Linux Commands Handbook

A comprehensive guide to essential Linux commands, complete with descriptions, real-world use cases, and practical examples.

Table of Contents

- make
- strace
- gdb
- git
- vim/nano

- **Other Useful Commands**
  - whoami
  - users
  - hostname
  - env
  - export
  - date
  - cal
  - man
  - history
  - uname

Basic Linux Commands

1. pwd

- **Description**: Shows you the full path of the directory you are currently in. `pwd` stands for "print working directory".
- **Real-World Use Case**: When you've navigated through several directories and need to know your exact location in the filesystem, especially before running commands that depend on the current path.

Examples:

```
# Display the current working directory
pwd
# Store the current directory path in a variable for use in a script
current_dir=$(pwd)
echo "I am currently in: $current_dir"
```

2. ls

- **Description**: Lists the files and directories inside your current directory.
- **Real-World Use Case**: To see what files are in a folder, check their permissions, sizes, and modification dates. It's one of the most frequently used commands.

Examples:

```
# List all files and directories in the current location, including hidden ones, in a detailed format
ls -la
# List files in a human-readable format (e.g., KB, MB, GB) and sort them by modification time (newest first)
ls -lht
```

3. cd

- **Description**: Changes your current directory to a different one. `cd` stands for "change directory".
- **Real-World Use Case**: To navigate through the folders on your system to get to a specific file or run a program.

Examples:

```
# Navigate to the /home/user/documents directory
cd /home/user/documents
# Go back to the previous directory you were in
cd -
```

4. touch

- **Description**: Creates a new, empty file. If the file already exists, it updates its timestamp without changing the content.
- **Real-World Use Case**: Quickly creating a new file for notes, a script, or as a placeholder. Also used in scripts to update a file's modification time.

Examples:

```
# Create a new empty file named my_notes.txt
touch my_notes.txt
# Create multiple files at once
touch file1.txt file2.html file3.js
```

5. mkdir

- **Description**: Creates a new directory (or folder). `mkdir` stands for "make directory".
- **Real-World Use Case**: To organize your files by creating new folders for projects, documents, or photos.

Examples:

```
# Create a new directory named "Project_Alpha"
mkdir Project_Alpha
# Create a nested directory structure in one go
mkdir -p Projects/2024/Q1
```

6. rm

- **Description**: Deletes files or directories. Be careful, as this action is permanent. `rm` stands for "remove".
- **Real-World Use Case**: To clean up old files, delete logs, or remove unnecessary folders.

Examples:

```
# Forcefully remove a file without being prompted for confirmation
rm -f old_report.txt
# Recursively and forcefully remove a directory and all its contents
rm -rf old_project/
```

7. rmdir

- **Description**: Deletes *empty* directories. It will not work if the directory contains any files or other folders.
- **Real-World Use Case**: To safely remove a directory that you know should be empty, preventing accidental deletion of files.

Examples:

```
# Remove an empty directory named "temp_files"
rmdir temp_files
# Remove multiple empty directories at once
rmdir empty_folder1 empty_folder2
```

8. cp

- **Description**: Copies files or directories from one location to another. `cp` stands for "copy".
- **Real-World Use Case**: To back up a file before editing it, or to copy a directory of project files to a new location.

Examples:

```
# Copy a file named "config.txt" to a new file named "config.bak" in the same directory
cp config.txt config.bak
# Recursively copy an entire directory "ProjectX" to the "/backups/" directory
cp -r ProjectX /backups/
```

9. mv

- **Description**: Moves a file or directory from one location to another. It is also used to rename files and directories. `mv` stands for "move".
- **Real-World Use Case**: To reorganize files by moving them into different folders, or to give a file a more descriptive name.

Examples:

```
# Rename the file "draft.txt" to "final_report.txt"
mv draft.txt final_report.txt
# Move the file "image.jpg" into the "/home/user/Pictures" directory
mv image.jpg /home/user/Pictures/
```

10. cat

- **Description**: Displays the entire content of a file on the screen. `cat` comes from "catenate," as it can also be used to join files together.
- **Real-World Use Case**: To quickly view the contents of a small configuration file or a short text file without opening an editor.

Examples:

```
# Display the content of the file "shopping_list.txt"
cat shopping_list.txt
```

```
# Combine two files and save the result i
nto a new file
cat part1.txt part2.txt > full_document.t
xt
```

11. echo

- **Description**: Displays a line of text or a variable's value in the terminal.
- **Real-World Use Case**: Used in shell scripts to print status messages, debug information, or write content to a file.

Examples:

```
# Print a simple "Hello, World!" message
to the terminal
echo "Hello, World!"
# Append a new line of text to a file
echo "This is a new line." >> my_log.txt
```

12. clear

- **Description**: Clears all the text from your terminal screen, giving you a clean slate.
- **Real-World Use Case**: When your terminal is cluttered with output from previous commands and you want a fresh, clean view.

Examples:

```
# Clear the terminal screen
clear
# Clear the screen and then list the file
s in the current directory
clear && ls -l
```

Intermediate Linux Commands

13. chmod

- **Description**: Changes the access permissions (read, write, execute) of files and directories. chmod stands for "change mode".
- **Real-World Use Case**: To make a script executable so you can run it, or to make a file read-only to prevent accidental changes.

Examples:

```
# Make a shell script executable for the
owner of the file
chmod u+x my_script.sh
# Set permissions so the owner can read/w
rite/execute, group members can read/exec
ute, and others can only read
chmod 754 important_file.txt
```

14. chown

- **Description**: Changes the owner and group of a file or directory. chown stands for "change owner".
- **Real-World Use Case**: When you need to transfer ownership of files to another user, for example, after copying files from a system administrator account to a user's home directory.

Examples:

```
# Change the owner of "file.txt" to the u
ser "john"
chown john file.txt
# Recursively change the owner and group
of the "/var/www/html" directory to "www-
data"
chown -R www-data:www-data /var/www/html
```

15. find

- **Description**: Searches for files and directories in a directory hierarchy based on criteria like name, size, modification time, etc.
- **Real-World Use Case**: To find a specific file you can't remember the location of, or to find all files of a certain type (e.g., all .log files) to delete them.

Examples:

```
# Find all files in the current directory
and subdirectories with the ".log" extens
ion
find . -name "*.log"
# Find all directories named "temp" insid
e the /home directory
find /home -type d -name "temp"
```

16. grep

- **Description**: Searches for a specific pattern of text inside files. grep stands for "global regular expression print".
- **Real-World Use Case**: To find a specific error message in a log file, or to find a file that contains a particular function name in a codebase.

Examples:

```
# Search for the word "error" in the syst
em log file, ignoring case
grep -i "error" /var/log/syslog
# Recursively search for the text "API_KE
Y" in all files in the current directory
grep -r "API_KEY" .
```

17. wc

- **Description**: Counts the number of lines, words, and characters in a file. wc stands for "word count".
- **Real-World Use Case**: To quickly check the length of a document or to count how many lines are in a log file.

Examples:

```
# Count the number of lines in "document.
txt"
wc -l document.txt
# Count the number of words in a file
wc -w document.txt
```

18. head

- **Description**: Displays the first few lines of a file. By default, it shows the first 10 lines.
- **Real-World Use Case**: To quickly peek at the beginning of a large file to see what it's about, without opening the whole thing.

Examples:

```
# Show the first 5 lines of a log file
head -n 5 access.log
# Display the first 20 lines of a CSV fil
e
head -n 20 data.csv
```

19. tail

- **Description**: Displays the last few lines of a file. By default, it shows the last 10 lines.
- **Real-World Use Case**: To monitor a log file in real-time to see new entries as they are added.

Examples:

```
# Show the last 20 lines of the system lo
g
tail -n 20 /var/log/syslog
# Follow a log file in real-time to see n
ew lines as they are written
tail -f application.log
```

20. sort

- **Description**: Sorts the lines of a text file alphabetically or numerically.
- **Real-World Use Case**: To sort a list of names in a file alphabetically or to sort a list of numbers from smallest to largest.

Examples:

```
# Sort the lines in "names.txt" alphabeti
cally
sort names.txt
# Sort a file of numbers in numeric order
sort -n numbers.txt
```

21. uniq

- **Description**: Removes duplicate adjacent lines from a sorted file. It's important to use sort first.
- **Real-World Use Case**: To get a list of unique visitors from a web server log after sorting the IP addresses.

Examples:

```
# Count the occurrences of each line in a
file (after sorting)
sort data.txt | uniq -c
# Show only the duplicate lines from a so
rted file
sort data.txt | uniq -d
```

22. diff

- **Description**: Compares two files line by line and shows the differences between them.
- **Real-World Use Case**: To see what changes have been made between two versions of a configuration file or a source code file.

Examples:

```
# Compare "config_v1.txt" and "config_v2.txt" and show the differences
diff config_v1.txt config_v2.txt
# Show a "unified" diff, which is often easier to read
diff -u old_file.txt new_file.txt
```

23. tar

- **Description**: Creates or extracts files from a `.tar` archive, also known as a tarball. It bundles multiple files into a single file.
- **Real-World Use Case**: To archive a whole project directory into a single file for backup or transfer.

Examples:

```
# Create a gzipped tar archive of the "my_project" directory
tar -czvf my_project.tar.gz my_project/
# Extract the contents of a tarball into the current directory
tar -xzvf backup.tar.gz
```

24. zip/unzip

- **Description**: Compresses files into a `.zip` archive or extracts files from one.
- **Real-World Use Case**: To compress files to save space or to bundle them for easy sharing with others, especially on non-Linux systems.

Examples:

```
# Zip a directory and its contents
zip -r archive.zip my_folder/
# Unzip a file into the current directory
unzip archive.zip
```

25. df

- **Description**: Shows the amount of disk space used and available on your filesystems. `df` stands for "disk free".
- **Real-World Use Case**: To check if your hard drive is getting full.

Examples:

```
# Display disk space usage in a human-readable format (KB, MB, GB)
df -h
# Show usage for a specific filesystem, like the root filesystem
df -h /
```

26. du

- **Description**: Shows the disk space used by files and directories. `du` stands for "disk usage".
- **Real-World Use Case**: To find out which folders are taking up the most space on your disk.

Examples:

```
# Show the size of the current directory and its subdirectories in a human-readable format
du -h
# Show a summarized, human-readable total for the current directory
du -sh .
```

27. top

- **Description**: Displays a real-time, dynamic view of the running processes on your system. It shows CPU usage, memory usage, and more.
- **Real-World Use Case**: To identify which programs are using the most CPU or memory, especially when your system is slow.

Examples:

```
# Start the top utility
top
# Run top and show processes for a specific user
top -u username
```

28. ps

- **Description**: Shows a snapshot of the currently active processes.
- **Real-World Use Case**: To find the Process ID (PID) of a program so you can terminate it, or to check if a specific program is running.

Examples:

```
# List all running processes on the syste
m in a detailed format
ps aux
# Find a specific process by name (e.g.,
find the firefox process)
ps aux | grep firefox
```

29. kill

- **Description**: Sends a signal to a process, typically to terminate it. You need the Process ID (PID) of the process.
- **Real-World Use Case**: To force-quit an application that has frozen or is not responding.

Examples:

```
# Terminate a process with PID 1234 grace
fully
kill 1234
# Forcefully terminate a process with PID
5678 (this is more aggressive)
kill -9 5678
```

30. ping

- **Description**: Sends a small packet of data to a network host (like a website) to see if it's reachable and how long it takes for the packet to come back.
- **Real-World Use Case**: To check if you have a working internet connection or to see if a specific server is online.

Examples:

```
# Ping google.com to check for a response
ping google.com
# Ping a server only 4 times and then sto
p
ping -c 4 8.8.8.8
```

31. wget

- **Description**: A command-line utility for downloading files from the internet.
- **Real-World Use Case**: To download a file directly to your server, or to download files in the background.

Examples:

```
# Download a file from a URL
wget [https://example.com/somefile.zip](h
ttps://example.com/somefile.zip)
# Download a file and save it with a diff
erent name
wget -O my_download.zip https://example.c
om/somefile.zip
```

32. curl

- **Description**: A versatile tool to transfer data from or to a server, using various protocols. It's often used for testing APIs.
- **Real-World Use Case**: To download a file, test if a web API is working, or send data to a web server.

Examples:

```
# Fetch the content of a webpage and disp
lay it in the terminal
curl https://example.com
# Download a file from a URL and save it
with its original name
curl -O [https://example.com/largefile.is
o](https://example.com/largefile.iso)
```

33. scp

- **Description**: Securely copies files between two computers on a network using the SSH protocol. `scp` stands for "secure copy".
- **Real-World Use Case**: To upload a file from your local machine to a remote server, or to download a file from a server.

Examples:

```
# Copy a local file to a remote server
scp /path/to/local/file.txt user@remote_h
ost:/path/to/remote/directory/
# Copy a file from a remote server to you
r local machine
scp user@remote_host:/path/to/remote/file
.txt /path/to/local/directory/
```

34. rsync

- **Description**: Efficiently synchronizes files and directories between two locations. It only copies the parts of files that have changed.
- **Real-World Use Case**: For making backups, as it's much faster than cp for subsequent backups of large directories.

Examples:

```
# Synchronize a local directory with a re
mote one, showing progress
rsync -avz --progress /local/path/ user@r
emote_host:/remote/path/
# Do a "dry run" to see what files would
be transferred without actually transferr
ing them
rsync -avzn /local/path/ /backup/path/
```

Advanced Linux Commands

35. awk

- **Description**: A powerful pattern-scanning and text-processing language. It can process data, perform calculations, and generate reports.
- **Real-World Use Case**: To extract specific columns of data from a text file or log, or to calculate sums and averages from structured data.

Examples:

```
# Print the first and third columns of a
space-separated file
awk '{print $1, $3}' data.txt
# Sum the values in the second column of
a file
awk '{s+=$2} END {print s}' numbers.txt
```

36. sed

- **Description**: A stream editor for filtering and transforming text. It can perform text substitutions, deletions, and insertions on a file or a stream of data.
- **Real-World Use Case**: To find and replace a word in a file, or to delete all lines containing a specific pattern.

Examples:

```
# Replace all occurrences of "apple" with
"orange" in a file
sed 's/apple/orange/g' fruits.txt
# Delete all lines that contain the word
"error"
sed '/error/d' logfile.txt
```

37. cut

- **Description**: Removes sections (columns) from each line of a file. It's simpler than awk for basic column extraction.
- **Real-World Use Case**: To extract a list of usernames from the `/etc/passwd` file, which uses colons as delimiters.

Examples:

```
# Extract the first field (username) from
the password file
cut -d':' -f1 /etc/passwd
# Extract characters 2 through 5 from eac
h line of a file
cut -c2-5 some_data.txt
```

38. tr

- **Description**: Translates or deletes characters from a stream of text.
- **Real-World Use Case**: To change all lowercase letters in a file to uppercase, or to replace spaces with tabs.

Examples:

```
# Convert all lowercase letters to upperc
ase
cat file.txt | tr 'a-z' 'A-Z'
# Replace all spaces with newlines, putti
ng each word on a new line
cat sentence.txt | tr ' ' '\n'
```

39. xargs

- **Description**: Builds and executes command lines from standard input. It takes the output of one command and uses it as the input (arguments) for another.
- **Real-World Use Case**: To delete all files with a `.log` extension that were found using the `find` command.

Examples:

```
# Find all .tmp files and delete them usi
ng rm
find . -name "*.tmp" | xargs rm
# Find all JPEG files and archive them wi
th tar
find . -name "*.jpg" | xargs tar -czvf im
ages.tar.gz
```

40. ln

- **Description**: Creates links between files. A “hard link” is a direct reference to the file data, while a “symbolic link” (or symlink) is a pointer to the file’s path.
- **Real-World Use Case**: To create a shortcut to a program in a different directory, or to have a file appear in multiple locations without duplicating it.

Examples:

```
# Create a symbolic link to a file
ln -s /path/to/original/file.txt /path/to
/link.txt
# Create a hard link to a file
ln /path/to/original/file.txt /path/to/ha
rdlink.txt
```

41. free

- **Description**: Displays the amount of free and used memory in the system, including physical memory (RAM) and swap space.
- **Real-World Use Case**: To quickly check if your system is running out of memory.

Examples:

```
# Display memory usage in a human-readabl
e format (MB, GB)
free -h
# Display memory usage every 2 seconds
free -s 2
```

42. iostat

- **Description**: Reports Central Processing Unit (CPU) statistics and input/output statistics for devices and partitions.
- **Real-World Use Case**: To identify performance bottlenecks by checking if your storage devices (hard drives, SSDs) are being overworked.

Examples:

```
# Display a basic CPU and I/O report
iostat
# Display a report every 2 seconds, 5 tim
es, in a more readable format (kilobytes)
iostat -k 2 5
```

43. netstat

- **Description**: Displays network connections, routing tables, interface statistics, and more. (Note: `ss` is the modern replacement).
- **Real-World Use Case**: To see which programs are listening for network connections on your server or to check active network connections.

Examples:

```
# List all listening TCP and UDP ports
netstat -tuln
# Show the routing table
netstat -r
```

44. ip

- **Description**: The modern tool for showing and manipulating routing, network devices, interfaces, and tunnels. It replaces older tools like `ifconfig` and `route`.
- **Real-World Use Case**: To find your computer's IP address or to configure network settings from the command line.

Examples:

```
# Show the IP addresses of all network in
terfaces
ip addr show
# Show the system's routing table
ip route show
```

45. iptables

- **Description**: A powerful tool for configuring the Linux kernel firewall. It allows you to set up rules for filtering network traffic.
- **Real-World Use Case**: To block incoming traffic from a specific IP address or to allow traffic only on specific ports (like for a web server).

Examples:

```
# List all current firewall rules
iptables -L
# Block all incoming traffic from the IP
address 192.168.1.100
iptables -A INPUT -s 192.168.1.100 -j DRO
P
```

46. systemctl

- **Description**: The main tool for controlling `systemd`, the system and service manager in modern Linux distributions. It's used to start, stop, and manage services.
- **Real-World Use Case**: To start a web server after a reboot, check the status of a database service, or enable a service to start automatically at boot.

Examples:

```
# Check the status of the Apache web serv
er service
systemctl status apache2
# Start the nginx service
systemctl start nginx
```

47. journalctl

- **Description**: Queries and displays logs from `journald`, the systemd logging service. It provides a centralized and structured way to view system logs.
- **Real-World Use Case**: To troubleshoot a service that failed to start or to view all system error messages from the last hour.

Examples:

```
# View all logs for the sshd service
journalctl -u sshd
# Follow the system log in real-time
journalctl -f
```

48. crontab

- **Description**: Manages `cron` jobs, which are scheduled tasks that run automatically at specified times or intervals.
- **Real-World Use Case**: To schedule a backup script to run every night at 2 AM, or to run a cleanup script every hour.

Examples:

```
# Edit the current user's crontab file
crontab -e
# List the scheduled cron jobs for the cu
rrent user
crontab -l
```

49. at

- **Description**: Schedules a command to be run once at a specific time in the future.
- **Real-World Use Case**: To schedule a system reboot for later in the evening, or to run a resource-intensive command during off-peak hours.

Examples:

```
# Schedule a command to run at 10:30 PM
at 10:30 PM
# (You will then be prompted to enter the
command, e.g., /sbin/shutdown -r now)
# List the user's pending jobs
atq
```

Networking Commands

50. ip addr

- **Description**: A specific command within the `ip` utility to show and manage network addresses on devices.
- **Real-World Use Case**: The primary way to find your machine's IP address on modern Linux systems.

Examples:

```
# Show addresses for all network interfac
es
ip addr
# Show the address for a specific interfa
ce, like eth0
ip addr show eth0
```

51. ip route

- **Description**: A specific command within the `ip` utility to show and manage the kernel routing table.
- **Real-World Use Case**: To check the default gateway (your router's IP) that your system uses to connect to the internet.

Examples:

```
# Show the main routing table
ip route
# Add a new route to a specific network v
ia a gateway
ip route add 192.168.5.0/24 via 192.168.1
.1
```

52. traceroute

- **Description**: Traces the network path (the "hops") that packets take to reach a remote host.
- **Real-World Use Case**: To diagnose network slowdowns by seeing where in the path the connection is getting delayed or failing.

Examples:

```
# Trace the route to google.com
traceroute google.com
# Use TCP packets for the trace instead o
f the default UDP
traceroute -T example.com
```

53. nslookup

- **Description**: A tool for querying Domain Name System (DNS) servers to find the IP address associated with a domain name, or vice versa.
- **Real-World Use Case**: To check if a domain name is resolving to the correct IP address.

Examples:

```
# Find the IP address for google.com
nslookup google.com
# Find the domain name for a specific IP
address (reverse DNS lookup)
nslookup 8.8.8.8
```

54. dig

- **Description**: A more advanced and flexible tool for querying DNS servers than `nslookup`. It provides more detailed information.
- **Real-World Use Case**: For network administrators to get detailed DNS information, like mail exchange (MX) records for a domain.

Examples:

```
# Perform a basic DNS lookup for example.
com
dig example.com
# Get only the MX (mail) records for a do
main
dig mx google.com
```

55. ssh

- **Description**: Securely connects to a remote computer over a network. All traffic is encrypted. `ssh` stands for "Secure Shell".
- **Real-World Use Case**: The standard way for system administrators to log in to and manage remote servers.

Examples:

```
# Connect to a remote server with the use
rname "admin"
ssh admin@192.168.1.100
# Connect to a server on a specific port
(e.g., 2222)
ssh -p 2222 user@remote.server.com
```

56. nmap

- **Description**: A powerful network scanning tool used for network discovery and security auditing. It can discover hosts and services on a network.
- **Real-World Use Case**: For security professionals to scan a network for open ports and potential vulnerabilities.

Examples:

```
# Scan a host to see which common ports a
re open
nmap scanme.nmap.org
# Perform an aggressive scan to detect OS
versions and services
nmap -A 192.168.1.1
```

57. netcat (nc)

- **Description**: A versatile networking utility for reading from and writing to network connections using TCP or UDP.
- **Real-World Use Case**: For testing network services, transferring files, or creating simple client/server applications.

Examples:

```
# Check if a port is open on a remote hos
t
nc -zv google.com 80
# Create a simple chat server on port 123
4
nc -l 1234
```

File Management & Search

58. locate

- **Description**: Finds files quickly by searching a pre-built database of all files on the system. It's much faster than `find` but may not have the most up-to-date information.
- **Real-World Use Case**: When you need to find a file instantly and you know its name (or part of it).

Examples:

```
# Find any file or directory containing t
he word "bashrc"
locate bashrc
# Update the locate database (requires ad
min privileges)
sudo updated
```

59. stat

- **Description**: Displays detailed information about a file or filesystem, such as size, permissions, access/modification times, and inode number.
- **Real-World Use Case**: To see exactly when a file was last accessed or modified, beyond the basic information `ls -l` provides.

Examples:

```
# Display detailed status information for
a file
stat my_document.txt
# Display information about a directory
stat /home/user
```

60. tree

- **Description**: Lists the contents of directories in a tree-like format, making it easy to visualize the directory structure.
- **Real-World Use Case**: To get a clear overview of a project's directory and file structure.

Examples:

```
# Display the directory tree starting fro
m the current location
tree
# Display the tree up to 2 levels deep
tree -L 2
```

61. file

- **Description**: Determines the type of a file by examining its contents (not just its extension).
- **Real-World Use Case**: To figure out what a file is when it has no extension or an incorrect one.

Examples:

```
# Determine the type of a file named "mys
tery_file"
file mystery_file
# Check the type of an image file
file logo.png
```

62. basename

- **Description**: Extracts the filename from a given path, stripping away the directory part.
- **Real-World Use Case**: In shell scripts, to get just the name of a file when you have its full path.

Examples:

```
# Get the filename from a full path
basename /home/user/documents/report.txt
# Remove a specific suffix from the filen
ame
basename /home/user/images/photo.jpg .jpg
```

63. dirname

- **Description**: Extracts the directory part from a given path, stripping away the filename.
- **Real-World Use Case**: In shell scripts, to find the directory where a script is located.

Examples:

```
# Get the directory path from a full file
path
dirname /home/user/documents/report.txt
# Get the parent directory of a directory
dirname /var/log/apache2/
```

System Monitoring

64. vmstat

- **Description**: Reports information about processes, memory, paging, block IO, traps, and CPU activity. `vmstat` stands for “virtual memory statistics”.
- **Real-World Use Case**: For a quick overview of system performance, including memory usage and CPU load.

Examples:

```
# Run vmstat and provide a report every 2 seconds
vmstat 2
# Show statistics in a more readable format (megabytes)
vmstat -S M
```

65. htop

- **Description**: An interactive process viewer and system monitor. It’s a more user-friendly and colorful alternative to `top`.
- **Real-World Use Case**: The preferred tool for many to interactively manage processes (e.g., kill, change priority) and monitor system resources.

Examples:

```
# Start the htop process viewer
htop
# Show only processes for a specific user
htop -u www-data
```

66. lsof

- **Description**: Lists all files that are currently opened by processes. `lsof` stands for “list open files”.
- **Real-World Use Case**: To find out which process is using a specific file or port, which is useful when you can’t unmount a disk because it’s “busy”.

Examples:

```
# List all processes that have a specific file open
lsof /var/log/syslog
# Find out which process is using TCP port 80
lsof -i :80
```

67. dmesg

- **Description**: Prints the message buffer of the kernel. These messages are generated by device drivers or the kernel itself.
- **Real-World Use Case**: To troubleshoot hardware issues, for example, to see if a USB drive was recognized when you plugged it in.

Examples:

~~~
# View all kernel messages
dmesg
```bash
# View the messages and pipe them to 'less' for easy scrolling
dmesg | less
~~~

68. uptime

- **Description**: Shows how long the system has been running since the last boot, the number of logged-in users, and the system load average.
- **Real-World Use Case**: To quickly check the stability of a server (a long uptime is often a good sign) and its current load.

Examples:

```
# Display the system uptime and load
uptime
# Get a "pretty" formatted output
uptime -p
```

69. iotop

- **Description**: A top-like utility for monitoring real-time disk I/O usage by processes.
- **Real-World Use Case**: To find out which process is causing heavy disk reads or writes, which can slow down the system.

Examples:

```
# Start iotop (requires root privileges)
sudo iotop
# Show only processes that are actually d
oing I/O
sudo iotop -o
```

Package Management

70. apt

- **Description**: The primary package manager for Debian-based Linux distributions like Ubuntu. Used to install, update, and remove software.
- **Real-World Use Case**: To install a new application like a web browser or to keep your system's software up-to-date.

Examples:

```
# Update the list of available packages
sudo apt update
# Install the package named "vlc"
sudo apt install vlc
```

71. yum/dnf

- **Description**: The package manager for RHEL-based distributions like CentOS and Fedora. `dnf` is the modern replacement for `yum`.
- **Real-World Use Case**: To install or update software on a Red Hat-based server.

Examples:

```
# Install the httpd package using dnf
sudo dnf install httpd
# Check for system updates using yum
sudo yum check-update
```

72. snap

- **Description**: A universal package management system that allows you to install self-contained applications ("snaps") that work across different Linux distributions.
- **Real-World Use Case**: To install an application that might not be available in your distribution's default repositories, like Spotify or Slack.

Examples:

```
# Find a snap package
snap find spotify
# Install the spotify snap
sudo snap install spotify
```

73. rpm

- **Description**: A command-line tool for managing `.rpm` packages, which are used by Red Hat-based systems. It can install, query, and verify packages.
- **Real-World Use Case**: To install a piece of software that was downloaded as an `.rpm` file directly from a vendor's website.

Examples:

```
# Install an rpm package file
sudo rpm -ivh some-package.rpm
# Query to see if a package is installed
rpm -q httpd
```

74. mount/umount

- **Description**: `mount` attaches a storage device or filesystem to a directory, making it accessible. `umount` detaches it.
- **Real-World Use Case**: To access the files on a USB drive after plugging it in, or to safely remove it.

Examples:

```
# Mount a USB drive (e.g., /dev/sdb1) to
the /mnt/usb directory
sudo mount /dev/sdb1 /mnt/usb
# Unmount the USB drive
sudo umount /mnt/usb
```

75. fsck

- **Description**: Checks and repairs inconsistencies in a Linux filesystem. `fsck` stands for “file system check”.
- **Real-World Use Case**: To fix filesystem errors that can occur after an improper shutdown or system crash.

Examples:

```
# Check the filesystem on a specific part
ition (run on an unmounted partition)
sudo fsck /dev/sda1
# Automatically repair any errors found w
ithout prompting
sudo fsck -y /dev/sdb1
```

76. mkfs

- **Description**: Creates a new filesystem (like `ext4`, `XFS`, etc.) on a storage device or partition. `mkfs` stands for “make filesystem”.
- **Real-World Use Case**: To format a new hard drive or USB stick so that it can be used by the operating system.

Examples:

```
# Create an ext4 filesystem on a partitio
n
sudo mkfs.ext4 /dev/sdb1
# Create a vfat filesystem (compatible wi
th Windows)
sudo mkfs.vfat /dev/sdc1
```

77. blkid

- **Description**: Displays information about available block devices (like hard drives), including their universally unique identifiers (UUIDs) and filesystem types.
- **Real-World Use Case**: To find the UUID of a partition, which is the recommended way to identify it in the `/etc/fstab` file for auto-mounting.

Examples:

```
# List all block devices and their attrib
utes
sudo blkid
# Get information for a specific device
sudo blkid /dev/sda1
```

78. lsblk

- **Description**: Lists information about all available or the specified block devices in a tree-like format.
- **Real-World Use Case**: To get a clear overview of your disks and partitions and how they are mounted.

Examples:

```
# List all block devices
lsblk
# Show more information, including filesy
stem type and UUID
lsblk -f
```

79. parted

- **Description**: A powerful tool for creating and manipulating disk partition tables.
- **Real-World Use Case**: To partition a new hard drive before creating filesystems on it.

Examples:

```
# List the partitions on a specific disk
sudo parted /dev/sda print
# Start parted in interactive mode for a
```

```
disk
sudo parted /dev/sdb
```

Scripting and Automation

80. bash

- **Description**: The Bourne Again Shell, a command-line interpreter and a powerful scripting language. It's the default shell for most Linux distributions.
- **Real-World Use Case**: Writing scripts to automate repetitive tasks, like backups, system updates, or processing files.

Examples:

```
# Start a new bash shell
bash
# Execute a shell script
bash my_script.sh
```

81. sh

- **Description**: The original Bourne shell, a simpler command interpreter. On modern systems, `/bin/sh` is often a symbolic link to a more feature-rich shell like `bash` or `dash`.
- **Real-World Use Case**: Writing scripts that need to be highly portable and work on older or more minimal systems.

Examples:

```
# Execute a script using the sh interpret
er
sh portable_script.sh
# Start an interactive sh shell
sh
```

82. cron

- **Description**: The system daemon that executes scheduled commands. `crontab` is the command used to manage the schedule.
- **Real-World Use Case**: The underlying service that runs the backup script you scheduled with `crontab` every night.

Examples:

```
# (cron is a service, typically managed w
ith systemctl)
# Check if the cron service is running
systemctl status cron
# View cron-related log messages
grep CRON /var/log/syslog
```

83. alias

- **Description**: Creates a shortcut or an alternative name for a command or a sequence of commands.
- **Real-World Use Case**: To create short, easy-to-remember commands for long, complex ones you use frequently.

Examples:

```
# Create an alias 'll' for the 'ls -la' c
ommand
alias ll='ls -la'
# Create an alias to quickly update your
system (for Debian/Ubuntu)
alias update='sudo apt update && sudo apt
upgrade -y'
```

84. source

- **Description**: Executes commands from a file in the *current* shell session. This is useful for loading environment variables or functions.
- **Real-World Use Case**: To apply changes to your `.bashrc` file immediately without having to log out and log back in.

Examples:

```
# Apply changes made to the .bashrc file
source ~/.bashrc
# A dot (.) is a shortcut for the source
command
. ~/.bashrc
```

Development and Debugging

85. gcc

- **Description**: The GNU Compiler Collection, a standard compiler for the C, C++, and Objective-C programming languages.
- **Real-World Use Case**: To compile source code written in C into an executable program.

Examples:

```
# Compile a simple C program and create a
n executable named "hello"
gcc hello.c -o hello
# Compile a program and include debugging
information
gcc -g my_program.c -o my_program
```

86. make

- **Description**: A build automation tool that automatically builds executable programs and libraries from source code by reading a file called a `Makefile`.
- **Real-World Use Case**: To simplify the compilation of large projects with many source files, by only recompiling files that have changed.

Examples:

```
# Run the default build process defined i
n the Makefile
make
# Run a specific target, like "clean", to
remove compiled files
make clean
```

87. strace

- **Description**: A debugging tool that traces system calls (interactions between a program and the Linux kernel) and signals.
- **Real-World Use Case**: To figure out why a program is crashing or behaving unexpectedly, by seeing which files it's trying to open or what system resources it's accessing.

Examples:

```
# Trace the system calls of a program
strace ./my_program
# Attach to a running process (with PID 1
234) and trace its calls
sudo strace -p 1234
```

88. gdb

- **Description**: The GNU Debugger, a powerful tool that allows you to see what is going on inside another program while it executes.
- **Real-World Use Case**: To step through code line-by-line, inspect variables, and find the exact location of a bug in a C or C++ program.

Examples:

```
# Start debugging an executable (must be
compiled with -g flag)
gdb ./my_program
# (Inside gdb) Set a breakpoint at the ma
in function and then run the program
(gdb) break main
(gdb) run
```

89. git

- **Description**: A distributed version control system for tracking changes in source code during software development.
- **Real-World Use Case**: The standard tool for managing codebases, collaborating with other developers, and tracking project history.

Examples:

```
# Clone a remote repository to your local
machine
git clone [https://github.com/user/repo.g
it](https://github.com/user/repo.git)
# Check the status of your local changes
git status
```

90. vim/nano

- **Description**: `vim` is a highly configurable and powerful text editor, while `nano` is a simpler, more user-friendly text editor.
- **Real-World Use Case**: For editing configuration files, writing scripts, or coding directly in the terminal.

Examples:

```
# Open a file with vim
vim /etc/hosts
# Open a file with nano
nano ~/.bashrc
```

Other Useful Commands

91. whoami

- **Description**: Prints the username of the user who is currently logged in.
- **Real-World Use Case**: To confirm which user account you are operating under, especially after using `su` or `sudo`.

Examples:

```
# Display the current user's username
whoami
# Use in a script to log which user ran i
t
echo "Script was run by: $(whoami)" >> sc
ript.log
```

92. users

- **Description**: Displays a list of usernames of all users currently logged in to the system.
- **Real-World Use Case**: To quickly see who else is on a multi-user server.

Examples:

```
# List logged-in users
users
# Pipe the output to wc to count the numb
er of logged-in users
users | wc -w
```

93. hostname

- **Description**: Displays or sets the system's host name.
- **Real-World Use Case**: To quickly identify which machine you are working on when connected to multiple servers.

Examples:

```
# Display the current hostname
hostname
# Display the IP address for the hostname
hostname -i
```

94. env

- **Description**: Displays all the environment variables for the current shell session.
- **Real-World Use Case**: To check the value of a specific variable like `PATH` to troubleshoot why a command isn't found.

Examples:

```
# List all environment variables
env
# Use with grep to find a specific variab
le
env | grep PATH
```

95. export

- **Description**: Sets an environment variable, making it available to all child processes started from that shell.
- **Real-World Use Case**: To set a temporary API key or configuration setting for a script you are about to run.

Examples:

```
# Create a new environment variable
export MY_VARIABLE="some_value"
# Add a new directory to the PATH variabl
e
export PATH=$PATH:/path/to/my/bin
```

96. date

- **Description**: Displays or sets the system date and time.
- **Real-World Use Case**: To check the current time on a server or to format the date for use in a script's log file name.

Examples:

```
# Display the current date and time
date
# Format the date to YYYY-MM-DD for a fil
ename
backup_filename="backup-$(date +%F).tar.g
z"
echo $backup_filename
```

97. cal

- **Description**: Displays a simple calendar in the terminal.
- **Real-World Use Case**: A quick and easy way to check the calendar without leaving the command line.

Examples:

```
# Display the calendar for the current mo
nth
cal
# Display the calendar for the entire yea
r 2025
cal 2025
```

98. man

- **Description**: Displays the online manual page (man page) for a specific command. This is the primary way to get help in Linux.
- **Real-World Use Case**: When you can't remember the options for a command like `tar` or `find`, the man page has all the details.

Examples:

```
# Show the manual page for the 'ls' comma
nd
man ls
# Show the manual for the crontab file fo
rmat (section 5)
man 5 crontab
```

99. history

- **Description**: Displays a list of the commands you have previously run in the current shell session.
- **Real-World Use Case**: To find and re-run a complex command you used earlier without having to type it all out again.

Examples:

```
# Display the command history
history
# Execute the 100th command from the hist
ory list
!100
```

100. uname

- **Description**: Prints basic information about the system's kernel and hardware architecture. `uname` stands for "unix name".
- **Real-World Use Case**: To quickly find out if you're on a 32-bit or 64-bit system, or to see the kernel version.

Examples:

```
# Print all available system information
uname -a
# Print just the kernel release version
uname -r
```