Introduction

Linux commands may seem intimidating at rst glance if you are not used to using the terminal. There are many commands for performing operations and processes on your Linux system.

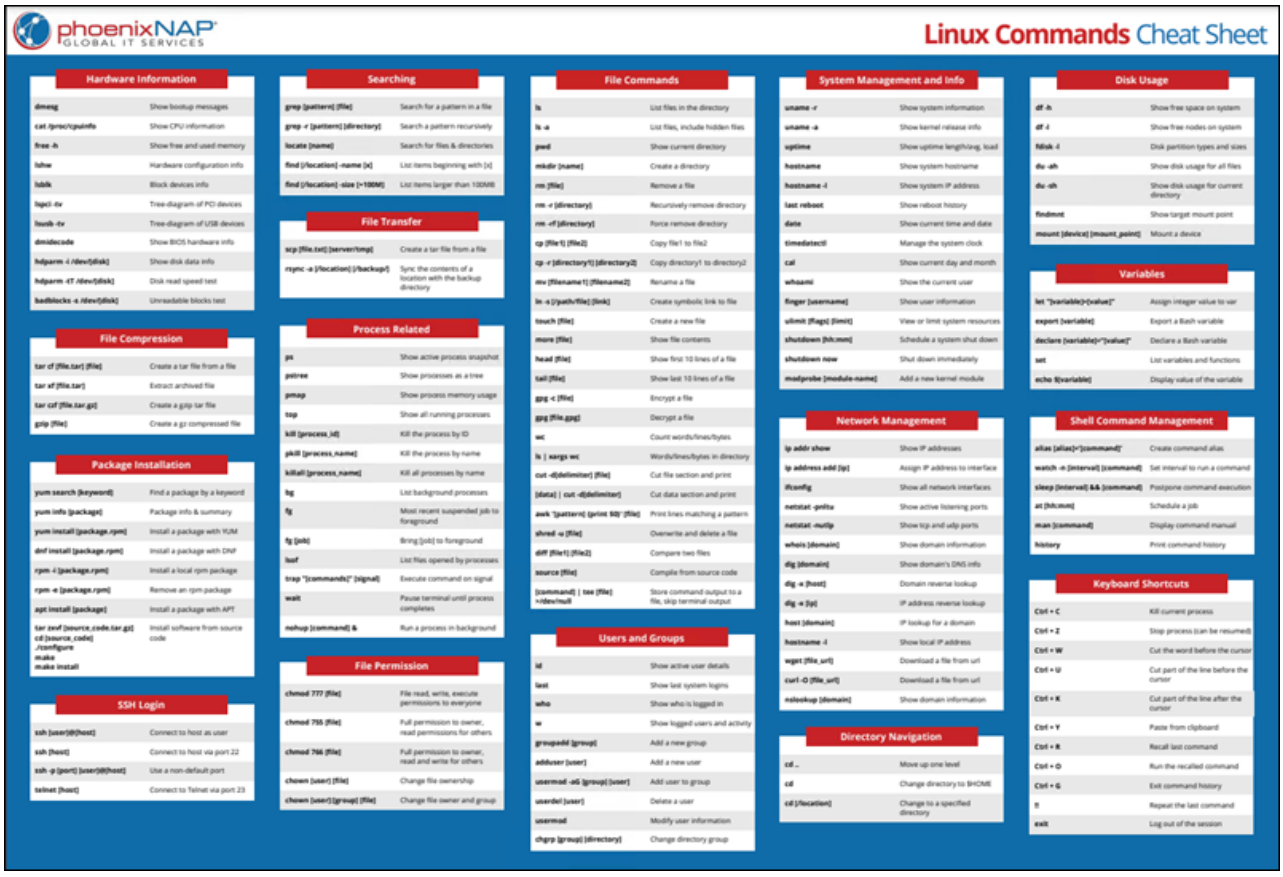
No matter whether you are new to Linux or an experienced user, having a list of common commands close at hand is helpful.

In this tutorial, you will nd commonly used Linux commands as well as a downloadable cheat sheet with syntax and examples.

Important: Depending on your system setup, some of the commands below may require invoking **sudo** to be executed.

Linux Commands Cheat Sheet PDF

If you prefer having all the commands on a one-page reference sheet, we created a helpful **Linux command line cheat sheet**. You can save the **list of linux commands** in PDF format by clicking the **Download Linux Cheat Sheet** button below.



Linux Commands List

The commands found in the downloadable cheat sheet are listed below.

Hardware Information

Show **bootup messages**:

```
dmesg
```

See **CPU information**:

```
cat /proc/cpuinfo
```

Display **free and used memory** with:

```
free -h
```

List **hardware configuration** information:

```
lshw
```

See information about **block devices**:

```
lsblk
```

Show **PCI devices** in a tree-like diagram:

```
lspci -tv
```

Display **USB devices** in a tree-like diagram:

```
lsusb -tv
```

Show **hardware information** from the BIOS:

```
dmidecode
```

Display **disk data** information:

```
hdparm -i /dev/disk
```

Conduct a **read-speed test** on device/disk:

```
hdparm -tT /dev/[device]
```

Test for **unreadable blocks** on device/disk:

```
badblocks -s /dev/[device]
```

```
fsck [disk-or-partition-location]
```

Searching

Search for a specific pattern in a file with grep:

```
grep [pattern] [file_name]
```

Recursively search for a pattern in a directory:

```
grep -r [pattern] [directory_name]
```

Find all files and directories **related to a particular name**:

```
locate [name]
```

List names that **begin with a specified character `[a]`** in a specified location **`[/folder/location]`** by using the **`find`** command:

```
find [/folder/location] -name [a]
```

See **files larger than a specified size `[+100M]`** in a folder:

```
find [/folder/location] -size [+100M]
```

Note: Some commands are not recommended to use. Learn about them in our list of dangerous Linux commands.

File Commands

List files in the directory:

```
ls
```

List all files (shows hidden files):

```
ls -a
```

Show directory you are currently working in:

```
pwd
```

Create a new directory:

```
mkdir [directory]
```

Remove a file:

```
rm [file_name]
```

Remove a directory recursively:

```
rm -r [directory_name]
```

Recursively remove a directory without requiring confirmation:

```
rm -rf [directory_name]
```

Copy the contents of one file to another file:

```
cp [file_name1] [file_name2]
```

Recursively copy the contents of one file to a second file:

```
cp -r [directory_name1] [directory_name2]
```

Rename `[file_name1]` to **`[file_name2]`** with the command:

```
mv [file_name1] [file_name2]
```

Create a symbolic link to a file:

```
ln -s /path/to/[file_name] [link_name]
```

Create a **new file** using touch:

```
touch [file_name]
```

Show the contents of a file:

```
more [file_name]
```

or use the **`cat`** command:

```
cat [file_name]
```

Append file contents to another file:

```
cat [file_name1] >> [file_name2]
```

Display the **first 10 lines** of a file with head command:

```
head [file_name]
```

Show the **last 10 lines** of a file with tail command:

```
tail [file_name]
```

Encrypt a file:

```
gpg -c [file_name]
```

Decrypt a file:

```
gpg [file_name.gpg]
```

Show the **number of words, lines, and bytes** in a file using wc:

```
wc
```

List number of lines/words/characters in each file in a directory with the xargs command:

```
ls | xargs wc
```

Cut a section of a file and print the result to standard output:

```
cut -d[delimiter] [filename]
```

Cut a section of piped data and print the result to standard output:

```
[data] | cut -d[delimiter]
```

Print all lines matching a pattern in a file:

```
awk '[pattern] {print $0}' [filename]
```

Note: Learn also about gawk command, the GNU version of awk.

Overwrite a file to prevent its recovery, then delete it:

```
shred -u [filename]
```

Compare two files and display differences:

```
diff [file1] [file2]
```

Read and execute the file content in the current shell:

```
source [filename]
```

Sort file contents and print the result in standard output:

```
sort [options] filename
```

Store the command output in a file and skip the terminal output:

```
[command] | tee [filename] >/dev/null
```

how to create a le in Linux using the command line.
And if you want to find out how to determine the type of a file and its data, read our article about Linux file command.
To view a file's contents one screen at a time read about less command in Linux.

Directory Navigation

Move **up one level** in the directory tree structure:

```
cd ..
```

Change directory **to `$HOME`**:

```
cd
```

Change location to a specified directory:

```
cd /chosen/directory
```

File Compression

Archive an existing file:

```
tar cf [compressed_file.tar] [file_name]
```

Extract an archived file:

```
tar xf [compressed_file.tar]
```

Create a **gzip compressed tar file** by running:

```
tar czf [compressed_file.tar.gz]
```

Compress a file with the **.gz** extension:

```
gzip [file_name]
```

Note: For a more comprehensive overview of how to use **tar** refer to our guide tar Command in Linux With Examples.

File Transfer

Copy a file to a server directory securely using the Linux scp command:

```
scp [file_name.txt] [server/tmp]
```

Synchronize the contents of a directory **with a backup directory** using the rsync command:

```
rsync -a [/your/directory] [/backup/]
```

Users and Groups

See details about the **active users**:

```
id
```

Show **last system logins**:

```
last
```

Display who is **currently logged into the system** with the who command:

```
who
```

Show which users are **logged in** and **their activity**:

```
w
```

Add a new group by typing:

```
groupadd [group_name]
```

Add a **new user**:

```
adduser [user_name]
```

Add a **user to a group**:

```
usermod -aG [group_name] [user_name]
```

Temporarily **elevate user privileges** to superuser or root using the sudo command:

```
sudo [command_to_be_executed_as_superuser]
```

Delete a user:

```
userdel [user_name]
```

Modify user information with:

```
usermod
```

Change directory group:

```
chgrp [group-name] [directory-name]
```

Note: If you want to learn more about users and groups, take a look at our article on how to add a user to a group in Linux.

Package Installation

List all installed packages with **yum**:

```
yum list installed
```

Find a package by a **related keyword**:

```
yum search [keyword]
```

Show **package information and summary**:

```
yum info [package_name]
```

Install a package using the **YUM package manager**:

```
yum install [package_name.rpm]
```

Install a package using the **DNF package manager**:

```
dnf install [package_name.rpm]
```

Install a package using the **APT package manager**:

```
apt install [package_name]
```

Install an `.rpm` package from a local file:

```
rpm -i  [package_name.rpm]
```

Remove an `.rpm` package:

```
rpm -e [package_name.rpm]
```

Install software from **source code**:

```
tar zxvf [source_code.tar.gz]
cd [source_code]
./configure
make
make install
```

Process Related

See a **snapshot of active processes**:

```
ps
```

Show **processes in a tree-like diagram**:

```
pstree
```

Display a **memory usage map** of processes:

```
pmap
```

See all running processes:

```
top
```

Terminate a Linux process under a **given ID**:

```
kill [process_id]
```

Terminate a process under a **specific name**:

```
pkill [proc_name]
```

Terminate all processes **labelled "proc"**:

```
killall [proc_name]
```

List and resume stopped jobs in the background:

```
bg
```

Bring the most **recently suspended job to the foreground**:

```
fg
```

Bring a **particular job to the foreground**:

```
fg [job]
```

List **files opened by running processes** with lsof command:

```
lsof
```

Catch a system error signal in a shell script:

```
trap "[commands-to-execute-on-trapping]" [signal]
```

Pause terminal or a Bash script until a running process is completed:

```
wait
```

Run a Linux process in the background:

```
nohup [command] &
```

Note: If you want to learn more about shell jobs, how to terminate jobs or keep them running after you log off, check out our article on how to use disown command.

System Management and Information

Show **system information** via uname command:

```
uname -r
```

See kernel release information:

```
uname -a
```

Display **how long the system has been running**, including load average:

```
uptime
```

See system **hostname**:

```
hostname
```

Show the **IP address** of the system:

```
hostname -i
```

List system **reboot history**:

```
last reboot
```

See current time and date:

```
date
```

Query and **change the system clock** with:

```
timedatectl
```

Show current **calendar** (month and day):

```
cal
```

List logged in users:

```
w
```

See which **user you are using**:

```
whoami
```

Show **information about a particular user**:

```
finger [username]
```

View or limit system resource amounts:

```
ulimit [flags] [limit]
```

Schedule a system shutdown:

```
shutdown [hh:mm]
```

Shut Down the system immediately:

```
shutdown now
```

Add a new kernel module:

```
modprobe [module-name]
```

Disk Usage

You can use the df and du commands to check disk space in Linux.

See **free and used space** on mounted systems:

```
df -h
```

Show **free inodes** on mounted filesystems:

```
df -i
```

Display **disk partitions, sizes, and types** with the command:

```
fdisk -l
```

See **disk usage** for all files and directory:

```
du -ah
```

Show **disk usage of the directory** you are currently in:

```
du -sh
```

Display **target mount point** for all filesystem:

```
findmnt
```

Mount a device:

```
mount [device_path] [mount_point]
```

SSH Login

Connect to host as user:

```
ssh user@host
```

Securely **connect to host via SSH** default port 22:

```
ssh host
```

Connect to host **using a particular port**:

```
ssh -p [port] user@host
```

Connect to host **via telnet default port 23**:

```
telnet host
```

Note: For a detailed explanation of SSH Linux Commands, refer to our 19 Common SSH Commands in Linux tutorial.

File Permission

Chown command in Linux changes file and directory ownership.

Assign **read, write, and execute permission** to everyone:

```
chmod 777 [file_name]
```

Give **read, write, and execute permission to owner**, and r**ead and execute permission to group and others**:

```
chmod 755 [file_name]
```

Assign **full permission to owner**, and **read and write permission to group and others**:

```
chmod 766 [file_name]
```

Change the **ownership of a file**:

```
chown [user] [file_name]
```

Change the **owner and group ownership of a file**:

```
chown [user]:[group] [file_name]
```

Note: To learn more about how to check and change permissions, refer to our Linux File Permission Tutorial.

Network

List IP addresses and **network interfaces**:

```
ip addr show
```

Assign an **IP address to interface eth0**:

```
ip address add [IP_address]
```

Display **IP addresses of all network interfaces** with:

```
ifconfig
```

See **active (listening) ports** with the netstat command:

```
netstat -pnltu
```

Show **tcp** and **udp ports** and their programs:

```
netstat -nutlp
```

Display more **information about a domain**:

```
whois [domain]
```

Show **DNS information** about a domain using the dig command:

```
dig [domain]
```

Do a **reverse lookup on domain**:

```
dig -x host
```

Do **reverse lookup of an IP address**:

```
dig -x [ip_address]
```

Perform an **IP lookup for a domain**:

```
host [domain]
```

Show the **local IP address**:

```
hostname -I
```

Download a file from a domain using the **`wget`** command:

```
wget [file_name]
```

Receive information about an internet domain:

```
nslookup [domain-name]
```

Save a remote file to your system using the filename that corresponds to the filename on the server:

```
curl -O [file-url]
```

Variables

Assign an integer value to a variable:

```
let "[variable]=[value]"
```

Export a Bash variable:

```
export [variable-name]
```

Declare a Bash variable:

```
declare [variable-name]= "[value]"
```

List the names of all the shell variables and functions:

```
set
```

Display the value of a variable:

```
echo $[variable-name]
```

Shell Command Management

Create an alias for a command:

```
alias [alias-name]='[command]'
```

Set a custom interval to run a user-defined command:

```
watch -n [interval-in-seconds] [command]
```

Postpone the execution of a command:

```
sleep [time-interval] && [command]
```

Create a job to be executed at a certain time (**Ctrl+D** to exit prompt after you type in the command):

```
at [hh:mm]
```

Display a built-in manual for a command:

```
man [command]
```

Print the history of the commands you used in the terminal:

```
history
```

Linux Keyboard Shortcuts

Kill process running in the terminal:

```
Ctrl + C
```

Stop **current process**:

```
Ctrl + Z
```

The process can be **resumed** in the **foreground** with `fg` or in the **background** with `bg`.

Cut **one word before the cursor** and add it to clipboard:

```
Ctrl + W
```

Cut **part of the line before the cursor** and add it to clipboard:

```
Ctrl + U
```

Cut **part of the line after the cursor** and add it to clipboard:

```
Ctrl + K
```

Paste from clipboard:

```
Ctrl + Y
```

Recall last command that matches the provided characters:

```
Ctrl + R
```

Run the previously recalled command:

```
Ctrl + O
```

Exit command history without running a command:

```
Ctrl + G
```

Run the last command again:

```
!!
```

Log out of current session:

```
exit
```