**User Information**

1. **who** It is used to get information about currently logged in user on to system. If you don't provide any option or arguments, the command displays the following information for each logged-in user.
   1. Login name of the user
   2. User terminal
   3. Date & Time of login
   4. Remote host name of the user
2. $ who

sudheer :0 2019-08-04 01:21 (:0)

1. **whoami:** It display the system’s username
2. $ whoami

sudheer

1. **id:** It display the user identification(the real and effective user and group IDs) information
2. $ id

uid=1000(sj) gid=1000(sj) groups=1000(sj),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),120(lpadmin),131(lxd),132(sambashare)

1. **groups:** This command is used to display all the groups for which the user belongs to.
2. $ group

sj: sj, adm, cdrom, sudo, dip, plugdev, lpadmin, lxd, sambashare

1. **finger:** Used to check the information of any currently logged in users. i.e, It displays users login time, tty (name), idle time, home directory, shell name etc.
2. $ finger
3. Login Name Tty Idle Login Time Office Office Phone

sj sj \*:0 Aug 28 01:27 (:0)

This may not be available by default in many linux machines. In this case, you need to install it manually.

$ sudo apt install finger

1. **users:** Displays usernames of all users currently logged on the system.
2. $ users

sj

1. **grep:** It is a powerful pattern searching tool to find information about a specific user from the system accounts file: /etc/passwd.
2. $ grep -i sj /etc/passwd

sj:x:1000:1000:sj,,,:/home/sj:/bin/bash

1. **W Command:** It(W) is a command-line utility that displays information about currently logged in users and what each user is doing.
2. w [OPTIONS] [USER]
3. Example:
4. w
5. 18:45:04 up 2:09, 1 user, load average: 0.09, 0.07, 0.02
6. USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT

sj :0 :0 01:27 ?xdm? 1:14 0.01s /usr/lib/gdm3/g

1. **last or lastb:** Displays a list of last logged in users on the system. You can pass user names to display their login and hostname details.
2. last [options] [username...] [tty...]
3. Example:
4. last
5. sj :0 :0 Fri Aug 28 01:27 gone - no logout
6. reboot system boot 5.4.0-29-generic Fri Aug 28 01:27 still running
7. sj :0 :0 Wed Jul 29 11:46 - crash (29+13:40)
8. reboot system boot 5.4.0-29-generic Wed Jul 29 11:45 still running
9. sj :0 :0 Thu May 14 21:04 - crash (75+14:41)
10. reboot system boot 5.4.0-29-generic Thu May 14 21:03 still running

wtmp begins Thu May 14 21:03:56 2020

1. **lastlog:** The lastlog command is used to find the details of a recent login of all users or of a given user.
2. $ lastlog
3. Username Port From Latest
4. root \*\*Never logged in\*\*
5. daemon \*\*Never logged in\*\*
6. bin \*\*Never logged in\*\*
7. sys \*\*Never logged in\*\*
8. sync \*\*Never logged in\*\*
9. games \*\*Never logged in\*\*
10. man \*\*Never logged in\*\*
11. lp \*\*Never logged in\*\*
12. mail \*\*Never logged in\*\*

news \*\*Never logged in\*\*

**File and directory commands**

1. **pwd** The pwd(Present Working Directory) command is used to print the name of the present/current working directory starting from the root.
2. $ pwd

/home/sj/Desktop/Linux

1. **ls**: The ls command is used to list files or directories. It also accepts some flags or options that changes how files or directories are listed in your terminal.
2. Syntax:
3. ls [flags] [directory]
4. Example:
5. $ ls
6. bin dev lib libx32 mnt
7. //Listing files & directories with time in a rever order
8. $ ls -ltr
9. drwxr-xr-x 2 sj sj 4096 May 14 2020 Videos
10. drwxr-xr-x 2 sj sj 4096 May 14 2020 Templates
11. drwxr-xr-x 2 sj sj 4096 May 14 2020 Public
12. //Home directory
13. $ ls ~
14. Desktop Downloads Pictures Sudheer test test.txt

Documents Music Public Templates test1 Videos

Below are the list of possible options for ls command,

-a Show all (including hidden)

-R Recursive list

-r Reverse order

-t Sort by last modified

-S Sort by file size

-l Long listing format

-1 One file per line

-m Comma-­sep­arated output

-Q Quoted output

1. **mkdir** The mkdir(make directory) command allows users to create directories or folders.
2. $ mkdir ubuntu
3. $ ls

ubuntu

The option '-p' is used to create multiple directories or parent directories at once.

$ mkdir -p dir1/dir2/dir3

$ cd dir1/dir2/dir3

~/Desktop/Linux/dir1/dir2/dir3$

1. **rmdir**: The rmdir(remove directories) is used to remove *empty* directories. Can be used to delete multiple empty directories as well. Safer to use compared to rm -r FolderName. This command can also be forced to delete non-empty directories.
   1. Remove empty directory:

rmdir FolderName

* 1. Remove multiple directories:

rmdir FolderName1 FolderName2 FolderName3

* 1. Remove non-empty directories:

rmdir FolderName1 --ignore-fail-on-non-empty

* 1. Remove entire directory tree. This command is similar to rmdir a/b/c a/b a:

rmdir -p a/b/c

1. **rm**: The rm(remove) command is used to remove objects such as files, directories, symbolic links etc from the file system.
   1. Remove file: The rm command is used to remove or delete a file

rm file\_name

* 1. Remove file forcefully: The rm command with -f option is used for removal of file without prompting for confirmation.

rm -f filename

* 1. Remove directory: The rm command with -r option is used to remove the directory and its contents recursively.

rm -r myDir

* 1. Remove directory forcefully: The rm command with -rf option is used to forcefully remove directory recursively.

rm -rf myDir

1. **touch**: The touch command is is used to create, change and modify timestamps of a file without any content.
   1. **Create a new file:** You can create a single file at a time using touch command. The file created is an empty file.

touch file\_name

* 1. **Create multiple files:** You can create the multiple numbers of files at the same time.

touch file1\_name file2\_name file3\_name

* 1. **Change access time:** The touch command with a option is used to change the access time of a file.

touch -a file\_name

* 1. **Change modification time:** The touch command with m option is used to change the modified time.

touch -m file\_name

* 1. **Use timestamp of other file:** The touch command with r option is used to get timestamp of another file.

touch -r file2 file1

In the above example, we get the timestamp of file1 for file2.

* 1. **Create file with Specific time:** The touch command with 't' option is used to create a file with specified time.

touch -t 1911010000 file\_name

1. **cat**: The cat command is used to create single or multiple files, view contain of file, concatenate files and redirect output in terminal or files.

$ cat [OPTION] [FILE]...

* 1. **Create a file:** Used to create a file with specific name, content and press exit using CTRL + D
  2. cat > file\_name1.txt

Hello, How are you?

* 1. **View file contents:** You can view contents of a single or more files by mentioning the filenames.

cat file\_name1 file\_name2

* 1. **More & Less options:** If a file having a large number of content that won’t fit in the output terminal then more & less options can be used to indiate additional content.
  2. cat file\_name1.txt | more

cat file\_name1.txt | less

**File permissions**

Since Linux is a multi-user operating system, it is necessary to provide security to prevent people from accessing each other’s confidential files. So Linux divides authorization into 2 levels,

1. **Ownership:** Each file or directory has assigned with 3 types of owners i. **User:** Owner of the file who created it. ii. **Group:** Group of users with the same access permissions to the file or directory. iii. **Other:** Applies to all other users on the system
2. **Permissions:** Each file or directory has following permissions for the above 3 types of owners.

i. **Read:** Give you the authority to open and read a file and lists its content for a directory.

ii. **Write:** Give you the authority to modify the contents of a file and add, remove and rename files stored in the directory.

iii. **Execute:** Give you the authority to run the program in Unix/Linux.

The permissions are indicated with below characters,

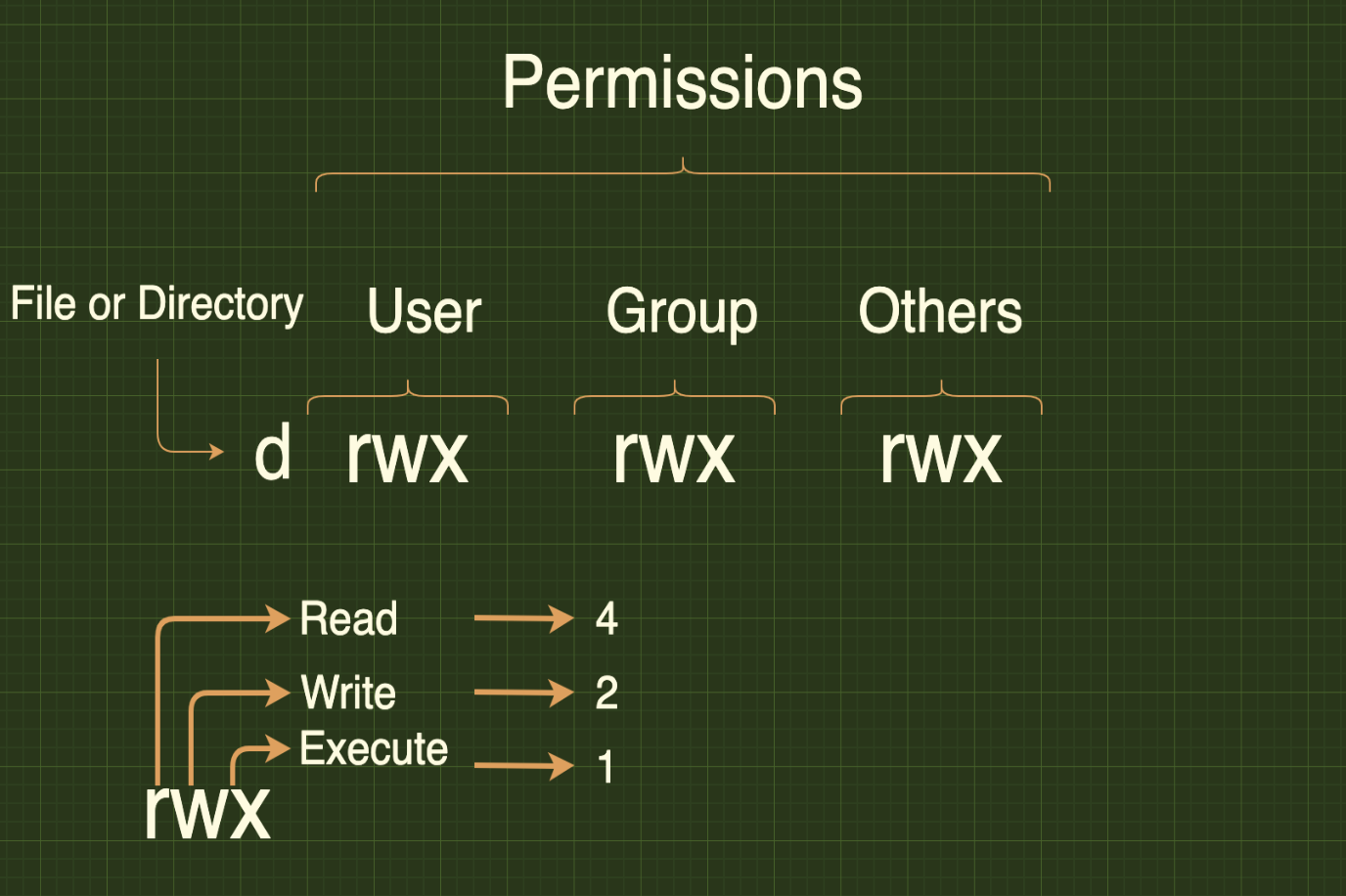
r = read permission

w = write permission

x = execute permission

\- = no permission

The above authorization levels represented in a diagram

[](https://github.com/sudheerj/Linux-cheat-sheet/blob/master/images/permissions.png)

There is a need to restrict own file/directory access to others.

**Change access:** The chmod command is used to change the access mode of a file. This command is used to set permissions (read, write, execute) on a file/directory for the owner, group and the others group.

chmod [reference][operator][mode] file...

Example

chmod ugo-rwx test.txt

There are 2 ways to use this command,

1. **Absolute mode:** The file permissions will be represented in a three-digit octal number.

The possible permissions types represented in a number format as below.

| **Permission Type** | **Number** | **Symbol** |
| --- | --- | --- |
| No Permission | 0 | --- |
| Execute | 1 | --x |
| Write | 2 | -w- |
| Execute + Write | 3 | -wx |
| Read | 4 | r-- |
| Read + Execute | 5 | r-x |
| Read + Write | 6 | rw- |
| Read + Write + Execute | 7 | rwx |

Let's update the permissions in absolute mode with an example as below,

chmode 764 test.txt

1. **Symbolic mode:** In the symbolic mode, you can modify permissions of a specific owner unlike absolute mode.

The owners are represented as below,

| **Owner** | **Description** |
| --- | --- |
| u | user/owner |
| g | group |
| o | other |
| a | all |

and the list of mathematical symbols to modify the file permissions as follows,

| **Operator** | **Description** |
| --- | --- |
| + | Adds permission |
| - | Removes the permission |
| = | Assign the permission |

**Changing Ownership and Group:** It is possible to change the the ownership and group of a file/directory using chown command.

chown user filename

chown user:group filename

Example:

chown John test.txt

chown John:Admin test.txt

**Change group-owner only:** Sometimes you may need to change group owner only. In this case, chgrp command need to be used

chgrp group\_name filename

Example:

sudo chgrp Administrator test.txt

**Networking**

1. **Display network information:** ifconfig command is used to display all network information(ip address, ports etc)

ifconfig -a

1. **Test connection to a remote machine:** Send an echo request to test connection of a remote machine.
2. ping <ip-address> or hostname
3. Example:

ping 10.0.0.11

1. **Show IP Address:** Display ip address of a currennt machine
2. hostname -I
3. (OR)

ip addr show

1. **Active ports:** Shows active or listening ports

netstat -pnltu

1. **Find information about domain:** whois command is used to find out information about a domain, such as the owner of the domain, the owner’s contact information, and the nameservers used by domain.
2. whois [domain]
3. Example:

whois google.com

**Installing packages**

1. **Install package:**

yum install package\_name

1. **Package description:** The info command is used to display brief details about a package.

yum info package\_name

1. **Uninstall package:** The remove command is used to remove or uninstall package name.

yum remove package\_name

1. **Install package from local file:**

It is also possible to install package from local file named package\_name.rpm.

rpm -i package\_name.rpm

1. **Install from source code:**

tar zxvf sourcecode.tar.gz

cd sourcecode

./configure

make

make install

**Disk usage**

1. **Synopsis:** du command is used to check the information of disk usage of files and directories on a machine

du [OPTION]... [FILE]...

1. **Disk usage of a directory:** To find out the disk usage summary of a /home/ directory tree and each of its sub directories

du /home/

1. **Disk usage in human readable format:** To find out the disk usage in human readable format

du -h /home/

1. **Total disk usage of a directory:** To find out the total disk usage

du -sh /home/

1. **Total disk usage of all files and directories:** To find out the total disk usage of files and directories

du -ah /home/

1. **Total disk usage of all files and directories upto certain depth:** print the total for a directory only if it is N or fewer levels below the command

du -ah --max-depth 2 /home/

1. **Total disk usage with excluded files:** To find out the total disk usage of files and directories, but excludes the files that matches given pattern.

du -ah --exclude="\*.txt" /home/

1. **Help:** This command gives information about du

du --help

**System and Hardware information**

1. **Print all information**: uname is mainly used to print system information.

$ uname -a

1. **Print kernel name**:

$ uname -s

1. **Print kernel release**:

$ uname -r

1. **Print Architecture**:

$ uname -m

1. **Print Operating System**:

$ uname -o

**Search Files**

1. **Pattern search:** The grep command is used to search patterns in files.

grep pattern files

grep -i // Case sensitive

grep -r // Recursive

grep -v // Inverted search

Example:

grep "^hello" test.txt // Hello John

grep -i "hELLo" text.txt // Hello John

1. **Find files and directories:**

The find command is used to find or search files and directories by file name, folder name, creation date, modification date, owner and permissions etc and perform subsequent operations on them.

i. **Search file with name:**

find ./directory\_name -name file\_name

Example:

find ./test -name test.txt // ./test/test.txt

ii. **Search file with pattern:**

find ./directory\_name -name file\_pattern

Example:

find ./test -name \*.txt // ./test/test.txt

iii. **Search file with executable action:**

find ./directory\_name -name file\_name -exec command

Example:

find ./test -name test.txt -exec rm -i {} \; // Search file and delete it after confirmation

iv. **Search for empty files or directories:**

The find command is used to search all empty folders and files in the entered directory or sub-directories.

find ./directory\_name -empty

Example:

find ./test -empty

//./test/test1

//./test/test2

//./test/test1.txt

v. **Search for files with permissions:**

The find command is used to find all the files in the mentioned directory or sub-directory with the given permissions

find ./directory\_name -perm permission\_code

Example:

find ./test -perm 664

vi. **Search text within multiple files:**

find ./ -type f -name file\_pattern -exec grep some\_text {} \;

Example:

find ./ -type f -name "\*.txt" -exec grep 'World' {} \; // Hello World

1. **Whereis to locate binary or source files for a command:** The whereis command in Linux is used to locate the binary, source, and manual page files for a command. i.e, It is used to It is used to find executables of a program, its man pages and configuration files.

whereis command\_name

Example:

whereis netstat //netstat: /bin/netstat /usr/share/man/man8/netstat.8.gz(i.e, executable and location of its man page)

1. **Locate to find files:** The locate command is used to find the files by name. This command is faster compared to find command because it searches database for the filename instead of searching your filesystem.

locate [OPTION] PATTERN

Example:

locate "\*.txt" -n 10 // 10 file search results ending with .txt extension

**SSH**

SSH (Secure Shell) is a network protocol that enables secure remote connections between two systems.

1. **Connect remote machine using IP address or machine name:** The remote server can be connected with local user name using either host name or IP address

ssh <host-name> or <ip-address>

Example:

ssh 192.111.66.100

ssh test.remoteserver.com

1. **Connect remote machine using username:** It is also possible specify a user for an SSH connection.

ssh username@hostname\_or\_ip-address

Example:

ssh john@192.0.0.22

ssh john@test.remoteserver.com

1. **:Connect remote machine using custom port** By default, the SSH server listens for a connection on port 22. But you can also specify the custom port.

ssh <host-name> or <ip-address> -p port\_number

Example:

ssh test.remoteserver.com -p 3322

1. **Generate SSH keys using keygen:** SSH Keygen is used to generate a key pair which consists of public and private keys to improve the security of SSH connections.

ssh-keygen -t rsa

1. **Copying SSH keys to servers:** For SSH authentication, ssh-copy-id command will be used to copy public key(id\_rsa.pub) to server.

ssh-copy-id hostname\_or\_IP

1. **Copy a File Remotely over SSH:** SCP tool is used to securely copy files over the SSH protocol.

scp fileName user@remotehost:destinationPath

Example:

scp test.txt test@10.0.0.64:/home/john/Desktop

1. **Edit SSH Config File** SSH server options customized by editing the settings in sshd\_config file.

sudo vim /etc/ssh/sshd\_config

1. **Run commands on a remote server** SSH commands can be executed on remote machine using the local machine.

ssh test.remoteserver.com mkdir NewDirectoryName // Creating directory on remote machine

1. **Restart SSH service:** You need to restart the service in Linux after making changes to SSH configuration.

sudo ssh service restart

(or)

sudo sshd service restart

**Vi/Vim-commands**

Vi editor is the most popular text editor from the early days of Unix. Whereas Vim(Vi IMproved) is an improved version of vi editor to be used in CLI (command line interface) for mainly text editing tasks in many configuration files. Some of the other alternatives are Elvis, Nvi, Nano, Joe, and Vile. It has two main operation modes,

1. **Command Mode:** It allows the entry of commands to manipulate text.
2. **Entry mode(Or Insert mode):** It allows typed characters on the keyboard into the current file.

**1. Start with Vi Editor**

You can create a new file or open an existing file using vi filename command.

vi <filename\_NEW> or <filename\_EXISTING> // Create a new file or open an existing file

Example:

vi first.txt

Let's see how do you create file, enter the content and leave the CLI by saving the changes.

1. Create a new file named first.txt
2. Press i to enter the insert mode
3. Enter the text "Hello World!"
4. Save the text and exit by pressing :wq! command
5. Check the entered text

**2. Cursor movement**

These commands will be used in Command mode.

**Move cursor**

You can use arrow keys(left, right, up and down) to move the cursor on the terminal. But you can also other keys for this behavior.

h # Move left

j # Move down

k # Move up

l # Move right

**Jump one word**

These commands used to jump one word at a time

w # Jump forwards to the start of a word

W # Jump forwards to the start of a WORD

e # Jump forwards to the start of a word

E # Jump forwards to the start of a WORD

b # Jump backwords to the start of a word

B # Jump backwords to the start of a WORD

**Jump to start or end of a line or next line**

These commands used to jump starting or ending of a line or a next line.

^ # Jump to the start of a current line

$ # Jump to the end of a current line

return # Jump to the start of a next line

**Move sides**

These commands used to moves all sides of the screen

Backspace # Move cursor one character to the left

Spacebar # Move cursor one character to the right

H(High) # Move cursor to the top of the screen

M(Middle) # Move cursor to the middle of the screen

L(Low) # Move cursor to the bottom of the screen

**Paging and Scrolling**

Paging is used to moves the cursor up or down through the text a full screen at a time. Whereas Scrolling happens line by line.

Ctrl + f # move forward one full screen

Ctrl + b # move backward one full screen

Ctrl + d # move forward half a screen

Ctrl + u # move backward half a screen

**Inserting Text**

These commands places vi in entry mode from command mode. First, you need to be in command mode to use the below commands.

**Insert**

i # Insert text to the left of the cursor

I # Insert text at the beginning of a line

ESC # Exit insert mode

**Append**

a # Insert(or Append) text to the right of the cursor

A # Insert(or Append) text at the end of a line

**Open a line**

o # Open a line below the current cursor position

O # open a line above the current cursor position

**Editing Text**

1. **Change word:** Change word/part of word to right of cursor

cw

1. **Change line** Change entire line

cc

1. **Change line from specific character** Change from cursor to end of line

C

**Deleting Text**

1. **Deleting One Character:** Position the cursor over the character to be deleted and type x
2. x

X //To delete one character before the cursor

1. **Deleting a Word:** Position the cursor at the beginning of the word and type dw

dw

1. **Deleting a Line:** Position the cursor anywhere on the line and type dd.

dd

**Cut, Copy & Paste**

Copy, Cut and Paste operations can be done in either Normal or visual Mode.

1. **Normal mode:** This mode appears on click of Esc key.

**Copy** There are various copy or yank commands based on amount of text to be copied. The y character is used to perform this operation.

i. Copy an entire line: Just place the cursor at the beginning of the line and type yy

yy

ii.Copy three lines: Just place the cursor from where to start copying and type 3yy

3yy

iii. Copy word with trailing whitespace: Place the cursor at the beginning of the word and type yaw

yaw

iv. Copy word without trailing whitespace: Place the cursor at the beginning of the word and type yiw.

yiw

v. Copy right of the cursor: Copy text right of the cursor to the end of line using y$ command

y$

vi.Copy left of the cursor: Copy text left of the cursor to the end of line using y^ command

y^

vii. Copy text between the cursor and character: Copy text between the cursor and specified character.

ytx(Copy until x and x is excluded)

yfx(Copy until x and x is included)

**Cut** There are various cutting or deleting commands based on amount of text to be deleted. The d character is used to perform this operation.

i. Cut entire line: Cut the entire line where the cursor is located

dd

ii.Cut three lines: Cut the three lines starting from the place where cursor is located

3dd

iii.Cut right of the cursor: Cut the text from the right of the cursor till the end of line

d$

iii.Cut left of the cursor: Cut the text from the left of the cursor till the beginning of line

d^

**Paste** This operation is performed using p command to paste the selected text

p

1. **Visual Mode** In this mode, first select the text using below keys
   1. v (lowercase): To select individual characters
   2. V (uppercase): To select the entire line
   3. Ctrl+v: To select by block

and perform copy, cut and paste operations using y,d and p commands

**Exiting**

These commands are used to exit from the file.

```cmd

:w # Write (save) the file, but don't exit

:wq # Write (save) and quit

:wq! # Force write (save) and quit

:q # Quit, but it fails if anything has changed

:q! # Quit and throw away for any changes

```

**Top/TaskManager Command**

top: The top command provides dynamic real-time viewing of the running system. It can display system summary information as well as a list of tasks currently being managed by the Linux kernel.

**options**

| **Option** | **Description** |
| --- | --- |
| -b | Batch mode |
| -n | Number of iterations |
| -s | Sec delay between updates |
| -d | Delay time in seconds |
| -p | Monitor process ID |

top [options]

**Kill Command**

kill: The kill command sends a signal to a process. Useful when trying to terminate a hung process.

**options:**

| **Option** | **Description** |
| --- | --- |
| -s | Specify the signal to send (e.g., -s SIGTERM) |
| -l | List available signals |
| -n | Process group ID |
| -signal | Specify the signal using signal name or number |
| -p | Specify the process ID to be signaled |

**kill Command Options**

1. -s Option Specifies the signal to send to the process. Use in the format -s SIGNAL. For example:

kill -s SIGTERM <process\_id>

1. -l Option Lists available signals that can be used with the kill command. This option does not send any signal; it only displays a list of signals.

kill -l

1. -n Option Specifies the process group ID. This option allows you to send a signal to a process group rather than an individual process.

kill -n <process\_group\_id>

1. -signal Option Specifies the signal to send using either the signal name or number directly.

kill -TERM <process\_id>

1. -p Option Specifies the process ID to be signaled. Use in the format -p <process\_id>.

kill -p <process\_id>

**History Command Options**

The history command shows a list of previously issued commands.

**history Command Options**

| **Option** | **Description** |
| --- | --- |
| -c | Clears the entire command history. |
| -d offset | Deletes the history entry at the specified offset. |
| -a | Appends the current session's history to the history file. |
| -r | Reads the history file and appends entries to the session. |
| -n | Prevents command numbers from being displayed. |
| -w | Writes the current session's history to the history file. |
| `-s "" | Appends the specified entry to the end of the history list. |
| -p offset | Prints the history entry at the specified offset. |
| -f file | Reads history entries from the specified file. |

1. -c Option Clears the entire command history.

history -c

1. -d offset Option Deletes the history entry at the specified offset.

history -d <offset>

1. -a Option Appends the current session's history to the history file.

history -a

1. -r Option Reads the history file and appends the entries to the current session's history.

history -r

1. -n Option Prevents command numbers from being displayed when using the history command.

history -n

1. -w Option Writes the current session's history to the history file.

history -w

1. -s Option Appends the specified entry to the end of the history list.

history -s "<command>"

1. -p Option Prints the history entry at the specified offset.

history -p <offset>

1. -f file Option Reads history entries from the specified file and appends them to the history list.

history -f <file>

**Curl Command**

Curl: The curl command transfers data to or from a server. It supports dozens of protocols, including HTTP, HTTPS, FTP, SFTP, Telnet, DICT, LDAP, LDAPS, FILE, IMAP, SMTP, POP3, RTSP and RTMP.

**Options**

| **Option** | **Description** |
| --- | --- |
| -X, --request <command> | Specify the HTTP request method (GET, POST, etc.). |
| -H, --header <header> | Pass custom headers to the request. |
| --data <data> | HTTP POST data. |
| --data-urlencode <data> | URL encode the POST data. |
| --form <name=content> | Specify multipart/form-data POST data. |
| -d, --data-raw <data> | Send data as-is without any encoding. |
| -i, --include | Include HTTP headers in the output. |
| --url <url> | Specify the URL to send the request to. |
| -o, --output <file> | Write output to a file instead of stdout. |
| -O, --remote-name | Save the file with the same name as in the URL. |
| -L, --location | Follow redirects. |
| --user <user:password> | Specify the user and password for server authentication. |
| -u, --user-agent <agent> | Set the user-agent string. |
| --cookie <data> | Send cookies from string/file. |
| --cookie-jar <file> | Save cookies to a file after receiving them. |
| --compressed | Request compressed response. |
| -v, --verbose | Make the operation more talkative for debugging. |
| -h, --help | Display help information. |

1. Basic GET Request: Sends a simple GET request to the specified URL.

curl http://example.com

1. Specify Request Method: Specifies a request method other than the default GET.

curl -X POST http://example.com

1. Include Headers:

Includes custom headers in the request.

curl -H "Content-Type: application/json" http://example.com

1. Send POST Data: Sends data as POST request payload.

curl -X POST --data "key1=value1&key2=value2" http://example.com

1. Save Output to File: Writes the output to a specified file.

curl -o output.html http://example.com

1. Download File: Downloads a file and saves it with the same name as in the URL.

curl -O http://example.com/file.zip