

# Commands

Most frequently used CLI (Terminal) commands on Linux

- Commands used for System Information
- Commands used for File Handling

# Commands used for System Information





SHUTDOWN

The command “shutdown” is used to shut down the system.

Note: The shutdown command needs superuser privileges. Hence, you should either be root or run the command with sudo.

Using the command with no flags will schedule a shutdown 1 minute from execution.

```
sudo shutdown
```

Use the following to IMMEDIATELY shutdown your system.

```
sudo shutdown now
```

You can schedule a shutdown in future by providing the time argument either in +t format or in hh:mm format. For example, if you want to shutdown the system after 15 minutes, you can use this command:

```
sudo shutdown +15
```

If you want to shutdown the system at 6 PM in the afternoon, you can use it in the following manner:

```
sudo shutdown 18:00
```

Cancel a shutdown

```
sudo shutdown -c
```

Reboot a system

```
sudo shutdown -r
```

```
sudo reboot
```

```
$ sudo shutdown
Shutdown scheduled for Thu 2023-03-02 20:12:13 EST, use
'shutdown -c' to cancel.
```



# Commands used for File Handling





LS

The command “ls” displays the list of all directories, folder, and files present in the current directory.

LS - LTR

The above-mentioned command displays the name of directories, folders, files with their respective owner name, group’s name, and rights your user has over these.

```
ls
ls -ltr
```

```
/$ ls
bin  dev  lib  libx32  mnt
root snap  sys  usr
boot etc  lib32 lost+found
opt  run  srv   tim  var
cdrom home lib64 media
proc sbin swapfile tmp
```

```
/$ ls -ltr
total 2097256
drwxr-xr-x  2 root root    4096
Feb  9  2021 mnt
drwxr-xr-x 15 root root
4096 Feb  9  2021 var
drwx-----  2 root root   16384
Feb 11  2022 lost+found
lrwxrwxrwx  1 root root      8
Feb 11  2022 sbin -> usr/sbin
lrwxrwxrwx  1 root root    10
Feb 11  2022 libx32 ->
usr/libx32
lrwxrwxrwx  1 root root      9
Feb 11  2022 lib64 -> usr/lib64
lrwxrwxrwx  1 root root      9
Feb 11  2022 lib32 -> usr/lib32
lrwxrwxrwx  1 root root      7
Feb 11  2022 lib -> usr/lib
lrwxrwxrwx  1 root root      7
Feb 11  2022 bin -> usr/bin
drwxrwxr-x  2 root root
4096 Feb 11  2022 cdrom
drwx----- 10 root root    4096
Feb 12  2022 tim
-rw-----  1 root root
2147487744 Aug  9  2022
swapfile
drwxr-xr-x  5 root root    4096
Aug 14  2022 home
drwxr-xr-x 14 root root
4096 Oct 22 14:08 usr
drwxrwxrwx  7 root root
4096 Nov 26 04:18 media
drwxrwxrwx 13 root root
```

### MKDIR

The command “mkdir” allows users to create directories/folders in the system. The user running this command must have suitable rights over the parent directory to create a directory or they will receive an error. Syntax: *mkdir New\_Directory's\_Name*

```
mkdir NewDirectory
```

```
~$ mkdir poopoo
```

```
~$
```

```
~$ ls
```

```
Android      Pictures
```

```
ApplImages   poopoo
```

### RMDIR

The command “rmdir” allows users to remove directories/folders from the system. The user running this command must have suitable rights over the parent directory to remove a directory AND the directory must not have any files or sub-directories within it or you will receive an error. Syntax: *rmdir Directory's\_Name*

```
rmdir DirectoryName
```

```
~$ rmdir poopoo
```

```
rmdir: failed to remove
```

```
'poopoo': Directory not empty
```

```
# Could not delete directory
```

```
# "poopoo" because it is not
```

```
# empty
```

```
~$ rm poopoo
```

```
rm: cannot remove 'poopoo': Is  
a directory
```

```
# Could not remove "poopoo"
```

```
# because it is not a file
```

## RM

The command “rm” is used to remove files from a directory.

## RM -RF

Permanently deletes the specified directory and ALL files and sub-directories beneath the specified directory.

Be VERY careful using this command as you can inadvertently delete your whole drive!

```
rm filename
```

```
rm -rf /path/to/dir/name
```

```
# Listing shows poopoo.txt
# file exists under
# directory "poopoo"
~/poopoo$ ls
poopoo.txt
```

```
~/poopoo$ rm poopoo.txt
```

```
# listing now shows
# poopoo.txt has been
# removed (deleted)
# from directory "poopoo"
~/poopoo$ ls
~/poopoo$
```

```
# Directory "poopoo" exists
# in the listing below
~$ ls
Android      Pictures
AppImages    poopoo
Audio        Public
```

```
~$ rm -rf poopoo
~$
# Successfully removed
# "poopoo" directory
# and all its contents
# as can be seen in the
# listing below
```

```
~$ ls
Android      Parkitect
AppImages    Pictures
Audio        Public
```

<p><b>MV</b></p> <p>The command “mv” is used for two purposes</p> <ul style="list-style-type: none"> <li>• To move files or directories from one path to another path in the system.</li> <li>• To rename a file or folder.</li> </ul>	<div data-bbox="580 114 1013 228">mv Source_File_name Destination_File_Name</div> <div data-bbox="580 264 1013 378">mv File_name New_name_for_file</div>	
<p><b>CP</b></p> <p>The command “cp” is used to copy data from a source file to the destination file. Its function is almost like the command “mv”. The only difference is by using the command “cp” the source file is not removed from the directory after its data is moved to the destination file.</p>	<div data-bbox="580 448 1013 562">cp source_file_name destination_file_name</div>	
<p><b>TOUCH</b></p> <p>Creates an empty file at the specified path with the specified name. Useful for creating a blank file you intend to edit with a CLI editor, such as VIM or NANO.</p>	<div data-bbox="580 781 1013 851">touch /path/name/filename.ext</div>	<div data-bbox="1051 786 1484 1097"> ~\$ ls doc.txt ls: cannot access 'doc.txt': No such file or directory ~\$ touch /home/tim/doc.txt ~\$ ls doc.txt doc.txt </div>
<p><b>CAT</b></p> <p>The command “cat” is a reverse of the command “tac”. It is used to display each line of the file starting from the first row and finishing on its last row. This command is more frequently used than “tac”.</p>	<div data-bbox="580 1162 1013 1232">cat file_name</div>	
<p><b>ECHO</b></p> <p>The command “echo” used to display any expression that is passed as an argument.</p>	<div data-bbox="580 1464 1013 1579">echo expression_to_be_displayed</div>	<div data-bbox="1051 1464 1484 1630"> ~/poopoo\$ echo something-poopoo something-poopoo </div>
<p><b>GREP</b></p> <p>The command “grep” is used to search for a text in the specified file/folder.</p>	<div data-bbox="580 1700 1013 1865">grep “expression_to_be_Searched” file_name_to_search_in</div>	

## ZIP

The command “zip” is used to compress one or more files and store them in a new file with .zip extension.

```
zip new_zip_file_name.zip
```

```
~/poopoo$ zip files.zip file1.txt
file2.txt file3.txt
  adding: file1.txt (stored 0%)
  adding: file2.txt (stored 0%)
  adding: file3.txt (stored 0%)
~/poopoo$ ls
file1.txt  file2.txt  file3.txt
files.zip
```

## UNZIP

The command “unzip” is used to decompress a .zip file and extract all the files within to current directory.

```
unzip zip_file_name.zip
```

```
~/poopoo$ unzip files.zip
Archive:  files.zip
replace file1.txt? [y]es, [n]o,
[A]ll, [N]one, [r]ename: A
  extracting: file1.txt
  extracting: file2.txt
  extracting: file3.txt
```

## SUDO

Sudo stands for SuperUser DO and is used to access restricted files and operations. By default, Linux restricts access to certain parts of the system preventing sensitive files from being compromised.

The `sudo` command temporarily elevates privileges allowing users to complete sensitive tasks without logging in as the root user.

**sudo -i** elevates the user to root for the remainder of the session rather than a command by command basis.

```
sudo some-command
```

```
sudo -i
```

```
# No directory called "peepee"
# exists

$ ls
bin  dev  lib  libx32  mnt
root snap  sys  usr
boot etc  lib32 lost+found
opt  run  srv   tim  var
cdrom home lib64 media
proc sbin swapfile tmp
```

```
# Attempt to make directory
# "peepee" as a normal user
# fails because I'm
# trying to make the
# directory in a path I
# don't have rights to
$ mkdir peepee
mkdir: cannot create directory
'peepee': Permission denied
```

```
# using SUDO to temporarily
# elevate my privileges, I
# can now create the
# directory "peepee"
# in the path as can be seen
# in the listing below:
$ sudo mkdir peepee
[sudo] password for tim:
```

```
$ ls
bin  dev  lib  libx32
mnt  proc sbin swapfile tmp
boot etc  lib32 lost+found
opt  root snap sys   usr
cdrom home lib64 media
peepee run  srv  tim   var
```

```
~/poopoo$ sudo -i
[sudo] password for tim:
~#
```

