# Users and Groups

- Most beautiful feature of Linux OS is it allows multiple users accessing the system at the same time.
- Security to protect each user from other permissions

# Types of user

| TYPE | EXAMPLE | USER ID (ID) | GROUP ID (GID) | HOME DIR | SHELL |
|---|---|---|---|---|---|
| ROOT | root | 0 | 0 | /root | /bin/bash |
| REGULAR | imran, vagrant | 1000 to 60000 | 1000 to 60000 | /home/username | /bin/bash |
| SERVICE | ftp, ssh, apache | 1 to 999 | 1 to 999 | /var/ftp etc | /sbin/nologin |

**3 types of users in linux**

1. Superuser or root user

- Super user or the root user is the most powerful user. administrator user

2. System user

- System users are created by the software or application, e.g when we install apache, it creates user apache.

3. Normal user

- Normal users are the users created by root user, only root has permission to create or remove the user.

**Whenever a user is created in linux.**

- A home directory is created(/home/username)
- A mailbox is created(/var/spool/mail)

- unique UID & GID are given to user

# Passwd file

**1. /etc/passwd**

```
root@ubuntu-focal:/home/vagrant# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
```

- root = name
- x = link to password file i.e /etc/shadow
- 0 or 1 = UID (user id)
- 0 or 1 = GID (group id)
- root or bin = comment (brief information about user)
- /root or /bin = home directory of the user
- /bin/bash or /sbin/nologin = shell

# Group file

**2. /etc/group**

The file /etc/group stores group information. Each line stores one group entry.

**Group name, group password, GID, group members**

```
root@ubuntu-focal:/home/vagrant# cat /etc/group
root:x:0:
daemon:x:1:
```

# 3. /etc/shadow file

This file stores users' password and password related information. Just like /etc/passwd file, this file also uses an individual line for each entry.

1. Username

2. Encrypted password

3. Number of days when password was last changed

4. Number of days before password can be changed

5. Number of days after password must be changed

6. Number of days before password expiry date to display the warning message

7. Number of days to disable the account after the password expiry

8. Number of days since the account is disabled

9. Reserved field

root@ubuntu-focal:/home/vagrant# cat /etc/shadow

root:*:19229:0:99999:7:::

daemon:*:19229:0:99999:7:::

bin:*:19229:0:99999:7:::

sys:*:19229:0:99999:7:::

sync:*:19229:0:99999:7:::

# Add user

adduser {username}

→ For ubuntu

Or

Useradd with non-default parameters.

• Syntax:[useradd <option> <username>]

Options : -c //comment

-d //home directory

-m //create home directory

-e //account expiry date

-g //primary group

-G //secondary group -s //shell

-u //user id

E.g

# groupadd staff

# groupadd hr

# groupadd admin

# useradd -c "Amrit Tamang" -m -d /var/amrit -e 2025-10-01 -g staff -G hr,admin -s /bin/bash -u 2000 amrit

# grep surya /etc/passwd

# groups surya

**To modify user's account**

# usermod [option] <username>

-c <new comment>

-d <new home dir>

-e <new A/C expiry date>

-g <new primary group>

-G <new secondary group>

-s <new shell>

-u <new uid>

- L <To lock the user's A/c>

-U <To unlock the user's locked A/c>

# usermod -e 2025-10-02 -s /bin/sh -u 3000 amrit

# grep surya /etc/passwd

# usermod -L surya

# usermod -U surya

To change the user's login name

# usermod -l suryaraj surya

# cat /etc/passwd

**To check account expire information**

Sudo chage -l {username}

**To check details**

Id {username}

**Userdel**

Syntax:[userdel <option> <username>]

Options: -r

#userdel surya

#userdel -r surya

**Defining password Aging policies**

# chage [option] <username>

-l <To list password and account expiry information>

-m <Day to set minimum no. of days between password change>

-M <Day to set maximum no. of days between password change>

-W <Day to set no. of days to alert for password renewal before password expires>

-I <Day to set grace period before locking account after the password has expired>

-E <Date to set A/c expiry date>

# chage -l surya

# chage -m 1 -M 14 -W 2 -I 3 -E 2025-12-06 surya

# chage -l surya

**Groupadd Groupmod Groupdel**

Syntax:[groupadd <option> <group_name>]

Option: -g

# groupadd staff

# groupadd devops

# groupadd system

# cat /etc/group

# groupmod -n itops devops

# cat /etc/group

# useradd -G staff sti

# cat /etc/passwd

# id sti

# cat /etc/group

# useradd -G staff rabin

# cat /etc/group | grep staff

Group change can be done by editing /etc/group file as well.

# Changing the Default useradd Values

useradd -D

—> Prints the default values

Config file

/etc/default/useradd

Last

→ Gives user who logged into the system

Who

→ Currently logged in user

Lsof

→ Gives all the open files by the user

# Permissions

**Read:** Allows to view the contents of a file and allows to list contents of a directory.

**Write:** Allows to modify contents of a file and allows create and remove, editing contents of a directory.

**Execute:** Allows to run a file or execute a program/ script and allows you to change directory and make it your working directory, allows long listing.

**-:** No permission

| Octal Value | Read | Write | Execute |
|---|---|---|---|
| 7 | r | w | x |
| 6 | r | w | - |
| 5 | r | - | x |
| 4 | r | - | - |
| 3 | - | w | x |
| 2 | - | w | - |
| 1 | - | - | x |
| 0 | - | - | - |

Changing permissions

• Syntax chmod [permission_type] [File/Directories]
-> For recursive add -R

• # chmod 644 testfile1 //only superuser can change the permission

• # chmod -R 755 /home/srtimsina/testdir1
• # chmod  o-x  /home/srtimsina/testdir1
• # chmod  u+x /home/srtimsina/testdir1

Managing file ownership

Changing file/ directory ownership with chown

• chown //change ownership

• Syntax chown [Username] [File/Directories]

• # chown srtimsina testfile1 //only superuser can change the ownership

• # chown -R srtimsina /home/srtimsina/testdir1

## Managing file group ownership

Changing file/ directory group ownership with chgrp

• chgrp //change group ownership

• Syntax [Group name] [File/Directories]

• # chgrp testgroup1 testfile1

//only superuser can change the ownership

• # chgrp -R testgroup1 testdir1

## Managing file and group ownership

Changing file and group ownership simultaneously.

• # chown apache:apache /data/index.php

//chown command can change user-owner and group-owner of a file simultaneously.

• # chown -R apache:apache /data/srtimsina.com.np

• # chown apache.apache /data/srtimina.com.np

// a dot (.) can be used instead of colon (:).

# Special permission

**Sticky-bit**
If a sticky bit is set on a directory then only the owner can delete their own file/directory.

# groupadd admin
# groupadd itops

# adduser surya
# adduser rabindra
# adduser pankaj

# usermod -aG admin surya
# usermod -aG admin rabindra
# usermod -aG admin pankaj

# ls -ld /root/
# mkdir shareddir

# ls -ld shareddir/
# chgrp admin shareddir
# ls -ld shareddir/
# chmod g+w shareddir
# ls -ld shareddir/
# ls -ld /root/
# chmod o+rwx /root/
# su - surya
# cd
--> With user surya, create some files and dir in /root/shareddir
# su - rabindra
--> With user rabindra, try to remove the files and dirs created by user surya.

# ls -ld /root/shareddir/
--> Now switch to user root and set sticky bit on the directory
# chmod 1775 shareddir
# ls -ld shareddir/
# su - surya
--> Again create files and dirs
# su - rabindra
--> Again try to remove files and dirs, this time unable to delete.
# su - surya

**SUID-bit**
SUID-bit, setuid bit let's to run commands on the behalf of other users.
# which ls
# ls -l /usr/bin/ls
# chmod u+s /usr/bin/ls
# ls -l /usr/bin/ls
# chmod u-s /usr/bin/ls

**SGID-bit**
If SGID-bit is set on a directory, then any files/directory created inside that directory will inherit group ownership of that directory.
# ls -al tarexample/
# chmod g+s tarexample
# ls -al tarexample/
# chmod g-s tarexample

# Archiving and Compressing

- Tar
- Gzip
- Zip2
- zip

# Creating Archive

tar -cvf [new_file_name].tar [target_directory]
• Extracting Archive
tar -xvf [target_file_name].tar

# Compressing

• tar -zcvf testdir1.tar.gz testdir1
• tar -zxvf tarfile.tar.gz
• zip -r [new_filename].zip [target_dir]
• zip -r testdir2.zip testdir2
• unzip testdir2.zip

# sudo

→ sudo gives the power to a normal user to execute commands which are owned by the root user.

sudo -i
→ changes from normal to root user.

visudo
username ALL=(ALL) ALL

→ *surya ALL=(ALL) NOPASSWD: /usr/sbin/useradd,/usr/bin/passwd,!/usr/sbin/fdisk*

NOPASSWD
Username ALL=(ALL) NOPASSWD: ALL

**Safer way is adding entry in dir**
/etc/sudoers.d/
*root@ubuntu-focal:/home/vagrant# cat /etc/sudoers.d/customsudo*
*User_Alias TRUSTED=rabindra,pankaj*
*Cmnd_Alias LIMITED=/usr/sbin/useradd,/usr/bin/passwd*
*TRUSTED ALL=ALL,!LIMITED*

**For groups**
%groupName ALL=(ALL) NOPASSWD:ALL

***root ALL=(ALL:ALL) ALL***
*The first field indicates the username that the rule will apply to (root).*
*First "ALL" indicates that this rule applies to all hosts.*
*Second "ALL" indicates that the root user can run commands as all users.*
*Third "ALL" indicates that the root user can run commands as all groups.*
*Fourth, "ALL" indicates these rules apply to all commands.*

# Services

apt-get install apache2

Command Syntax [service] [service_name] [command]
///etc/init.d
or

[systemctl] [command] [service_name]
///lib/systemd
#service apache2 status
#systemctl status apache2
Commands can be, start, restart, reload, status, stop

systemctl is-active apache2
systemctl is-enabled apache2
systemctl enable apache2

Configuration file
systemctl works based on the configuration file.
cat /etc/systemd/system/multi-user.target.wants/apache2.service

# Processes

**top**
→ shows all the dynamic processes based on their memory & cpu consumption. It is similar to the task manager in windows.

**CPU Load average**
System load for the last 1, 5, and 15 minutes.
Values below 1.0 mean the CPU is not overloaded.
Higher values indicate CPU congestion.

**Task information**
total → Total number of processes.
running → Number of currently active processes.
sleeping → Processes waiting for input/output.
stopped → Processes stopped manually.
zombie → Zombie processes (terminated but still in the process table).

**CPU Usage**

us (User CPU) → CPU time spent on user processes.
sy (System CPU) → CPU time spent on kernel processes.
ni (Nice CPU) → CPU used by processes with a nice priority.
id (Idle CPU) → Idle CPU (higher is better).
wa (Wait I/O) → CPU waiting for I/O (disk/network operations).
hi (Hardware IRQ) → CPU used by hardware interrupts.
si (Software IRQ) → CPU used by software interrupts.
st (Steal Time) → CPU stolen by hypervisor in virtualized environments.

**Memory Usage**
total → Total RAM.
free → Unused RAM.
used → RAM actively used by processes.
buff/cache → Memory used for buffers and cache.

**Process Table**
PID → Process ID.
USER → Owner of the process.
PR (Priority) → Process scheduling priority.
NI (Nice Value) → Process priority adjustment (-20 to 19, lower is higher priority).
VIRT (Virtual Memory) → Total memory used (includes swap).
RES (Resident Memory) → Actual memory used (without swap).
SHR (Shared Memory) → Memory shared with other processes.
S (State):
R → Running
S → Sleeping
T → Stopped
Z → Zombie
%CPU → CPU usage.
%MEM → Memory usage.
TIME+ → Total CPU time used.

COMMAND → Process name.

**Using TOP**

press

q → Exit top.

h → Display help menu.

k → Kill a process by PID.

M → Sort by memory usage.

P → Sort by CPU usage.

T → Sort by process run time.

Shift + R → Reverse sort order.

**Process status(ps)**

The ps command in Linux is used to display information about active processes. It provides details such as process IDs (PIDs), CPU usage, memory consumption, and other key attributes of running processes

**ps aux**

→ displays processes on the screen and quits.

→ view all running processes

→ process in [ ] are called kernel threads

a - Show processes of all users

u - Displays user-oriented format

x - List processes without a controlling terminal

**ps -ef**

→ displays not the utilization but also the parent process id

→ displays process hierarchy

e - show all processes

f - display full format(more details)

**ps -ef | grep apache2**

→ Filter by process name

**ps -u username**
→ Displays processes owned by a specific user

**ps -T -p <pid>**
→ Show threads of a process
T - display threads
p - process id

**ps aux --sort=-%mem**
→ Sorts by memory usage (highest first)

**ps aux --sort=-%cpu**
→ Sorts by CPU usage (highest first).

**Kill process**

kill (pid)
Kill -9 (pid)
→ kill forcefully

→ ps -ef | grep apache2 | grep -v 'grep' | awk "{print $2}" | xargs kill -9

**Zombie:**
Processes whose operations are done but their entry is still in the process table.

**Orphan:**
A child process that remains running even after it;s parent process is terminated or completed without waiting for the child process execution.