1. User, group & Permissions

- User Managment : Useradd, userdel, Usermod

What is user in Linux :-

In Linux :

- Every person who use the linux system = User
- Even the system itself users (for services)

Example of users :

- gautam → you
- root → superuser (admin)
- www-data, mysql → system users

Linux seperates user to keep the system secure.

Why Linux need user mongment ?

- Imagine :
  - One system
  - 10 people using it

Linux must decide :

- Who can read files ?
- Who can delete file ?
- Who can install software ?

Thats why users + permisions exit.

Where Linux stores user informations ?

Linux stores users in a file :

/etc/Passwd

You do not edit this file manually

Linux command do itself

You can view it :

cat /etc/passwd

Each line = one user

1. Useradd → Create a new user

Meaning useradd

" Add a new user to the system "

- Why use sudo ?

Because :

- Creating user affects the whole system
- only admin (root) can do it

Basic Command

sudo useradd testuse

What happen internally :

- User test user is created
- Entry added to /etc/passwd
- No home directory
- No password, so user exist, but cannot login in

Correct way

Sudo useradd -m testuser

- useradd → create user
- -m → make home directory
- testuser → username

Now linux creates:

/home/testuse

Set password (very Important)

Sudo passwd testuser

Now,

- user can log in
- User fully active

Check user details

Id testuser

Output explains:

- UID (user ID)
- GID (Group ID)
- Groups user belong to

2. userdel → Delete User

What userdel does

- Removes a user account

Basic delete

Sudo userdel testuser

User deleted

Home directory remains present.

Delete user + Home directory (Important)

sudo userdel -r testuser

- -r → remove home directory also.

3. usermod → Modify an existing user

What usermod does ?

- Change user properties.

Change username

sudo usermod -l newname oldname

Change home directory

sudo usermod -d /home/newname -m newname

- -d → new home directory
- -m → move files

Add user to a group ***

sudo usermod -aG sudo devuser (any user)

- -a → append (don't remove others groups)
- -G → group name

This give sudo access.

(ii) Group Managment in Linux

1. What is a Group ~~Managment in~~ in Linux

A group in linux is a collection of users.

Linux uses group to:

- manage permissions easily
- allow ~~group users~~ multiple users to access same files

→ Instead to giving permission to each group user.
, Linux gives permission to a group.

2. Why group needed ?

Without groups:

- permission must be given user by user
- mangment becomes difficult.

With groups:

- users are added to a group
- permissions are given to the group
- all user in the group get access.

→ Group simplify permission managment.

3. Type of groups in linux

(a) primary Group

- Every users has one primary group
- It ~~groups~~ is created automatically when a user is created.
- Folles created by users belong to the primary group.

`id username` → check primary group.

(b) Secondary group (supplementary)

- A user can belong to multiple secondary groups.
- Used to give additional permissions.

Check groups :

`groups username`

4. Where Group Informations is stored ?

`/etc/group`

view groups :

`cat /etc/group`

Each line contains :

group_name : password : GID : members

Example :

sudo : x : 27 : gautam.

5. groupadd Command

groupadd is used to create a new group in Linux.

only root or sudo user can create groups.

sudo groupadd groupname

ex:

sudo groupadd developers

⑥ Adding Users to a Group

This is done using usermod, not groupmod.

Syntax:

sudo usermod -aG groupname username

ex:

sudo usermod -aG developers gautam

⑦ Groupdel Command

groupdel is used to delete an existing group.

Syntax:

sudo groupdel groupname

(ii) Ownership in Linux

Chown and chgrp

What is Ownership in Linux?

Every file and directory in Linux has 3 things

1. owner (user)
2. Group
3. Permissions

Ex:

ls -l file.txt

-rw-r--r-- 1 gautam developers 120 Jan 10 10:00 file.txt

Break it slowly:

gautam — owner (user)

developers — Group owners

So,

- user gautam owns this file
- Group developers owns the file

Why ownership matters

Ownership decides:

- who change permission
- who can delete or modify files
- who control access.

Permissions works on ownership

Without ownership, permission are meaningless.

1. Chown - Change Owners (users)

Meaning of chown

is change owner of a file or directory

Only

root / sudo can do this

Sytax:

sudo chown newuser file.txt

Check:

ls -l file.txt

Change owner of directory (and everything inside)

sudo chown -R testuser mydir

• -R → recursive (inside all files / folders)

2. Chgrp — Change Group

Meaning of chgrp

Changing group ownership of a file

Sytax:

sudo chgrp groupname file.txt

Change group recursively

sudo chgr -R developers mydir

(ii) chown can change both user and group (★)

Instead using chgrp, you can do both with chown

Sytax

sudo chown user:group file.txt

(iii) Special permission in Linux

SUID, GID, sticky Bit

These are called special permission bits

Before, this, you learned:

- r → read
- w → write
- x → execute

No we added special behaviours on top of them

- Why special permission exit?

Normal permissions answer:

- Who can read?
- Who can write?
- Who can execute?

Special permissions answers:

- Who does the program run as?
- Who controls files in shared folders?

1. SUID - Set User ID

- What SUID Means ?

- When a user a file,
the program run with the file owner's permission,
not user's permission.

- Why SUID Needed ?

Ex: passwd command

you run:

passwd

what does it do ?

- Modifies /etc/shadow
- /etc/shadow is owned by root

But you are NOT root

Then how does it work ?

Because passwd has SUID set

How to identify SUID

ls -l /usr/bin/passwd

you see:

-rwsr-xr-x

Notce:

- s istead of x in owner section

That s = SUID

Set SUID (for learning only)

Chmod u+s file

Numric:

Chomod 4755 file

Note:

SUID allows a programs to run with the file owner's privitedge.

2. SGID — Set Group ID

SGID work like SUID, but for groups.

SGID on files

When a file is executed,
it run with the group owner's permission.

SGID on directories

Files created inside the directory
automatically belong to directory's group.

This very useful for shared folders.

Identify SGID

ls -ld shared_dir

you see:

drwxrwsr-x

Notice:

's' in group position

Set SGID

Chmod g+s dir

Numeric:

Chmod 2755 dir

Note:

SGID ensures group ownership consitency in shared directory

3. Sticky bit

What is stiky bit means

In a shared directory,
only the file owner can delete their file,
even if others have write permission.

Why sticky bit needed

Example : /tmp

Everyone can write in /tmp.

Without tmp sticky Bit :

- Anyone could delete anyone else files

With sticky Bit :

- only owners can delete their own files

Identify sticky Bit :

ls -ld /tmp

O/p :

drwx rwx rwt

That t at the end = sticky bit

Set sticky bit

Chmod +t dir

Numric :

Chmod 1777 dir

Note!

Sticky Bit protect files in shared directory from being deleted by others.