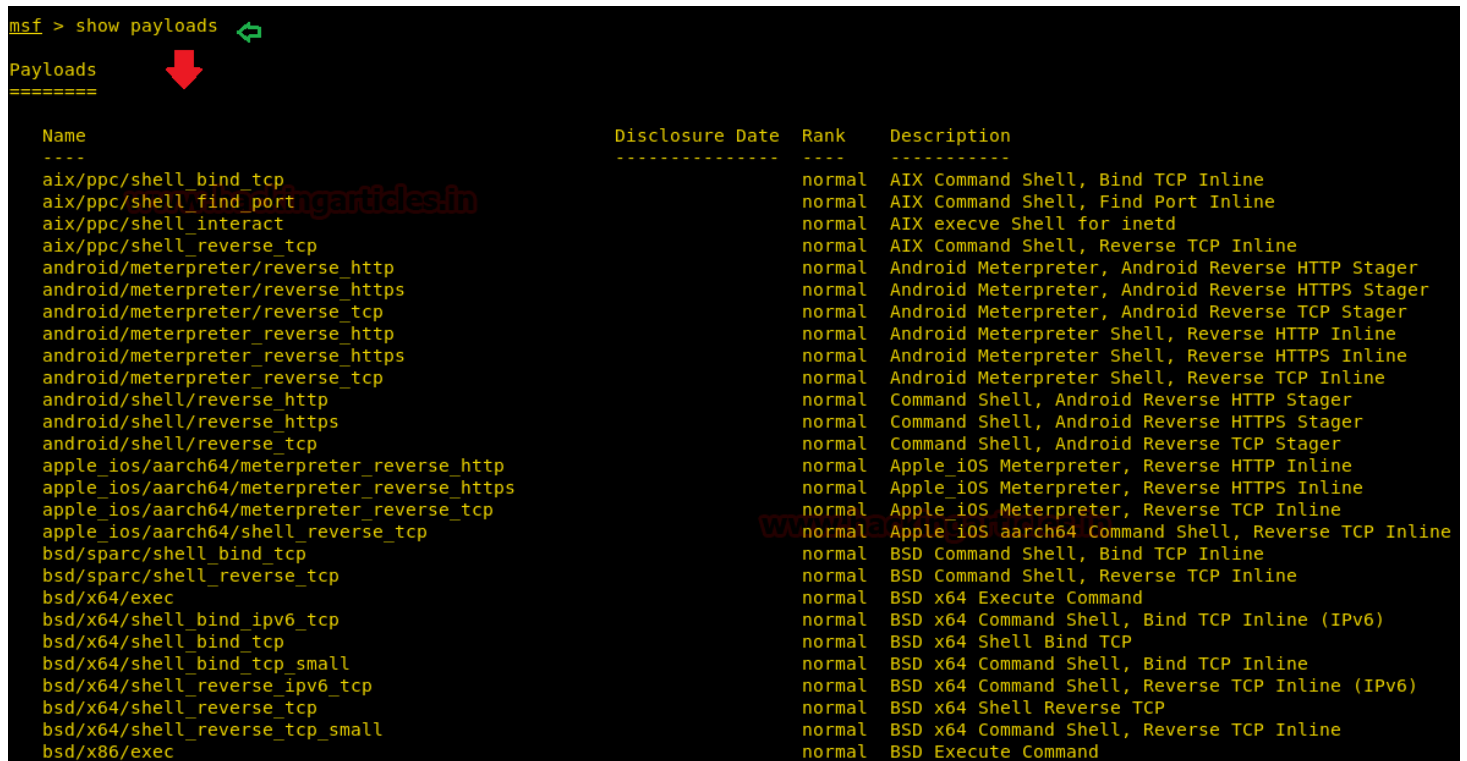
## Database Backend Commands

=====

Command	Description
-----	-----
db_connect	Connect to an existing database
db_disconnect	Disconnect from the current database instance
db_export	Export a file containing the contents of the database
db_import	Import a scan result file (filetype will be auto-detected)
db_nmap	Executes nmap and records the output automatically
db_rebuild_cache	Rebuilds the database-stored module cache
db_status	Show the current database status
hosts	List all hosts in the database
loot	List all loot in the database
notes	List all notes in the database
services	List all services in the database
vulns	List all vulnerabilities in the database
workspace	Switch between database workspaces

To see all the payloads that are available on the Metasploit framework we use command “show payloads”. It lists all the available payloads in alphabetic order.

```
msf > show payloads
```



```
msf > show payloads
```

↓

Name	Disclosure Date	Rank	Description
aix/ppc/shell_bind_tcp		normal	AIX Command Shell, Bind TCP Inline
aix/ppc/shell_find_port		normal	AIX Command Shell, Find Port Inline
aix/ppc/shell_interact		normal	AIX execve Shell for inetd
aix/ppc/shell_reverse_tcp		normal	AIX Command Shell, Reverse TCP Inline
android/meterpreter/reverse_http		normal	Android Meterpreter, Android Reverse HTTP Stager
android/meterpreter/reverse_https		normal	Android Meterpreter, Android Reverse HTTPS Stager
android/meterpreter/reverse_tcp		normal	Android Meterpreter, Android Reverse TCP Stager
android/meterpreter_reverse_http		normal	Android Meterpreter Shell, Reverse HTTP Inline
android/meterpreter_reverse_https		normal	Android Meterpreter Shell, Reverse HTTPS Inline
android/meterpreter_reverse_tcp		normal	Android Meterpreter Shell, Reverse TCP Inline
android/shell/reverse_http		normal	Command Shell, Android Reverse HTTP Stager
android/shell/reverse_https		normal	Command Shell, Android Reverse HTTPS Stager
android/shell/reverse_tcp		normal	Command Shell, Android Reverse TCP Stager
apple_ios/aarch64/meterpreter_reverse_http		normal	Apple iOS Meterpreter, Reverse HTTP Inline
apple_ios/aarch64/meterpreter_reverse_https		normal	Apple iOS Meterpreter, Reverse HTTPS Inline
apple_ios/aarch64/meterpreter_reverse_tcp		normal	Apple iOS Meterpreter, Reverse TCP Inline
apple_ios/aarch64/shell_reverse_tcp		normal	Apple iOS aarch64 Command Shell, Reverse TCP Inline
bsd/sparc/shell_bind_tcp		normal	BSD Command Shell, Bind TCP Inline
bsd/sparc/shell_reverse_tcp		normal	BSD Command Shell, Reverse TCP Inline
bsd/x64/exec		normal	BSD x64 Execute Command
bsd/x64/shell_bind_ipv6_tcp		normal	BSD x64 Command Shell, Bind TCP Inline (IPv6)
bsd/x64/shell_bind_tcp		normal	BSD x64 Shell Bind TCP
bsd/x64/shell_bind_tcp_small		normal	BSD x64 Command Shell, Bind TCP Inline
bsd/x64/shell_reverse_ipv6_tcp		normal	BSD x64 Command Shell, Reverse TCP Inline (IPv6)
bsd/x64/shell_reverse_tcp		normal	BSD x64 Shell Reverse TCP
bsd/x64/shell_reverse_tcp_small		normal	BSD x64 Command Shell, Reverse TCP Inline
bsd/x86/exec		normal	BSD Execute Command

To see all the exploits that are available on the Metasploit framework we use command “show exploits”. It lists all the available payloads in alphabetic order and it also shows the date it was disclosed and the rank of the exploit ranging from “Excellent-average”.

```
msf > show exploits
```



```
msf > show exploits ↩
```

Exploits  
=====

Name	Description	Disclosure Date
aix/local/ibstat_path		2013-09-24
ellent ibstat \$PATH Privilege Escalation		
aix/rpc_cmsd_opcode21		2009-10-07
at AIX Calendar Manager Service Daemon (rpc.cmsd) Opcode 21 Buffer Overflow		
aix/rpc_ttdbserverd_realpath		2009-06-17
at ToolTalk rpc.ttdbserverd _tt_internal_realpath Buffer Overflow (AIX)		
android/adb/adb_server_exec		2016-01-01
ellent Android ADB Debug Server Remote Payload Execution		
android/browser/samsung_knox_smdm_url		2014-11-12
ellent Samsung Galaxy KNOX Android Browser RCE		
android/browser/stagefright_mp4_tx3g_64bit		2015-08-13
mal Android Stagefright MP4 tx3g Integer Overflow		
android/browser/webview_addjavascriptinterface		2012-12-21
ellent Android Browser and WebView addJavaScriptInterface Code Execution		
android/fileformat/adobe_reader_pdf_js_interface		2014-04-13
d Adobe Reader for Android addJavaScriptInterface Exploit		
android/local/futex_requeue		2014-05-03
ellent Android 'Towelroot' Futex Requeue Kernel Exploit		
android/local/put_user_vroot		2013-09-06
ellent Android get_user/put_user Exploit		

To see the list of all the auxiliaries available in Metasploit framework we can use the command “show auxiliary”.

As mentioned earlier, auxiliary modules include scanners, denial of service modules, fuzzers, and more.

```
msf > show auxiliary
```

```
msf > show auxiliary ↩
```

Auxiliary

=====



Name	Disclosure Date	Rank
Description		
-----	-----	----
admin/2wire/xslt_password_reset	2007-08-15	normal
2Wire Cross-Site Request Forgery Password Reset Vulnerability		
admin/android/google_play_store_uxss_xframe_rce		normal
Android Browser RCE Through Google Play Store XFO		
admin/appletv/appletv_display_image		normal
Apple TV Image Remote Control		
admin/appletv/appletv_display_video		normal
Apple TV Video Remote Control		
admin/atg/atg_client		normal
Veeder-Root Automatic Tank Gauge (ATG) Administrative Client		
admin/aws/aws_launch_instances		normal
Launches Hosts in AWS		
admin/backupexec/dump		normal
Veritas Backup Exec Windows Remote File Access		
admin/backupexec/registry		normal
Veritas Backup Exec Server Registry Access		
admin/chromecast/chromecast_reset		normal
Chromecast Factory Reset DoS		
admin/chromecast/chromecast_youtube		normal
Chromecast YouTube Remote Control		
admin/cisco/cisco_asa_extrabacon		normal
Cisco ASA Authentication Bypass (EXTRABACON)		

To see the list of all the post-exploitation modules available in Metasploit framework we can use command “show post”. Post modules are used for post exploitation that is used on a compromised target machine to gather evidence or pivot deep within the network.

```
msf > show post
```

```
msf > show post
```

```
Post
```

```
====
```

Name	Disclosure Date	Rank
tion	-----	----
aix/hashdump		normal
her Dump Password Hashes		
android/capture/screen		normal
Screen Capture		
android/gather/wireless_ap		normal
s wireless SSIDs and PSKs		
android/manage/remove_lock	2013-10-11	normal
Settings Remove Device Locks (4.0-4.3)		
android/manage/remove_lock_root		normal
Root Remove Device Locks (root)		
android/sub_info		normal
s subscriber info from target device		
cisco/gather/enum_cisco		normal
ather Device General Information		
firefox/gather/cookies	2014-03-26	normal
Gather Cookies from Privileged Javascript Shell		
firefox/gather/history	2014-04-11	normal
Gather History from Privileged Javascript Shell		
firefox/gather/passwords	2014-04-11	normal
Gather Passwords from Privileged Javascript Shell		

To see the list of all the encoders available in Metasploit framework we can use the command “show encoder”.

These are used to obfuscate modules to avoid detection by a protection mechanism such as an antivirus or a firewall.

```
msf > show encoders
```

```
msf > show encoders
```

```
Encoders
```

```
=====
```

Name	Disclosure Date	Rank	Description
cmd/brace		low	Bash Brace Expansion Command Encoder
cmd/echo		good	Echo Command Encoder
cmd/generic_sh		manual	Generic Shell Variable Substitution Command Encoder
cmd/ifs		low	Bourne \${IFS} Substitution Command Encoder
cmd/perl		normal	Perl Command Encoder
cmd/powershell_base64		excellent	Powershell Base64 Command Encoder
cmd/printf_php_mq		manual	printf(1) via PHP magic_quotes Utility Command Encoder
generic/eicar		manual	The EICAR Encoder
generic/none		normal	The "none" Encoder
mipsbe/byte_xori		normal	Byte XORi Encoder
mipsbe/longxor		normal	XOR Encoder
mipsle/byte_xori		normal	Byte XORi Encoder
mipsle/longxor		normal	XOR Encoder
php/base64		great	PHP Base64 Encoder
ppc/longxor		normal	PPC LongXOR Encoder
ppc/longxor_tag		normal	PPC LongXOR Encoder
ruby/base64		great	Ruby Base64 Encoder
sparc/longxor_tag		normal	SPARC DWORD XOR Encoder
x64/xor		normal	XOR Encoder
x64/zutto_dekiru		manual	Zutto Dekiru
x86/add_sub		manual	Add/Sub Encoder
x86/alpha_mixed		low	Alpha2 Alphanumeric Mixedcase Encoder
x86/alpha_upper		low	Alpha2 Alphanumeric Uppercase Encoder
x86/avoid_underscore_tolower		manual	Avoid underscore/tolower
x86/avoid_utf8_tolower		manual	Avoid UTF8/tolower
x86/bloxor		manual	BloXor - A Metamorphic Block Based XOR Encoder
x86/bmp_polyglot		manual	BMP Polyglot
x86/call4_dword_xor		normal	Call+4 Dword XOR Encoder
x86/context_cpuid		manual	CPUID-based Context Keyed Payload Encoder
x86/context_stat		manual	stat(2)-based Context Keyed Payload Encoder
x86/context_time		manual	time(2)-based Context Keyed Payload Encoder
x86/countdown		normal	Single-byte XOR Countdown Encoder
x86/fnstenv_mov		normal	Variable-length Fnstenv/mov Dword XOR Encoder
x86/jmp_call_additive		normal	Jump/Call XOR Additive Feedback Encoder
x86/nonalpha		low	Non-Alpha Encoder
x86/nonupper		low	Non-Upper Encoder
x86/opt_sub		manual	Sub Encoder (optimised)
x86/service		manual	Register Service
x86/shikata_ga_nai		excellent	Polymorphic XOR Additive Feedback Encoder
x86/single_static_bit		manual	Single Static Bit
x86/unicode_mixed		manual	Alpha2 Alphanumeric Unicode Mixedcase Encoder
x86/unicode_upper		manual	Alpha2 Alphanumeric Unicode Uppercase Encoder

To see the list of all the nops available in Metasploit framework we can use the command “show nops”. They are used to keep the size of payload consistent across exploit attempts.

```
msf > show nops
```

msf > show nops 

## NOP Generators

=====



Name	Disclosure Date	Rank	Description
----	-----	----	-----
aarch64/simple		normal	Simple
armle/simple		normal	Simple
mipsbe/better		normal	Better
php/generic		normal	PHP Nop Generator
ppc/simple		normal	Simple
sparc/random		normal	SPARC NOP Generator
tty/generic		normal	TTY Nop Generator
x64/simple		normal	Simple
x86/opty2		normal	Opty2
x86/single_byte		normal	Single Byte