Linux for devops.

User and file management

1. uname:
    - This command displays system information about the current operating system.
    - Commonly used options include:
        - `-a`: **Print all system information.**
        - `-s`: **Print the kernel name.**
        - `-r`: **Print the kernel release.**
        - `-m`: **Print the machine hardware name.**
    - **Example:** `uname -a` **displays all system information.**

```
ubuntu@ip-172-31-2-194:~$ uname
Linux
```

2. **uptime:**
    - **This command displays how long the system has been running and the current system load average.**
    - **It typically shows the current time, the length of time the system has been up, the number of users logged in, and the system load averages for the past 1, 5, and 15 minutes.**
    - **Example:** `uptime` **displays system uptime and load averages.**

```
ubuntu@ip-172-31-2-194:~$ uptime
 20:43:33 up 0 min,  1 user,  load average: 1.42, 0.43, 0.15
```

3. **date:**
    - **This command displays or sets the system date and time.**
    - **It can be used to print the current date and time or to set the system clock to a specific date and time.**
    - **Example:** `date` **displays the current date and time.**

```
ubuntu@ip-172-31-2-194:~$ date
Sat May 11 20:43:37 UTC 2024
```

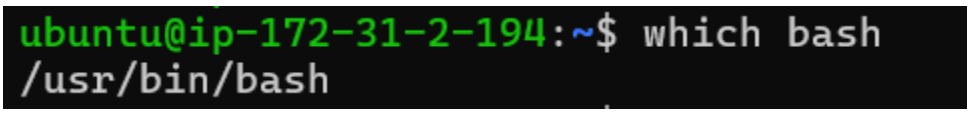
4. **who:**
    - **This command displays information about currently logged-in users.**
    - **It shows the username, terminal, login time, and source IP address (if available) of each logged-in user.**
    - **Example:** `who` **displays a list of logged-in users.**

```
ubuntu@ip-172-31-2-194:~$ who
ubuntu   pts/0        2024-05-11 20:43 (99.251.184.125)
```

5. **whoami:**
    - **This command prints the username of the current user.**
    - **It simply returns the username associated with the current user's session.**
    - **Example:** `whoami` **displays the current username.**

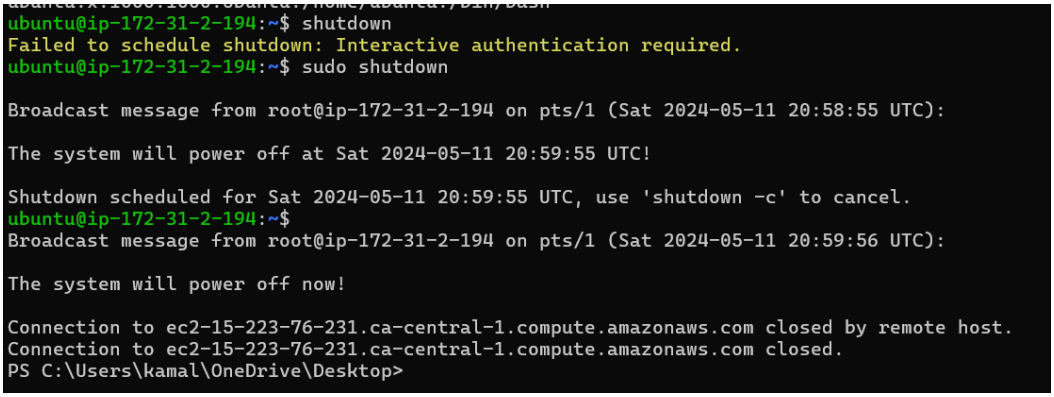
```
ubuntu@ip-172-31-2-194:~$ whoami
ubuntu
```

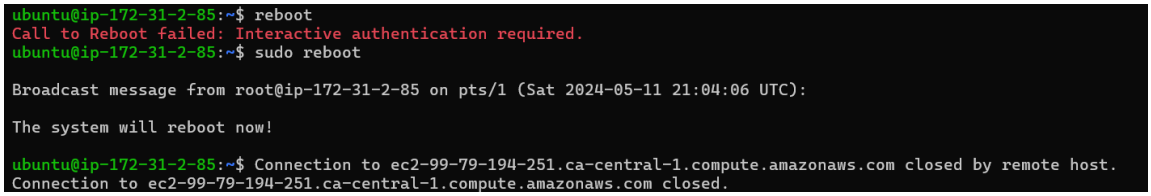
6. **which:**
    - **This command displays the full path to the executable file of a given command.**
    - **It helps to identify the location of a command's executable file in the system's PATH environment variable.**
    - **Example:** `which ls` **displays the path to the** `ls` **command.**



7. **id:**
    - **This command displays the user and group IDs of the current user or a specified user.**
    - **It provides information about the user's UID (user ID), GID (group ID), and supplementary group memberships.**
    - **Example:** `id` **displays the user and group IDs of the current user.**

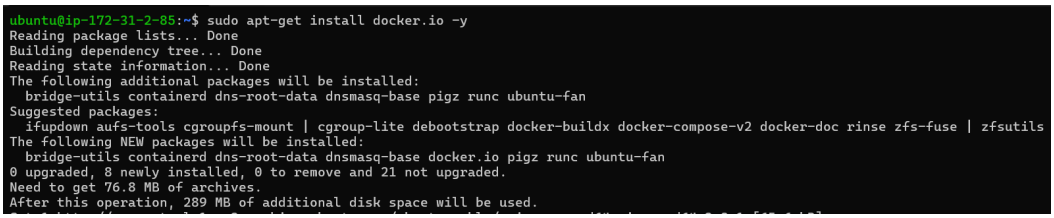
8. **sudo:**
    - **This command allows users to execute commands with the privileges of another user, typically the root user.**
    - **It is commonly used to perform administrative tasks that require elevated permissions.**
    - **Example:** `sudo apt update` **updates the package repository using elevated privileges.**
9. **shutdown and reboot:**
    - **These commands are used to shut down or reboot the system.**
    - **The** `shutdown` **command can be used to schedule a system shutdown or reboot at a specified time, and it also allows sending a message to users.**
    - **Example:** `shutdown -r now` **reboots the system immediately.**
    - **Example:** `shutdown +10 "System will be rebooting in 10 minutes"` **schedules a system reboot in 10 minutes.**

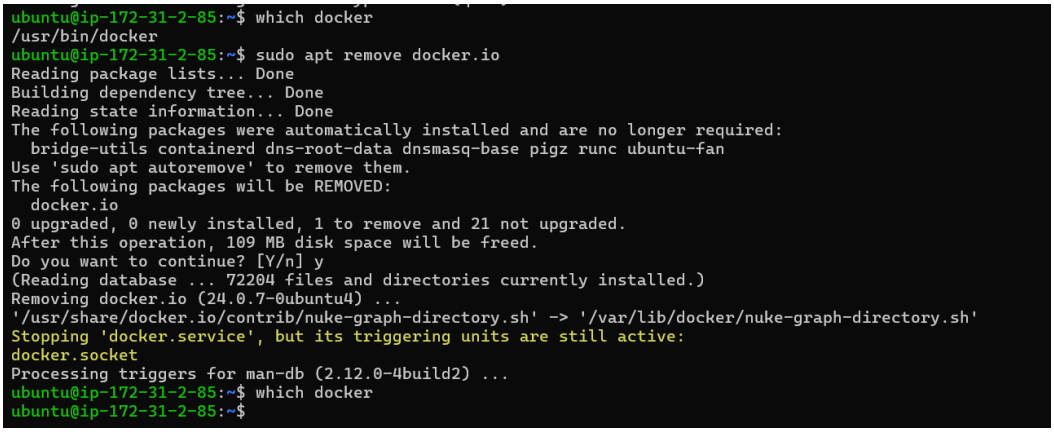




10. **apt, yum, dnf, pacman, portage:**

- **These are package management commands used in various Linux distributions to install, update, and manage software packages.**
- **They are specific to different package management systems used by different distributions:**
    - `apt`**: Used by Debian-based distributions like Ubuntu.**
    - `yum` **and** `dnf`**: Used by RPM-based distributions like Fedora and CentOS/RHEL (dnf is the newer version).**
    - `pacman`**: Used by Arch Linux and its derivatives.**
    - `portage`**: Used by Gentoo Linux.**
- **Example:** `apt install <package>` **installs a package using the APT package manager.**
  Sudo apt-get update
  Sudo apt-get install docker.io -y





11. Cat /etc/passwd:

- It will show list of users

```
ubuntu@ip-172-31-2-194:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
_apt:x:42:65534::/nonexistent:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:998:998:systemd Network Management:/:/usr/sbin/nologin
systemd-timesync:x:996:996:systemd Time Synchronization:/:/usr/sbin/nologin
dhcpcd:x:100:65534:DHCP Client Daemon,,,:/usr/lib/dhcpcd:/bin/false
messagebus:x:101:101::/nonexistent:/usr/sbin/nologin
syslog:x:102:102::/nonexistent:/usr/sbin/nologin
systemd-resolve:x:991:991:systemd Resolver:/:/usr/sbin/nologin
uuidd:x:103:103::/run/uuidd:/usr/sbin/nologin
tss:x:104:104:TPM software stack,,,:/var/lib/tpm:/bin/false
sshd:x:105:65534::/run/sshd:/usr/sbin/nologin
pollinate:x:106:1::/var/cache/pollinate:/bin/false
tcpdump:x:107:108::/nonexistent:/usr/sbin/nologin
landscape:x:108:109::/var/lib/landscape:/usr/sbin/nologin
fwupd-refresh:x:990:990:Firmware update daemon:/var/lib/fwupd:/usr/sbin/nologin
polkitd:x:989:989:User for polkitd:/:/usr/sbin/nologin
ec2-instance-connect:x:109:65534::/nonexistent:/usr/sbin/nologin
_chrony:x:110:112:Chrony daemon,,,:/var/lib/chrony:/usr/sbin/nologin
ubuntu:x:1000:1000:Ubuntu:/home/ubuntu:/bin/bash
```

These commands are fundamental tools for managing and interacting with Linux systems, performing various tasks ranging from system administration to software installation and maintenance.

User and group management commands

1. useradd:
    - This command is used to create a new user account on the system.
    - Syntax: `useradd [options] username`
    - **Common options:**
        - `-m`**: Create the user's home directory if it doesn't exist.**
        - `-s <shell>`**: Set the user's login shell.**
    - **Example:** `useradd -m john` **creates a new user account named "john" with a home directory.**

```
ubuntu@ip-172-31-2-85:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-2-85:~$ sudo useradd -m pardeep
ubuntu@ip-172-31-2-85:~$ ls
ubuntu@ip-172-31-2-85:~$ cd ..
ubuntu@ip-172-31-2-85:/home$ ls
pardeep  ubuntu
```

2. **whoami:**

- **This command prints the username of the current user.**
- **It simply returns the username associated with the current user's session.**
- **Example:** `whoami` **displays the current username.**

```
ubuntu@ip-172-31-2-85:/home$ whoami
ubuntu
ubuntu@ip-172-31-2-85:/home$
```

3. **passwd:**
   - **This command is used to change a user's password.**
   - **If run without any arguments, it prompts the current user to change their password.**
   - **Syntax:** `passwd [username]`
   - **Example:** `passwd john` **allows the user "john" to change their password.**

```
ubuntu@ip-172-31-2-85:/home$ sudo passwd pardeep
New password:
Retype new password:
passwd: password updated successfully
```

4. **su:**
   - **This command allows switching to another user account.**
   - **By default, it switches to the root user account, but you can specify another user's username as an argument.**
   - **Syntax:** `su [username]`
   - **Example:** `su pardeep` **switches to the user account "pardeep".**

```
ubuntu@ip-172-31-2-85:/home$ su pardeepp
Password:
$ whoami
pardeepp
$ id
uid=1002(pardeepp) gid=1002(pardeepp) groups=1002(pardeepp)
$ ls
pardeep  pardeepp  ubuntu
$ cd pardeepp
$ ls
$ pwd
/home/pardeepp
$ exit
ubuntu@ip-172-31-2-85:/home$
```

```
ubuntu@ip-172-31-2-85:/home$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
_apt:x:42:65534::/nonexistent:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:998:998:systemd Network Management:/:/usr/sbin/nologin
systemd-timesync:x:996:996:systemd Time Synchronization:/:/usr/sbin/nologin
dhcpcd:x:100:65534:DHCP Client Daemon,,,:/usr/lib/dhcpcd:/bin/false
messagebus:x:101:101::/nonexistent:/usr/sbin/nologin
syslog:x:102:102::/nonexistent:/usr/sbin/nologin
systemd-resolve:x:991:991:systemd Resolver:/:/usr/sbin/nologin
uuidd:x:103:103::/run/uuidd:/usr/sbin/nologin
tss:x:104:104:TPM software stack,,,:/var/lib/tpm:/bin/false
sshd:x:105:65534::/run/sshd:/usr/sbin/nologin
pollinate:x:106:1::/var/cache/pollinate:/bin/false
tcpdump:x:107:108::/nonexistent:/usr/sbin/nologin
landscape:x:108:109::/var/lib/landscape:/usr/sbin/nologin
fwupd-refresh:x:990:990:Firmware update daemon:/var/lib/fwupd:/usr/sbin/nologin
polkitd:x:989:989:User for polkitd:/:/usr/sbin/nologin
ec2-instance-connect:x:109:65534::/nonexistent:/usr/sbin/nologin
_chrony:x:110:112:Chrony daemon,,,:/var/lib/chrony:/usr/sbin/nologin
ubuntu:x:1000:1000:Ubuntu:/home/ubuntu:/bin/bash
dnsmasq:x:999:65534:dnsmasq:/var/lib/misc:/usr/sbin/nologin
pardeep:x:1001:1001::/home/pardeep:/bin/sh
pardeepp:x:1002:1002::/home/pardeepp:/bin/sh
```

5. **userdel:**
   - **This command is used to delete a user account from the system.**
   - **By default, it only removes the user's entry from the `/etc/passwd` file and does not delete their home directory or any files owned by the user.**
   - **Syntax:** `userdel [options] username`
   - **Common options:**
     - `-r`: **Remove the user's home directory and mail spool.**
   - **Example:** `userdel -r pardeep` **deletes the user account "pardeep" and removes their home directory.**

```
ubuntu@ip-172-31-2-85:/home$ sudo userdel pardeepp
ubuntu@ip-172-31-2-85:/home$ su pardeepp
su: user pardeepp does not exist or the user entry does not contain all the required fields
ubuntu@ip-172-31-2-85:/home$
```

6. **groupadd:**
   - **This command is used to create a new group on the system.**
   - **Syntax:** `groupadd [options] groupname`

- **Example:** `groupadd developers` **creates a new group named "developers".**

```
ubuntu@ip-172-31-2-85:/home$ sudo useradd kamal
ubuntu@ip-172-31-2-85:/home$ sudo useradd ranbir
ubuntu@ip-172-31-2-85:/home$ sudo useradd raman
ubuntu@ip-172-31-2-85:/home$ sudo groupadd developers
ubuntu@ip-172-31-2-85:/home$ cat /etc/passwd/groups
```

Cat /etc/group

```
ubuntu:x:1000:
docker:x:113:
pardeep:x:1001:
kamal:x:1002:
ranbir:x:1003:
raman:x:1004:
developers:x:1005:
```

7. **gpasswd:**
   - **This command is used to administer the** `/etc/group` **file and manage group membership.**
   - **Syntax:**
     - **To add a user to a group:** `gpasswd -a username groupname`
     - **To set the group's password:** `gpasswd groupname`
     - **To make a user the group's owner:** `gpasswd -m username groupname`
   - **Example:** `gpasswd -a pardeep developers` **adds the user "pardeep" to the group "developers".**

```
ubuntu@ip-172-31-2-85:/home$ sudo gpasswd --add pardeep developers
Adding user pardeep to group developers
ubuntu@ip-172-31-2-85:/home$ sudo gpasswd --add raman developers
Adding user raman to group developers
ubuntu@ip-172-31-2-85:/home$ sudo gpasswd --add ranbir tester
Adding user ranbir to group tester
ubuntu@ip-172-31-2-85:/home$ sudo gpasswd --add kamal tester
Adding user kamal to group tester
ubuntu@ip-172-31-2-85:/home$ sudo gpasswd --add kamal developers
Adding user kamal to group developers
ubuntu@ip-172-31-2-85:/home$
```

```
developers:x:1005:pardeep,raman,kamal
tester:x:1006:ranbir,kamal
ubuntu@ip-172-31-2-85:/home$
```

To add multiple users

```
ubuntu@ip-172-31-2-85:/home$ sudo gpasswd -M pardeep,kamal,raman devops
ubuntu@ip-172-31-2-85:/home$ cat /etc/group
root:x:0:
```

```
developers:x:1005:pardeep,raman,kamal
tester:x:1006:ranbir,kamal
devops:x:1007:pardeep,kamal,raman
```

8. **groupdel:**
    - **This command is used to delete a group from the system.**
    - **Syntax:** `groupdel groupname`
    - **Example:** `groupdel developers` **deletes the group "developers" from the system.**

```
ubuntu@ip-172-31-2-85:/home$ sudo groupdel tester
ubuntu@ip-172-31-2-85:/home$ cat /etc/group
root:x:0:
daemon:x:1:
```

```
raman:x:1004:
developers:x:1005:pardeep,raman,kamal
devops:x:1007:pardeep,kamal,raman
```

These commands are essential for managing user accounts, groups, and permissions on a Linux system, allowing administrators to control access to resources and perform user-related tasks efficiently.

File permission commands:

1. umask:
    - The `umask` **command sets the default file permissions for newly created files and directories.**
    - **It works by subtracting the specified file mode from the default permission settings (usually 666 for files and 777 for directories).**
    - **The result is the actual permission that is applied when a new file or directory is created.**
    - **Syntax:** `umask [mode]`
    - **Example:** `umask 022` **sets the default permission for newly created files to 644 and for directories to 755.**

```
ubuntu@ip-172-31-2-85:~$ touch newfile.txt
ubuntu@ip-172-31-2-85:~$ ls -l
total 8
drwxrwxrwx 2 ubuntu ubuntu 4096 May 11 22:07 document
-rw-rw-r-- 1 ubuntu ubuntu    0 May 11 22:07 file1.txt
-rw-rw-r-- 1 ubuntu ubuntu    0 May 11 22:40 newfile.txt
-rwxrwxrwx 1 ubuntu ubuntu   19 May 11 22:12 script.sh
ubuntu@ip-172-31-2-85:~$ umask
0002
ubuntu@ip-172-31-2-85:~$
```

2. **ls -l:**
   - **The `ls -l` command is used to list files and directories in long format, showing detailed information about permissions, ownership, size, modification date, and filename.**
   - **It displays file permissions in the format `-rwxr-xr-x`, where the first character indicates file type, and the next nine characters represent permission settings for the owner, group, and others.**
   - **Syntax: `ls -l [path]`**
   - **Example: `ls -l /home/user` lists files and directories in long format in the `/home/user` directory.**

```
ubuntu@ip-172-31-2-85:~$ mkdir document
ubuntu@ip-172-31-2-85:~$ touch file1.txt
ubuntu@ip-172-31-2-85:~$ ls
document  file1.txt
ubuntu@ip-172-31-2-85:~$ ls -a
.  ..  .bash_history  .bash_logout  .bashrc  .cache  .profile  .ssh  .sudo_as_admin_successful  document  file1.txt
```

```
ubuntu@ip-172-31-2-85:~$ ls -l
total 4
drwxrwxr-x 2 ubuntu ubuntu 4096 May 11 22:07 document
-rw-rw-r-- 1 ubuntu ubuntu    0 May 11 22:07 file1.txt
ubuntu@ip-172-31-2-85:~$
```

3. **chmod:**
   - **The `chmod` command is used to change the permissions of files and directories.**
   - **It allows you to add, remove, or set specific permissions for the owner, group, and others.**
   - **Permissions can be specified using symbolic notation (e.g., `u+r`, `g-w`, `o+x`) or octal notation (e.g., `755`, `644`).**
   - **Syntax: `chmod [options] mode file`**
   - **Example: `chmod u+x script.sh` adds execute permission for the owner to the file `script.sh`.**

```
ubuntu@ip-172-31-2-85:~$ ls -l
total 4
drwxrwxr-x 2 ubuntu ubuntu 4096 May 11 22:07 document
-rw-rw-r-- 1 ubuntu ubuntu    0 May 11 22:07 file1.txt
ubuntu@ip-172-31-2-85:~$ chmod 777 document
ubuntu@ip-172-31-2-85:~$ ls -l
total 4
drwxrwxrwx 2 ubuntu ubuntu 4096 May 11 22:07 document
-rw-rw-r-- 1 ubuntu ubuntu    0 May 11 22:07 file1.txt
ubuntu@ip-172-31-2-85:~$
```

```
ubuntu@ip-172-31-2-85:~$ vim script.sh
ubuntu@ip-172-31-2-85:~$ cat script.sh
#!/bin/bash
#hello
ubuntu@ip-172-31-2-85:~$ ls
document  file1.txt  script.sh
ubuntu@ip-172-31-2-85:~$ chmod 777 script.sh
ubuntu@ip-172-31-2-85:~$ ls
document  file1.txt  script.sh
ubuntu@ip-172-31-2-85:~$
```

```
ubuntu@ip-172-31-2-85:~$ chmod 700 script.sh
ubuntu@ip-172-31-2-85:~$ ls -l
total 8
drwxrwxrwx 2 ubuntu ubuntu 4096 May 11 22:07 document
-rw-rw-r-- 1 ubuntu ubuntu    0 May 11 22:07 file1.txt
-rwx------ 1 ubuntu ubuntu   19 May 11 22:12 script.sh
ubuntu@ip-172-31-2-85:~$ chmod 777 script.sh
ubuntu@ip-172-31-2-85:~$ ls -l
total 8
drwxrwxrwx 2 ubuntu ubuntu 4096 May 11 22:07 document
-rw-rw-r-- 1 ubuntu ubuntu    0 May 11 22:07 file1.txt
-rwxrwxrwx 1 ubuntu ubuntu   19 May 11 22:12 script.sh
ubuntu@ip-172-31-2-85:~$
```

4. **chown:**
   - **The `chown` command is used to change the ownership of files and directories.**
   - **It allows you to change both the user owner and the group owner of a file or directory.**
   - **Syntax:** `chown [options] owner[:group] file`
   - **Example:** `chown user:group file.txt` **changes the owner of the file** `file.txt` **to** `user` **and the group owner to** `group`**.**

```
ubuntu@ip-172-31-2-85:~$ touch demo.txt
ubuntu@ip-172-31-2-85:~$ ls -l
total 8
-rw-rw-r-- 1 ubuntu ubuntu    0 May 11 22:42 demo.txt
drwxrwxrwx 2 ubuntu ubuntu 4096 May 11 22:07 document
-rw-rw-r-- 1 ubuntu ubuntu    0 May 11 22:07 file1.txt
-rw-rw-r-- 1 ubuntu ubuntu    0 May 11 22:40 newfile.txt
-rwxrwxrwx 1 ubuntu ubuntu   19 May 11 22:12 script.sh
ubuntu@ip-172-31-2-85:~$ sudo chown pardeep demo.txt
ubuntu@ip-172-31-2-85:~$ ls -l
total 8
-rw-rw-r-- 1 pardeep ubuntu    0 May 11 22:42 demo.txt
drwxrwxrwx 2 ubuntu  ubuntu 4096 May 11 22:07 document
-rw-rw-r-- 1 ubuntu  ubuntu    0 May 11 22:07 file1.txt
-rw-rw-r-- 1 ubuntu  ubuntu    0 May 11 22:40 newfile.txt
-rwxrwxrwx 1 ubuntu  ubuntu   19 May 11 22:12 script.sh
ubuntu@ip-172-31-2-85:~$
```

5. **chgrp:**
   - **The `chgrp` command is used to change the group ownership of files and directories.**
   - **It allows you to change the group owner of a file or directory without changing the user owner.**
   - **Syntax:** `chgrp [options] group file`
   - **Example:** `chgrp group file.txt` **changes the group owner of the file** `file.txt` **to** `group`**.**

```
ubuntu@ip-172-31-2-85:~$ ls -l
total 8
-rw-rw-r-- 1 pardeep ubuntu    0 May 11 22:42 demo.txt
drwxrwxrwx 2 ubuntu  ubuntu 4096 May 11 22:07 document
-rw-rw-r-- 1 ubuntu  ubuntu    0 May 11 22:07 file1.txt
-rw-rw-r-- 1 ubuntu  ubuntu    0 May 11 22:40 newfile.txt
-rwxrwxrwx 1 ubuntu  ubuntu   19 May 11 22:12 script.sh
ubuntu@ip-172-31-2-85:~$ sudo chgrp devops demo.txt
ubuntu@ip-172-31-2-85:~$ ls -l
total 8
-rw-rw-r-- 1 pardeep devops    0 May 11 22:42 demo.txt
drwxrwxrwx 2 ubuntu  ubuntu 4096 May 11 22:07 document
-rw-rw-r-- 1 ubuntu  ubuntu    0 May 11 22:07 file1.txt
-rw-rw-r-- 1 ubuntu  ubuntu    0 May 11 22:40 newfile.txt
-rwxrwxrwx 1 ubuntu  ubuntu   19 May 11 22:12 script.sh
ubuntu@ip-172-31-2-85:~$
```

These file permission commands are essential for managing access control and security settings on Linux systems, allowing administrators to control who can read, write, and execute files and directories.

Compression commands:

1. zip:
    - The `zip` **command is used to compress files and directories into a ZIP archive format.**
    - **It can also be used to extract files from ZIP archives.**
    - **Syntax for compression:** `zip [options] zipfile files/directories`
    - **Syntax for extraction:** `unzip zipfile`
    - **Example for compression:** `zip -r archive.zip directory/` **compresses the contents of the "directory" into a ZIP archive named "archive.zip".**
    - **Example for extraction:** `unzip archive.zip` **extracts the contents of the ZIP archive "archive.zip" into the current directory.**

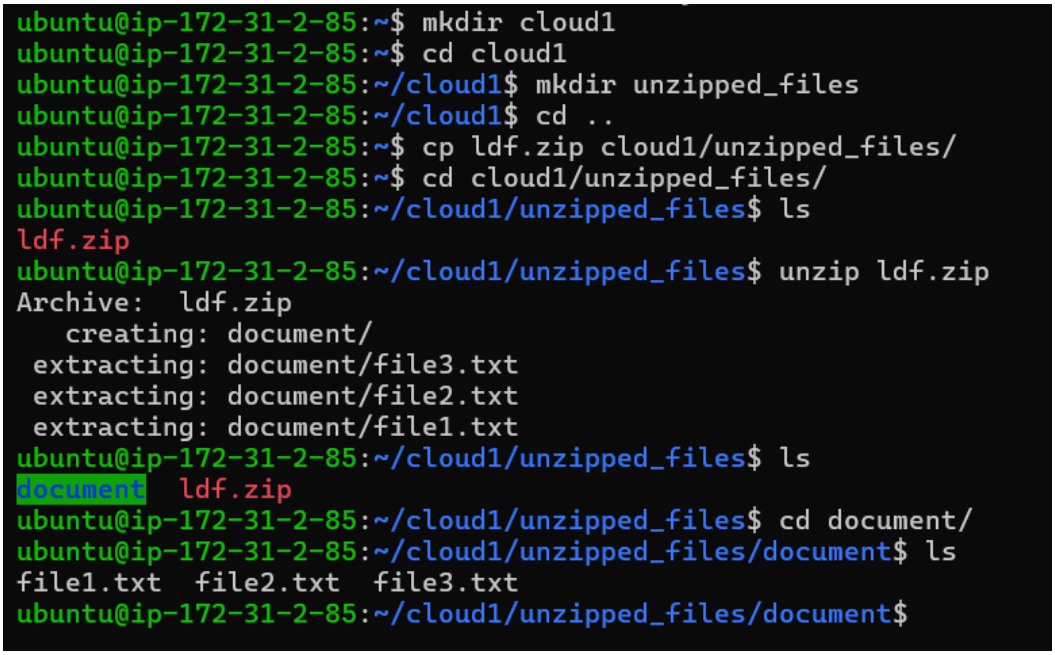
```
ubuntu@ip-172-31-2-85:~$ sudo apt install zip
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

```
ubuntu@ip-172-31-2-85:~$ ls
demo.txt  document  file1.txt  newfile.txt  script.sh
ubuntu@ip-172-31-2-85:~$ cd document/
ubuntu@ip-172-31-2-85:~/document$ touch file1.txt
ubuntu@ip-172-31-2-85:~/document$ touch file2.txt
ubuntu@ip-172-31-2-85:~/document$ touch file3.txt
ubuntu@ip-172-31-2-85:~/document$ ls
file1.txt  file2.txt  file3.txt
ubuntu@ip-172-31-2-85:~/document$ cd ..
ubuntu@ip-172-31-2-85:~$ ls
demo.txt  document  file1.txt  newfile.txt  script.sh
ubuntu@ip-172-31-2-85:~$
```

```
ubuntu@ip-172-31-2-85:~$ zip -r ldf.zip document
  adding: document/ (stored 0%)
  adding: document/file3.txt (stored 0%)
  adding: document/file2.txt (stored 0%)
  adding: document/file1.txt (stored 0%)
ubuntu@ip-172-31-2-85:~$ ls
demo.txt  document  file1.txt  ldf.zip  newfile.txt  script.sh
ubuntu@ip-172-31-2-85:~$
```

2. **Unzip, gunzip:**
    - **The** `unzip` **command is used to decompress files that have been compressed using the gzip algorithm.**
    - **It can decompress files with extensions like** `.gz`, `.z`, `.Z`.
    - **Syntax:** `unzip [options] filename`
    - **Example:** `unzip file.txt.zip` **decompresses the file "file.txt.zip" and creates a new uncompressed file "file.txt".**

```
ubuntu@ip-172-31-2-85:~$ mkdir cloud1
ubuntu@ip-172-31-2-85:~$ cd cloud1
ubuntu@ip-172-31-2-85:~/cloud1$ mkdir unzipped_files
ubuntu@ip-172-31-2-85:~/cloud1$ cd ..
ubuntu@ip-172-31-2-85:~$ cp ldf.zip cloud1/unzipped_files/
ubuntu@ip-172-31-2-85:~$ cd cloud1/unzipped_files/
ubuntu@ip-172-31-2-85:~/cloud1/unzipped_files$ ls
ldf.zip
ubuntu@ip-172-31-2-85:~/cloud1/unzipped_files$ unzip ldf.zip
Archive:  ldf.zip
   creating: document/
 extracting: document/file3.txt
 extracting: document/file2.txt
 extracting: document/file1.txt
ubuntu@ip-172-31-2-85:~/cloud1/unzipped_files$
```



3. **gzip:**
   - **The `gzip` command is used to compress files using the gzip algorithm.**
   - **It replaces the original file with a compressed version having the extension `.gz`.**
   - **Syntax: `gzip [options] filename`**
   - **Example: `gzip file.txt` compresses the file "file.txt" and creates a new compressed file "file.txt.gz".**
4. **tar:**
   - **The `tar` command is used to create, list, and extract files from tar archives.**
   - **It can be used with various compression algorithms (e.g., gzip, bzip2) to create compressed tar archives.**
   - **Syntax for compression: `tar -czvf archive.tar.gz files/directories`**
   - **Syntax for extraction: `tar -xzvf archive.tar.gz`**
   - **Example for compression: `tar -czvf archive.tar.gz directory/` creates a compressed tar archive named "archive.tar.gz" containing the contents of the "directory".**

- **Example for extraction:** `tar -xzvf archive.tar.gz` **extracts the contents of the compressed tar archive "archive.tar.gz" into the current directory.**

```
ubuntu@ip-172-31-2-85:~$ tar -cvzf cloud1.tar.gz cloud1
cloud1/
cloud1/unzipped_files/
cloud1/unzipped_files/document/
cloud1/unzipped_files/document/file3.txt
cloud1/unzipped_files/document/file2.txt
cloud1/unzipped_files/document/file1.txt
cloud1/unzipped_files/ldf.zip
ubuntu@ip-172-31-2-85:~$ ls
cloud  cloud.tar.gz  cloud1  cloud1.tar.gz  demo.txt  document  file1.txt  ldf.zip  newfile.txt  script.sh
```

5. **untar:**
   - **There's no specific "untar" command. To extract files from a tar archive, you use the** `tar` **command with appropriate options, as shown above.**

```
ubuntu@ip-172-31-2-85:~$ cp cloud1.tar.gz document/
ubuntu@ip-172-31-2-85:~$ cd document/
ubuntu@ip-172-31-2-85:~/document$ ls
cloud1.tar.gz  file1.txt.gz  file2.txt.gz  file3.txt.gz
ubuntu@ip-172-31-2-85:~/document$ tar -xvz cloud1.tar.gz
tar: Refusing to read archive contents from terminal (missing -f option?)
tar: Error is not recoverable: exiting now
ubuntu@ip-172-31-2-85:~/document$ tar -xvzf cloud1.tar.gz
cloud1/
cloud1/unzipped_files/
cloud1/unzipped_files/document/
cloud1/unzipped_files/document/file3.txt
cloud1/unzipped_files/document/file2.txt
cloud1/unzipped_files/document/file1.txt
cloud1/unzipped_files/ldf.zip
ubuntu@ip-172-31-2-85:~/document$
```

File transfer commands:

There are several commands and protocols commonly used for transferring files between systems in a Unix-like environment. Here are some of the most common ones:

1. **scp (Secure Copy):**
   - `scp` **is a command-line tool used to securely transfer files between a local and a remote host or between two remote hosts over SSH (Secure Shell).**
   - **Syntax for copying from local to remote:** `scp [options] local_file remote_user@remote_host:remote_path`
   - **Syntax for copying from remote to local:** `scp [options] remote_user@remote_host:remote_path local_path`
   - **Example:** `scp file.txt user@example.com:/remote/directory/` **copies the file "file.txt" from the local system to the remote host "example.com" into the directory "/remote/directory/".**

1. Copy from local to server

scp -i "/Users/kamal/OneDrive/Desktop/votingproject.pem" ffilee2.txt ubuntu@ec2-99-79-194-251.ca-central-1.compute.amazonaws.com:/home/ubuntu

Create file on local

```
PS C:\Users\kamal\OneDrive\Desktop> mkdir linux_for_devops


    Directory: C:\Users\kamal\OneDrive\Desktop

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         2024-05-11   7:05 PM                linux_for_devops
```

```
PS C:\Users\kamal\OneDrive\Desktop> cd .\linux_for_devops\
PS C:\Users\kamal\OneDrive\Desktop\linux_for_devops> ls
```

Now copy this file from local to server:

```
ubuntu@ip-172-31-2-85:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-2-85:~$
```

```
PS C:\Users\kamal\OneDrive\Desktop\linux_for_devops> notepad ffilee2.txt
PS C:\Users\kamal\OneDrive\Desktop\linux_for_devops> scp -i "/Users/kamal/OneDrive/Desktop/votingproject.pem" ffilee2.txt ubuntu@ec2-99-79-194-251.ca-central-1.compute.amazonaws.com:/home/ubuntu
ffilee2.txt                                                                                   100%   27     0.9KB/s   00:00
PS C:\Users\kamal\OneDrive\Desktop\linux_for_devops>
```

```
ubuntu@ip-172-31-2-85:~$ ls
cloud  cloud.tar.gz  cloud1  cloud1.tar.gz  demo.txt  document  ffilee2.txt  file1.txt  ldf.zip  newfile.txt  scpfile.txt  script.sh
ubuntu@ip-172-31-2-85:~$ cat ffilee2.txt
this secret is from pardeepubuntu@ip-172-31-2-85:~$
```

2. **Copy from server to local**

   scp -i "/Users/kamal/OneDrive/Desktop/votingproject.pem" -r ubuntu@ec2-99-79-194-251.ca-central-1.compute.amazonaws.com:/home/ubuntu/cloud1 .

```
PS C:\Users\kamal\OneDrive\Desktop\linux_for_devops> scp -i "/Users/kamal/OneDrive/Desktop/votingproject.pem" -r  ubuntu@ec2-99-79-194-251.ca-central-1.compute.amazonaws.com:/home/ubuntu/cloud1 .
file3.txt                                                                                     100%    0     0.0KB/s   00:00
file2.txt                                                                                     100%    0     0.0KB/s   00:00
file1.txt                                                                                     100%    0     0.0KB/s   00:00
ldf.zip                                                                                       100%  660    20.7KB/s   00:00
PS C:\Users\kamal\OneDrive\Desktop\linux_for_devops>
```

```
PS C:\Users\kamal\OneDrive\Desktop\linux_for_devops> ls


    Directory: C:\Users\kamal\OneDrive\Desktop\linux_for_devops

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         2024-05-11   7:24 PM                cloud1
-a---          2024-05-11   7:16 PM             27 ffilee2.txt
-a---          2024-05-11   7:07 PM          16665 scpfile.txt

PS C:\Users\kamal\OneDrive\Desktop\linux_for_devops> cd .\cloud1\
PS C:\Users\kamal\OneDrive\Desktop\linux_for_devops\cloud1> ls


    Directory: C:\Users\kamal\OneDrive\Desktop\linux_for_devops\cloud1

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         2024-05-11   7:24 PM                unzipped_files
```

2. **rsync:**
   - `rsync` **is a powerful command-line utility used for file synchronization and transfer.**
   - **It is particularly useful for synchronizing files between local and remote systems or between directories on the same system.**
   - **Syntax:** `rsync [options] source destination`

- **Example:** `rsync -avz /local/directory/ user@example.com:/remote/directory/` **synchronizes the contents of the local directory "/local/directory/" with the directory "/remote/directory/" on the remote host "example.com".**

3. **sftp (SSH File Transfer Protocol):**
   - `sftp` **is a command-line tool and interactive file transfer protocol similar to FTP, but it operates over an encrypted SSH connection.**
   - **It provides a secure way to transfer files between hosts and perform file operations such as listing directories, uploading, and downloading files.**
   - **Syntax:** `sftp [options] user@hostname`
   - **Example:** `sftp user@example.com` **starts an interactive SFTP session with the remote host "example.com".**
4. **wget and curl:**
   - `wget` **and** `curl` **are command-line tools used for downloading files from remote servers.**
   - **They support various protocols, including HTTP, HTTPS, FTP, and FTPS.**
   - **Syntax for wget:** `wget [options] URL`
   - **Syntax for curl:** `curl [options] URL`
   - **Example with wget:** `wget http://example.com/file.txt` **downloads the file "file.txt" from the URL "http://example.com/".**
   - **Example with curl:** `curl -O http://example.com/file.txt` **downloads the file "file.txt" from the URL "http://example.com/".**
5. **FTP (File Transfer Protocol):**
   - **FTP is a standard network protocol used for transferring files between a client and a server on a computer network.**
   - **It operates over TCP/IP and supports both ASCII and binary file transfer modes.**
   - **Various FTP clients such as** `ftp`, `lftp`, **and graphical clients like FileZilla are available for transferring files using FTP.**

These file transfer commands and protocols offer different features and capabilities, allowing users to transfer files securely and efficiently between systems in a Unix-like environment. The choice of command or protocol depends on factors such as security requirements, network configuration, and user preferences.