## Keyboard Shortcuts

| Shortcut | What It Does | Notes |
|---|---|---|
| `cmd` Tab Tab | Show all commands that start with "cmd" | |
| Esc + . | Insert last argument entered | Repeat to go back to previous arguments |
| Up Arrow | Auto-fill last command entered | Repeat to go back to previous commands |
| Ctrl+Arrow (Right/Left) | Move to beginning/end of a word | |
| Ctrl+Shift+T | Open new terminal tab | |
| Ctrl+L | Clear terminal output | |
| Ctrl+D | Logout / Exit | |

## Important Directories & Files

| Directory/File | (Stands For) Purpose |
|---|---|
| /root | Root user's home directory |
| /home | Normal users' home directories |
| /var | (Variables) Database locations, spool files for mail, etc. |
| | |
| /usr | (User System Resources) System resources used by users |
| — /usr/bin | (Binaries) Executables usable by normal users |
| — /usr/sbin | (System Binaries) Executables only usable by root user |
| — /usr/lib | 52-bit libraries |
| — /usr/lib64 | 64-bit libraries |
| | |
| /etc | (Extended Text Configurations) Config files |
| — /etc/passwd | User account config files |
| — /etc/group | Group membership info & config files |
| — /etc/shadow | User password info & config files |
| — /etc/sudoers | Main file for sudo config |
| — /etc/sudoers.d/ | Destination for sudo config "dropping files" |
| — /etc/login.defs | Defines default properties used for new user accounts |

## vim Editor Controls

| | --- Command Mode --- | |
|---|---|---|
| o | Starts new line **and** switches to Insert mode | |
| cw | "Change word" - deletes word **and** switches to Insert mode | |
| G | Go to end of file | |
| dd | Delete **or cut** entire line | |
| yy | Copy entire line | |
| p | Paste copied text | |
| x | Delete character cursor is on top of | |
| dw | Delete word cursor is on top of | |
| d$ | Delete from cursor to end of line | |
| ZZ | Save & quit (existing files) | |
| | --- Extended Command Mode --- | |
| :w *filename* | Save (new files) | |
| :wq | Save & quit (existing files) | |

## Users & Groups

| Command | What It Does | |
|---|---|---|
| `id` *user1* | Show UID, GID, & secondary groups (current user if not specified) | |
| `getent passwd` *usr1* | Find out if usr1 is known to the system | |
| `useradd` *user1* | Add a new user | |
| `-u` *#* | set a specific UID | |
| | `-s /sbin/nologin` | create a user without a login shell |
| `usermod` *user1* | Change properties of existing user | |
| `-c "`*text*`"` | adds text to comment field | |
| `-g` *group1* | changes **primary group** to group1 | |

## Users & Groups (cont)

| | |
|---|---|
| `-G` *group1* | **replaces** supplementary group with group1 |
| `-aG` *group1* | **appends** group1 to supplementary groups |
| | locks user1's account (instead of deleting) |
| `newgrp` *group1* | Change current user's primary group (**temporary**; current session only) |
| `userdel` *user1* | Deletes user1 but **not** their home directory |
| `-r` | deletes both user **and** home directory |
| `groups` *user1* | Displays simple list of groups user1 is a member of |
| `passwd` *user1* | Set password for user1 |
| `chage` *user1* | Change password aging properties for user |
| `-m` *days* | minimum # of days between password changes |
| `-M` *days* | maximum # of days between password changes |
| `-W` *days* | warning period before password expires |
| `-I` *days* | inactivity period (password usable after expiration) |

## Users & Groups (cont)

| | |
|---|---|
| `-d 0` | require password change on next login |
| `-E` *date* | date when **account** expires |
| `getent group` *grp1* | Find out if grp1 is known to the system |
| `groupadd` *group1* | Create a new group |
| `-g #` | set a specific GID |
| `-r` | create a **system** group |
| `groupmod` *group1* | Change properties of existing group |
| `-g #` | change GID to specified number |

To set the number of days from today when user's **account** expires:
`chage -E $(date +`*days* `%Y-%m-%d)`

To give full admin privileges to a user or group:
`echo "[`*user1*`|%`*group1*`] ALL=(ALL) ALL" >> /etc/s udo ers.d/`*name*

## Managing Processes

| | |
|---|---|
| `w` | Displays currently logged-in users, login method, time, & resource usage info |
| `--from` | |
| `-u` *username* | |
| *command* `&` | Starts a new job in the background |

## Managing Processes (cont)

| | |
|---|---|
| `jobs` | Displays jobs running in the background<br>("+" = default job) |
| `fg %1` | Brings job 1 to the foreground |
| `bg %1` | Sends job 1 to the background |
| `pidof` `command` | Returns the PID(s) of a currently running job |
| `systemctl status` `cmd` | Shows main PID of a process, among other things<br>(can use if `pidof` returns multiple) |
| --- CPU Load Handling --- ||
| `uptime` | Displays the load average for the last 1, 5, and 15 mins |
| `lscpu` | Displays number of CPUs in the system, among other things |
| `-p` | |
| --- Viewing & Managing Processes --- ||
| `ps` | Shows info about processes; pipe to `head`/`grep`/etc for less output<br>("[`output`]" = kernel thread) |
| `aux` | |
| `lax` | |

## Managing Processes (cont)

| | |
|---|---|
| `-ef` | |
| `--forest` | shows output in visual "tree" format |
| `pstree` | Shows processes in visual tree format<br>*(may not always be installed)* |
| `-p` | includes PIDs in tree output |
| | |
| `top` | Linux's in-terminal equivalent of Task Manager<br>(Also shows load average)<br>Keyboard Controls: |
| `k` | send a signal to a process from within `top` |
| Shift+M | |
| Shift+P | |
| l / t / m | toggle "load", "threads", & "memory" header lines |
| `kill` `PID` | Sends signal 15 to specified process<br>* *Must use process ID* * |
| `-signal` | |
| `-l` | displays list of signal options |
| `killall` `process` | Kills processes by name instead of PID |

## Managing Processes (cont)

| | |
|---|---|
| `pkill` | Look up & manage processes by name or other attributes (default is signal 15) |
| `-signal` | specify signal to send |
| `-u` *username* | sends signal to user's account (forces logout & shuts down their processes) |
| `-t` *pts/22* | kills specified terminal session (TTY) |
| | |
| `nice -n` *x* *command* | Starts a **new** process with an adjusted priority value (*x*) |
| `renice -n` *x* `-p` *PID* | Change the priority value (*x*) of an **existing** process |

Process Priority Values: **-20** to **19** (default = **0**)
(**negative** = **higher** priority | **positive** = **lower** priority)

## SSH

| | |
|---|---|
| `ssh` *user@ip.addr* | Start ssh connection |
| `-v` | verbose; shows in detail what's happening while establishing connection |
| `-Y` | enables graphical application support |
| `-p` *port#* | |

## SSH (cont)

| | |
|---|---|
| `-i` *keyfile* | |
| `-o` *option dest_server* | |

Example:
PreferredAuthentications=password

| | |
|---|---|
| `ssh-keygen` | Generates a public/private key pair |
| `-N '' -f` *filename* | |

| | |
|---|---|
| `ssh-co py-id` *user@s erv er* | Installs public key on destination server |
| `-i` *filepath* | |

## Wildcards & Shortcuts

| Wildcard | What It Represents | |
|---|---|---|
| ~ | Current user's home directory | |
| . | The directory you're currently in | |
| .. | Parent directory of your current directory | |
| * | All files in current directory | `ls i*` = all files starting with "i" |
| ? | "Any character" | `f??e` matches anything with 4 characters that starts with "f" & ends with "e" |
| [aou] | Match to any **one** of the enclosed characters | |
| !! | "Last command run" (inserts into current command) | |
| ^x^y | Rerun last command but replace x with y | `ls /usr` followed by `^usr^tmp` will run `ls /tmp` |
| !3 | | |
| !ls | Runs most recent command from history that had "ls" in it | |

## Command Redirection

| < | Gets input from somewhere other than stdin (keyboard) |
|---|---|
| > | Sends output somewhere other than stdout (monitor) |
| 1> | Alternative to ">" (not really used) |
| >> | Redirects stdout; **appends** it to existing content (instead of overwriting it) |
| 2> | Redirects only error messages (stderr) |
| &> | Redirects both stdout and stderr |
| cmd1 \| cmd2 | Pipes cmd1 into cmd2 (uses the **output** of cmd1 as the **input** for cmd2) |

## Basic File/Directory Commands

| `ls dir1` | List contents of dir1 |
|---|---|
| -l | additional info (owner, perms, etc.) |
| -a | lists all, including hidden files |
| -i | adds file inode to info displayed |
| -d | info for directory itself (rather than its contents) |
| `mkdir dir1` | Create a new directory |
| -p | creates full path (multiple directories if needed) |
| -v | verbose (output shows action taken) |
| `cp dir1 dir2` | Copy dir1 into dir2 |
| -r | recursively ("and its contents") |
| | |
| `rm dir1` | Removes (deletes) an empty directory |
| -r | recursively ("and it's contents"), including special files |
| `mv file1 dir1` | Move file1 into dir1 |
| -v | |
| | |
| `ln target link` | Creates a hard link from file "link" to file "target" |

### Basic File/Directory Commands (cont)

| | |
|---|---|
| `-s` | makes a symbolic link instead of a hard link |

### Ownership & Permissions

| | |
|---|---|
| `chown` `user:group` | Change owning user **and** owning group |
| `chown` `:group` | Change only owning group |
| `chgrp` `group file` | Alternate way to change owning group |
| | |
| `chmod` `[perms] file` | Change file/directory permissions |
| `-R` | |
| `g+r` | grants owning **g**roup the **r**ead permission |
| `a-x` | removes e**x**ecute perms for **a**ll parties (without changing other perms) |
| `-R a=rwX` | recursively set perms for **a**ll parties to **rw**, with **x** perms on directory **only** |
| `770` | octally set rwx perms for owning user & group, 0 for other |
| `chmod` `[perms] file` | (cont'd) Change special permissions |

### Ownership & Permissions (cont)

| | |
|---|---|
| `o+t` `dir` | |
| `1`*`xxx`* | set Sticky bit – octal form (*xxx* = regular perms) |
| `g+s` `dir/file` | set **S**etGID bit (in **g**roup) **\* For directories and files \*** |
| `2`*`xxx`* | |
| `3`*`xxx`* | set **both** Sticky & SetGID bits (chmod "1"+"2") |
| `u+s` `file` | |
| `4`*`xxx`* | |
| | |
| `umask` | Displays currently set umask (when used by itself) |
| `077` | removes default perms (0 from user, 7 from group/other) |
| `0077` | preserves any special perms when removing defaults |

Octal Expression Values: **r=4** | **w=2** | **x=1**

Default permissions (on vanilla Linux):
**directories=777** | **files=666**

By River L. (Tamaranth)
cheatography.com/tamaranth/

Not published yet.
Last updated 24th June, 2025.
Page 6 of 9.

## Process States & Signals

| | | |
|---|---|---|
| Running | `R` | currently using CPU |
| Runnable | `R` | |
| Sleeping | `S` | |
| `I` (idle) | | not counted when calculating load |
| `D` | | |
| `K` | | *(not yet addressed)* |
| | | |
| Stopped | `T` | defined (has PID) but prevented from getting CPU cycles |

| --- Signals --- | | |
|---|---|---|
| `1) SIGHUP` | Hang Up | toss current config file in memory & reload it from the file system |
| `9) SIGKILL` | Kill it dead! | |
| `15) SIGTERM` | Terminate | shut down process (default used by `kill`) |
| `18) SIGCONT` | Continue ("unstop") | resume allowing process to get CPU time |
| `19) SIGSTOP` | Stop | suspend process & prevent from getting CPU time |

## Managing Services & Daemons

| --- System Commands --- | |
|---|---|
| `systemctl reboot` | reboots the system |
| `systemctl daemon -reload` | tells systemd you've made changes to a unit file & it needs to reinitialize those unit files |
| `systemctl get-de fault` | shows which target is the default target that the system summons whenever it starts up |
| `systemctl set-de fault` *`graphi cal.targ et`* | sets default target to "graphical.target" |

## Managing Services & Daemons (cont)

| --- Unit Commands --- | | |
|---|---|---|
| `systemctl list-units` | | shows all the active & loaded units on the system |
| | `-t` *`type`* | filters output by specified type of unit |
| | `-a` | |
| `systemctl status` *`unit`* | | shows status & details of specified unit (including location of unit files) |
| `systemctl is-enabled` *`unit.s ervice`* | | shows whether the service is enabled (i.e. will start automatically on system boot) |
| `systemctl is-active` *`unit.s ervice`* | | shows whether the service is active |
| `systemctl enable --now` *`unit.s ervice`* | | enables the service & start it immediately (sets "enable" persistently) |
| `systemctl disable --now` *`unit`* | | stops service immediately & sets to "disabled" persistently |

| | |
|---|---|
| `systemctl reload` *unit* | sends the "hang up" signal; drops config file loaded in memory & reload it from file system |
| `systemctl restart` *unit* | kills the process & starts it up again fresh (main PID will change) |
| `systemctl reload -or -re start` *unit* | reloads if possible, otherwise restart (for when you don't know if unit supports reload) |
| `systemctl list-dependencies` *unit.service* | shows list of units required by *unit.service* for it to work |

## Managing Services & Daemons (cont)

| | |
|---|---|
| `--reverse` | shows units that call *unit.service* |
| `systemctl mask` *unit* | prevents unit from being started automatically **or** manually (not even via `enable --now` command) |
| `systemctl unmask` *unit* | undoes masking on a service |

## Network Info Commands

| | |
|---|---|
| `ip -br addr` | |
| `ip link show` | link-level properties of all interfaces (MAC addresses, etc.) |
| `ip address show` | equivalent of `ipconfig` (same info as `link show`, plus IP addresses) |
| `ip a s` | abbreviated form of `ip address show` |
| `ip route show` | shows default route |
| `ip r s` | |
| `ping` \| `ping6` | IPv4 ping \| IPv6 ping |
| `tracepath` \| `tracepath6` | equivalent of `tracert`/`traceroute` \| IPv6 version |
| `mtr` *address* | "My traceroute" program; gives extra info & continuous running stats |
| `nmap -sS` *host* | scans ports on host; shows port number/type, open state, associated service, etc. |
| `ss` | **s**ockets **s**tate information |
| `ss -p` | **p**rocesses responsible for opening ports |
| `ss -l` | **l**istening sockets |
| `ss -u` | **U**DP sockets |
| `ss -n` | translates names to **n**umbers (e.g. both process name & PID) |
| `ss -t` | **T**CP sockets |

## Misc. Commands

| | |
|---|---|
| `whatis` *command* | Displays a short, one-line summary of what the command does |
| `echo` | Creates output (equivalent of `print` in Python) |
| `date` | Displaying, setting, calculating... basically all things date-related |
| `+%F` | display current date in international format (YYYY-mm-dd) |
| `+%R` | display current time (24-hr clock) |
| `+%s` | display # of seconds since epoch |
| | |
| `tmux` | Splits terminal window into multiple separate panes *(I think)* |
| `gnome- cha racte rs` | Opens window of gnome icons/emotes |
| `gnome- cal culat or` | Opens built-in calculator (like Windows) |
| `sleep` *seconds* | Waits & does nothing for specified time |

By **River L.** (Tamaranth)
cheatography.com/tamaranth/

Not published yet.
Last updated 24th June, 2025.
Page 8 of 9.

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com