

# Linux Command Line Cheat Sheet: All the Commands You Need

## LINUX CHEAT SHEET



You may need to open a compressed file, but you've forgotten the TAR commands. Or you're new to Linux and need to know the top ten terminal commands that open and modify files and folders. The sheer volume of Linux terminal commands can overwhelm beginners, not to mention server administrators, IT professionals, and hobbyists.

Therefore, we've prepared this essential Linux command line cheat sheet to help you get familiar with Linux security commands categorized by the scope of their actions. We're confident that this compilation can help you master Linux quickly.

## Essential Commands

We recommend that you memorize these commands. You'll need them the most when operating Linux.

### File Management

In the following commands: `X` may refer to a single file, a string containing a wildcard symbol referring to a set of multiple files, e.g., `file*.txt`, or the stream output of a piped command (in which case the syntax would be `X | command` instead of `command X`); `Y` is a single directory; `A` and `B` are path strings of files/directories.

COMMAND	DESCRIPTION
<code>*</code>	Wildcard symbol for variable length, e.g., <code>*.txt</code> refers to all files with the TXT extension
<code>?</code>	Wildcard symbol referring to a single character, e.g., <code>Doc?.docx</code> can refer to <code>Doc1.docx</code> , <code>DocA.docx</code> , etc.
<code>ls</code>	List the names of files and subfolders in the current directory. Options include <code>-l</code> , <code>-a</code> , <code>-t</code> which you may combine, e.g., <code>-alt</code> .
<code>ls -l</code>	Also show details of each item displayed, such as user permissions and the time/date when the item was last modified
<code>ls -a</code>	Also display hidden files/folders. May be combined with <code>ls -l</code> to form <code>ls -al</code> .
<code>ls -t</code>	Sort the files/folders according to the last modified time/date, starting with the most recently modified item
<code>ls X</code>	List the files
<code>cd Y</code>	Change directory to <code>Y</code> .  Special instances of <code>Y</code> : <code>.</code> — current directory <code>..</code> — parent directory
<code>cd</code>	To the <code>\$HOME</code> directory
<code>cd ..</code>	Up one level to enclosing folder or parent directory
<code>cd /etc</code>	To the <code>/etc</code> directory

COMMAND	DESCRIPTION
<code>cmp A B</code>	Compare two files A and B for sameness. No output if A and B are identical, outputs character and line number otherwise.
<code>diff A B</code>	Compare two files A and B for differences. Outputs the difference.
<code>pwd</code>	Display the path of the current working directory.
<code>mkdir X</code>	Make a new directory named X inside the current directory.
<code>mv A B</code>	<p>Move a file from path A to path B. Also used for renaming files.</p> <p>Examples:</p> <ul style="list-style-type: none"><li>- Moving between directories <code>folder1</code> and <code>folder2</code>: <code>mv ./folder1/file.txt ./folder2</code> The file name will remain unchanged, and its new path will be <code>./folder2/file.txt</code>.</li><li>- Renaming a file: <code>mv new_doc.txt expenses.txt</code> The new file name is <code>expenses.txt</code>.</li></ul>
<code>cp A B</code>	<p>Copy a file from path A to path B. Usage similar to <code>mv</code> both in moving to a new directory and simultaneously renaming the file in its new location.</p> <p>Example: <code>cp ./f1/file.txt ./f2/expenses.txt</code> simultaneously copies the file <code>file.txt</code> to the new location with a new name <code>expenses.txt</code>.</p>
<code>cp -r Y Z</code>	Recursively copy a directory Y and its contents to Z. If Z exists, copy source Y into it; otherwise, create Z and Y becomes its subdirectory with Y's contents
<code>rm X</code>	Remove (delete) X permanently.
<code>rm -r Y</code>	Recursively delete a directory Y and its contents
<code>rm -f X</code>	Forcibly remove file X without prompts or confirmation
<code>rm -rf Y</code>	Forcibly remove directory Y and its contents recursively
<code>rmdir Y</code>	Remove a directory Y permanently, provided Y is empty.
<code>open X</code>	Open X in its default program.
<code>open -e X</code>	Opens X in the default text editor (macOS: TextEdit)

COMMAND	DESCRIPTION
<code>touch X</code>	Create an empty file <code>X</code> or update the access and modification times of <code>X</code> .
<code>cat X</code>	View contents of <code>X</code> .
<code>cat -b X</code>	Also display line numbers as well.
<code>wc X</code>	Display word count of <code>X</code> .
<code>head X</code>	Display the first 10 lines of <code>X</code> . If more than a single file is specified, each file is preceded by a header consisting of the string " <code>==&gt; X &lt;==</code> " where " <code>X</code> " is the name of the file.
<code>head -n 4 X</code>	Show the first 4 lines of <code>X</code> .
<code>ls *.c   head -n 5</code>	Display the first 5 items of a list of <code>*.c</code> files in the current directory.
<code>tail X</code>	Display the last (10, by default) lines of <code>X</code> . If more than a single file is specified, each file is preceded by a header consisting of the string " <code>==&gt; X &lt;==</code> " where " <code>X</code> " is the name of the file.
<code>tail -n +1 X</code>	Display entire contents of the file(s) <code>X</code> specified, with header of respective file names
<code>tail -f X</code>	Display the last 10 lines of the file(s) <code>X</code> specified, and track changes appended to them at the end. Overwriting <code>X</code> or modifying <code>X</code> with a text editor such as <code>vim</code> would mess up this command's output.
<code>less</code>	Read a file with forward and backward navigation. Often used with pipe, e.g., <code>cat file.txt   less</code>
<code>ln -s A S</code>	Create symbolic link of path <code>A</code> to link name <code>S</code> .

## Input/Output Redirection

These are helpful for logging program output and error messages.

COMMAND	DESCRIPTION
<code>echo TEXT</code>	Display a line of <code>TEXT</code> or the contents of a variable.
<code>echo -e TEXT</code>	Also interprets escape characters in <code>TEXT</code> , e.g., <code>\n</code> → new line, <code>\b</code> → backslash, <code>\t</code> → tab.
<code>echo -n TEXT</code>	Omits trailing newline of <code>TEXT</code> .
<code>cmd1   cmd2</code>	<code> </code> is the pipe character; feeds the output of the command <code>cmd1</code> and sends it to the command <code>cmd2</code> , e.g., <code>ps aux   grep python3</code> .
<code>cmd &gt; file</code>	Redirect output of a command <code>cmd</code> to a file <code>file</code> . Overwrites pre-existing content of <code>file</code> .
<code>cmd &gt;&amp; file</code>	Redirect output of <code>cmd</code> to <code>file</code> . Overwrites pre-existing content of <code>file</code> . Suppresses the output of <code>cmd</code> .
<code>cmd &gt; /dev/null</code>	Suppress the output of <code>cmd</code> .
<code>cmd &gt;&gt; file</code>	Append output of <code>cmd</code> to <code>file</code> .
<code>cmd &lt; file</code>	Read input of <code>cmd</code> from <code>file</code> .
<code>cmd &lt;&lt; delim</code>	<p>Read input of <code>cmd</code> from the standard input with the delimiter character <code>delim</code> to tell the system where to terminate the input. Example for counting the number of lines of ad-hoc input:</p> <pre> wc -l &lt;&lt; EOF &gt; I like &gt; apples &gt; and &gt; oranges. &gt; EOF 4 </pre> <p>Hence there are only 4 lines in the standard input delimited by <code>EOF</code>.</p>
<code>cmd &lt;&lt;&lt;</code>	Input a text <code>string</code> to <code>cmd</code> .

COMMAND	DESCRIPTION
<code>string</code>	
<code>cmd 2&gt; foo</code>	Redirect error messages of <code>cmd</code> to <code>foo</code> .
<code>cmd 2&gt;&gt; foo</code>	Append error messages of <code>cmd</code> to <code>foo</code> .
<code>cmd &amp;&gt; file</code>	Redirect output and error messages of <code>cmd</code> to <code>file</code> .

## Search and Filter

These commands help you find the files you want.

COMMAND	DESCRIPTION
<code>grep patt /path/to/src</code>	Search for a text pattern <code>patt</code> in <code>X</code> . Commonly used with pipe e.g., <code>ps aux   grep python3</code> filters out the processes containing <code>python3</code> from all running processes of all users.
<code>grep -r patt /path/to/src</code>	Search recursively (the target directory <code>/path/to/src</code> and its subdirectories) for a text pattern <code>patt</code> .
<code>grep -v patt X</code>	Return lines in <code>X</code> not matching the specified <code>patt</code> .
<code>grep -l patt X</code>	Write to standard output the names of files containing <code>patt</code> .
<code>grep -i patt X</code>	Perform case-insensitive matching on <code>X</code> . Ignore the case of <code>patt</code> .
<code>find</code>	Find files.
<code>find /path/to/src -name "*.sh"</code>	Find all files in <code>/path/to/src</code> matching the pattern <code>"*.sh"</code> in the file name.
<code>find /home -size +100M</code>	Find all files in the <code>/home</code> directory larger than 100MB.
<code>locate name</code>	Find files and directories by name.
<code>sort X</code>	Arrange lines of text in <code>X</code> alphabetically or numerically.

## Archives

These commands are for unpacking compressed files (.zip, .gz, .bz2, .tar, etc.) with large or complex contents, such as programs.

COMMAND	DESCRIPTION
<code>tar</code>	Manipulate archives with .tar extension.
<code>tar -v</code>	Get verbose output while manipulating TAR archives. May combine this option with others, e.g., <code>tar -tvf</code> .
<code>tar -cf archive.tar Y</code>	Create a TAR archive named <code>archive.tar</code> containing <code>Y</code> .
<code>tar -xf archive.tar</code>	Extract the TAR archive named <code>archive.tar</code> .
<code>tar -tf archive.tar</code>	List contents of the TAR archive named <code>archive.tar</code> .
<code>tar -czf archive.tar.gz Y</code>	Create a gzip-compressed TAR archive named <code>archive.tar.gz</code> containing <code>Y</code> .
<code>tar -xzf archive.tar.gz</code>	Extract the gzip-compressed TAR archive named <code>archive.tar.gz</code> .
<code>tar -cjf archive.tar.bz2 Y</code>	Create a bzip2-compressed TAR archive named <code>archive.tar.bz2</code> containing <code>Y</code> .
<code>tar -xjf archive.tar.bz2</code>	Extract the bzip2-compressed TAR archive named <code>archive.tar.bz2</code> .
<code>gzip</code>	Manipulate archives with .gz extension.
<code>gzip Y</code>	Create a gzip archive named <code>Y.gz</code> containing <code>Y</code> .
<code>gzip -l Y.gz</code>	List contents of gzip archive <code>Y.gz</code> .
<code>gzip -d Y.gz</code> <code>gunzip Y.gz</code>	Extract <code>Y.gz</code> and recover the original file <code>Y</code> .
<code>bzip2</code>	Manipulate archives with .bz2 extension.
<code>bzip2 Y</code>	Create a bzip2 archive named <code>Y.bz2</code> containing <code>Y</code> .

COMMAND	DESCRIPTION
<pre>bzip2 -d Y.gz bunzip2 Y.gz</pre>	Extract <code>Y.bz2</code> and recover the original file <code>Y</code> .
<pre>zip -r Z.zip Y</pre>	Zip <code>Y</code> to the ZIP archive <code>Z.zip</code> .
<pre>unzip Z.zip</pre>	Unzip <code>Z.zip</code> to the current directory.
<pre>unzip Z.zip</pre>	List contents of <code>Z.zip</code> .

## File Transfer

These commands are for logging in to local and remote hosts, and for uploading and downloading files, transferring them between devices. Remember to omit the square brackets "[" and "]" when you input the optional parameters they enclose.

COMMAND	DESCRIPTION
<pre>ssh user@access</pre>	Connect to <code>access</code> as <code>user</code> .
<pre>ssh access</pre>	Connect to <code>access</code> as your local username.
<pre>ssh -p port user@access</pre>	Connect to <code>access</code> as <code>user</code> using <code>port</code> .
<pre>scp [user1@]host1: [path1] [user2@]host2:[path2]</pre>	<p>Login to <code>hostN</code> as <code>userN</code> via secure copy protocol for <code>N=1, 2</code>.</p> <p>Example usage:</p> <pre>scp alice@pi:/home/source bob@arduino:/destination</pre> <p><code>path1</code> and <code>path2</code> may be local or remote, but ensure they're absolute rather than relative paths, e.g., <code>/var/www/*.html</code>, <code>/usr/bin</code>.</p> <p>If <code>user1</code> and <code>user2</code> are not specified, <code>scp</code> will use your local username.</p>





COMMAND	DESCRIPTION
chmod permission file	Change permissions of a file or directory. Permissions may be of the form [u/g/o/a] [+/-/=] [r/w/x] (see examples below) or a three-digit octal number.
chown user2 file	Change the owner of a file to user2.
chgrp group2 file	Change the group of a file to group2.

Usage examples:

- `chmod +x testfile` → allow all users to execute the file
- `chmod u-w testfile` → forbid the current user from writing or changing the file
- `chmod u+wx,g-x,o=rx testfile` → simultaneously add write and execute permissions to user, remove execute permission from group, and set the permissions of other users to only read and write.

## Numeric Representation

The table below compares Linux file permissions in octal form and in the format [u/g/o/a] [+/-/=] [r/w/x].

OCTAL	PERMISSION(S)	EQUIVALENT TO APPLICATION OF
0	No permissions	<code>-rwx</code>
1	Execute permission only	<code>=x</code>
2	Write permission only	<code>=w</code>
3	Write and execute permissions only: $2 + 1 = 3$	<code>=wx</code>
4	Read permission only	<code>=r</code>
5	Read and execute permissions only: $4 + 1 = 5$	<code>=rx</code>
6	Read and write permissions only: $4 + 2 = 6$	<code>=rw</code>

OCTAL	PERMISSION(S)	EQUIVALENT TO APPLICATION OF
7	All permissions: $4 + 2 + 1 = 7$	<code>=rwx</code>

### Examples

- `chmod 777 testfile` → allow all users to execute the file
- `chmod 177 testfile` → restrict current user (u) to execute-only, while the group (g) and other users (o) have read, write and execute permissions
- `chmod 365 testfile` → user (u) gets to write and execute only; group (g), read and write only; others (o), read and execute only.

### System Information

These commands come in handy when you’re developing new applications for Linux or troubleshooting your Linux machine.

### General

These provide information about your Linux machine and perform administrative tasks.


COMMAND	DESCRIPTION
<code>uname</code>	Show the Linux system information.
<code>uname -a</code>	Detailed Linux system information
<code>uname -r</code>	Kernel release information, such as kernel version
<code>uptime</code>	Show how long the system is running and load information.
<code>su</code>	Superuser; use this before a command that requires root access e.g., <code>su</code>
<code>date</code>	Show the current date and time of the machine.
<code>halt</code>	Stop the system immediately.
<code>shutdown</code>	Shut down the system.



COMMAND	DESCRIPTION
<code>dmidecode</code>	Display system hardware components, serial numbers, and BIOS version
<code>hdparm -i /dev/sda</code>	Display information about the disk <code>sda</code>
<code>hdparm -tT /dev/sda</code>	Perform a read speed test on the disk <code>sda</code>
<code>badblocks -s /dev/sda</code>	Test for unreadable blocks on the disk <code>sda</code>

## Disk Usage

These commands provide storage details regarding your Linux machine.

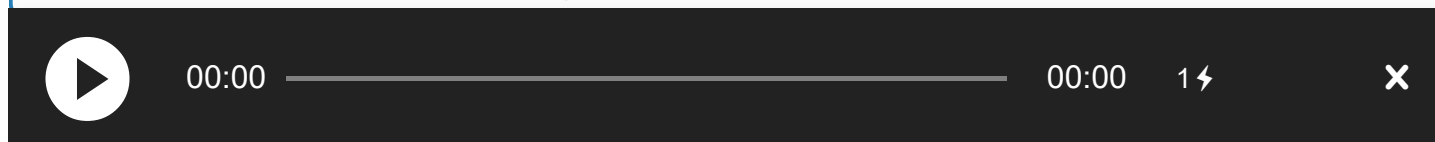
COMMAND	DESCRIPTION
<code>df</code>	Display free disk space.
<code>du</code>	Show file/folder sizes on disk.
<code>du -ah</code>	Disk usage in human readable format (KB, MB etc.)
<code>du -sh</code>	Total disk usage of the current directory
<code>du -h</code>	Free and used space on mounted filesystems
<code>du -i</code>	Free and used inodes on mounted filesystems
<div><div></div><div>00:00</div><div><div></div></div><div>00:00</div><div>1⚡</div><div>✕</div></div>	
<code>free -m</code>	Display free and used memory in MB.
<code>free -g</code>	Display free and used memory in GB.

## Process Management and Performance Monitoring

The following is redolent of functions in Windows Task Manager, but on the command line.

COMMAND	DESCRIPTION
&	Add this character to the end of a command/process to run it in the background
ps	Show process status. Often used with <code>grep</code> e.g., <code>ps aux   grep python3</code> displays information on processes involving <code>python3</code> .  Meaning of <code>aux</code> : <code>a</code> = show processes for all users <code>u</code> = show user or owner column in output <code>x</code> = show processes not attached to a terminal
ps -e ps -A	Either of these two commands prints all running processes in the system
ps -ef	Print detailed overview
ps -U root -u root	Display all processes running under the account <code>root</code> .
ps -eo pid,user,command	Display only the columns <code>pid</code> , <code>user</code> and <code>command</code> in <code>ps</code> output
top	Display sorted information about processes
htop	Display sorted information about processes with visual highlights. It allows you to scroll vertically and horizontally, so you can see every process running on your system and entire commands
<del>kill PID</del>	Kill a process specified by its process ID <code>PID</code> , which you obtain using the <code>ps</code> command
killall proc1	Kill all processes containing <code>proc1</code> in their names

COMMAND	DESCRIPTION
<code>lsuf</code>	List all open files on the system. (This command helps you pinpoint what files and processes are preventing you from successfully ejecting an external drive.)
<code>lsuf -u root</code>	List all files on the system opened by the <code>root</code> user. As the output can be long, you may use <code>lsuf -u root   less</code> to keep this list from taking up space in the terminal output.
<code>mpstat 1</code>	Display processor-related statistics, updated every second (hence the 1, whereas <code>mpstat 2</code> refreshes the output every 2 seconds)
<code>vmstat 1</code>	Display virtual memory statistics (information about memory, system processes, paging, interrupts, block I/O, disk, and CPU scheduling), updated every (1) second
<code>iostat 1</code>	Display system input/output statistics for devices and partitions. virtual memory statistics, updated every (1) second
<code>tail -n 100 /var/log/messages</code>	Display the last 100 lines in the system logs. Replace <code>/var/log/messages</code> with <code>/var/log/syslog</code> for Debian-based systems.
<code>tcpdump -i eth0</code>	Capture and display all packets on interface <code>eth0</code>
<code>tcpdump -i eth0 port 80</code>	Monitor all traffic on interface <code>eth0</code> port 80 (HTTP)
<code>watch df -h</code>	Execute <code>df -h</code> and show periodic updates. To exit, press <b>Ctrl+C</b> .



These commands give information on the system's users and allows superuser administrators to change user settings.

COMMAND	DESCRIPTION
<code>who</code>	Display who is logged in

COMMAND	DESCRIPTION
w	Display what users are online and what they are doing
users	List current users
whoami	Display what user you are logged in as
id	Display the user ID and group IDs of your current user
last	Display the last users who have logged onto the system
groupadd gp1	Create a group named gp1
useradd -c "Alice Bob" -m ab1	Create an account named ab1, with a comment of "Alice Bob" and create the new user's home directory
userdel ab1	Delete the account named ab1
usermod -aG gp1 ab1	Add the account ab1 to the group gp1

## Networking

These commands regulate how your Linux machine communicates with other computers, such as the local area network (LAN) router or external websites.

COMMAND	DESCRIPTION
ifconfig	Display all network interfaces with IP addresses
ifconfig -a	Display all network interfaces, even if any of them is down, with IP addresses
ifconfig eth0	Display IP addresses and details of the eth0 interface
ethtool eth0	<del>Query or control network driver and hardware settings of the interface eth0</del>
netstat	Print open sockets of network connections, routing tables, interface statistics, masquerade connections, and multicast memberships.



COMMAND	DESCRIPTION
	Pipe with the <code>less</code> command: e.g., <code>netstat -a   less</code>
<code>netstat -a</code>	Show both listening and non-listening sockets
<code>netstat -l</code>	Show only listening sockets
<code>netstat -n</code> <code>netstat -tulp</code>	Show listening TCP and UDP ports and corresponding programs
<code>ping host</code>	Send ICMP echo request to <code>host</code> , which may be a symbolic name, domain name or IP address
<code>whois domain</code>	Display whois information for <code>domain</code>
<code>dig domain</code>	Display DNS information for <code>domain</code>
<code>dig -x addr</code>	Do a reverse lookup on an IPv4 or IPv6 address <code>addr</code>
<code>host domain</code>	Display DNS IP address for <code>domain</code>
<code>wget LINK</code>	Download from location <code>LINK</code>
<code>curl LINK</code>	Display the HTML source of <code>LINK</code> . Check out our <a href="#">curl Cheat Sheet</a> for details.

## Installing New Programs

A package file is an archive containing compiled software and other resources it uses. The package file extension and the package installer (a utility for installing new programs) depend on the Linux distribution.



article's [Archives](#) section), `./setup.sh`, and `make install`.

## Package Management Overview

The following table is on package management in popular Linux distributions.

LINUX DISTRIBUTION	PACKAGE FILE EXTENSION	PACKAGE INSTALLER(S)
Debian / Ubuntu	.deb	apt, dpkg
Rocky / Fedora / Red Hat Enterprise Linux	.rpm	yum, dnf
Arch Linux / Manjaro / Garuda / Chakra	.pkg, .pacman, .pkg.tar(.xz/.zst/.gz)	pacman

## Package Management Commands

Here are the commands for package management in popular Linux distributions.

LINUX DISTRIBUTION	DEBIAN / UBUNTU	ROCKY / FEDORA / RED HAT ENTERPRISE LINUX	ARCH LINUX / MANJARO / GARUDA / CHAKRA
Update list of packages available from remote repositories	sudo apt update	dnf check-update	The command <code>pacman -Syy</code> achieves this purpose but may damage your system. Use <code>pacman -Syu</code> instead.
Upgrade installed packages	sudo apt upgrade	sudo dnf upgrade	<code>pacman -Syu</code>
Find a package with keyword in the name	apt search keyword	dnf search keyword	<code>pacman -Ss keyword</code>
Summary information about a package	<code>package</code>	<code>package</code>	<code>pacman -Si package</code>
Install a package (with appropriate file extension) on the local file system	sudo dpkg -i package.deb	sudo dnf install package.rpm	<code>pacman -S package</code>

LINUX DISTRIBUTION	DEBIAN / UBUNTU	ROCKY / FEDORA / RED HAT ENTERPRISE LINUX	ARCH LINUX / MANJARO / GARUDA / CHAKRA
Remove / uninstall a package	<code>sudo apt remove package</code>	<code>sudo dnf erase package</code>	<code>pacman -R package</code>



00:00

00:00

1 



