**Shri Ramdeobaba College of Engineering and Management, Nagpur**

**Department of Computer Science and Engineering**

**Session: 2024-2025**

**Fundamentals of Linux OS I Semester**

**PRACTICAL NO. 3**

**Aim: Explore and Execute User and Group Management in Linux.**

**Theory:**

In Linux, user and group management is fundamental for administering and securing a multi-user environment. By organizing users into groups, administrators can enforce permissions and access control, ensuring that users have appropriate access to files, directories, and other system resources.

#### **Understanding Users in Linux**

* **User Accounts:** Each user in Linux has a unique account, which is represented by a user name (UID) and associated with a unique numeric identifier. Users also have a home directory and a default shell.
* **Types of Users:**
  + **Root User:** The superuser with unrestricted access to the entire system. The root user can perform any operation, including system-wide configuration changes and administrative tasks.
  + **Regular Users:** These are standard user accounts created for day-to-day tasks. They have limited permissions and typically only have access to their home directories and files they own.
  + **System Users:** These accounts are used by system services and daemons. They usually do not have login privileges.
* **User Account Files:**
  + /etc/passwd: Contains user account information such as username, UID, GID, home directory, and default shell.
  + /etc/shadow: Stores password information and password aging policies in an encrypted format.
  + /etc/login.defs: Defines system-wide settings related to user accounts.

#### **Understanding Groups in Linux**

* **Group Accounts:** A group is a collection of users. Each group has a unique name (GID) and associated with a numeric identifier. Groups are used to manage permissions and access controls collectively rather than individually.
* **Types of Groups:**
  + **Primary Group:** Every user is associated with a primary group, typically matching the user’s name. Files created by the user are automatically assigned to this group.
  + **Secondary Groups:** Users can be members of additional groups, granting them access to files and resources shared by those groups.
* **Group Management Files:**
  + /etc/group: Contains group account information such as group name, GID, and group members.
  + /etc/gshadow: Stores encrypted group passwords and administrative settings.

# Commands

1. **sudo command -** Runs a specified command with elevated (root) privileges, allowing administrative tasks.
2. **Useradd** - This command creates a new user account

**Example:**

## Add a User in Linux

sudo useradd test\_user

## Add User by Specifying a home directory path

sudo useradd -d /home/test\_user test\_user

## Create a User with a Specific User ID (UID)

sudo useradd -u 1234 test\_user

## Create a User with a Specific Group ID (GID)

sudo useradd -g 1000 test\_user

## **Create a User Without a Home Directory**

sudo useradd -M test\_user

## Create User with an Expiry Date

sudo useradd -e 2020-05-30 test\_user

## Create a User with Changed Login Shell

sudo useradd -s /bin/sh test\_user

## Displaying Help

useradd --help

Creates a new user account, set up a home directory, and assigns a shell.

sudo useradd -m -s /bin/bash username

1. **Passwd** - Assigns or changes the password for the specified user.

**sudo passwd username**

1. **Userdel** - Removes a user account and deletes their home directory if the -r option is used.

**sudo userdel -r username**

1. **Groupadd** - Creates a new group in the system for managing user permissions.

**sudo groupadd groupname**

1. **Usermod** - Adds an existing user to a specified group, allowing additional permissions.

**sudo usermod -aG groupname username**

1. **Groupmod** - Renames an existing group, updating its name across the system.

**sudo groupmod -n newgroupname oldgroupname**

1. **Groupdel** - Deletes an existing group from the system.

**sudo groupdel groupname**

1. **chmod 755 filename** - Sets read, write, and execute permissions for the owner, and read and execute permissions for the group and others.
2. **sudo chown owner:group filename -** Changes the owner and group of a file or directory, adjusting access rights.
3. **sudo chgrp groupname filename -** Updates the group ownership of a file or directory, assigning it to a different group.
4. **cut -d: -f1 /etc/passwd -** Lists all user accounts on the system by extracting usernames from the /etc/passwd file.
5. **cut -d: -f1 /etc/group -** Displays all group names on the system by extracting them from the /etc/group file.
6. **id username -** Provides detailed information about a user, including their UID, GID, and group memberships.
7. **groups username -** Lists all the groups that the specified user is a member of, showing their permissions and access.

**Understanding File permissions in Linux**

The ls command along with its -l (for long listing) option will show you metadata about your Linux files, including the permissions set on the file.

$ ls -l

drwxr-xr-x. 4 root root 68 Jun 13 20:25 tuned

-rw-r--r--. 1 root root 4017 Feb 24 2022 vimrc

In this example, you see two different listings. The first field of the ls -l output is a group of metadata that includes the permissions on each file. Here are the components of the vimrc listing:

* File type: -
* Permission settings: rw-r--r--
* Extended attributes: dot (.)
* User owner: root
* Group owner: root

**EXPERIMENTATION**

### **Part 1: User Management**

1. Create a new user named student with a home directory and set their default shell to /bin/bash?
2. Set the password for the student user to password123?
3. Add the student user to the sudo group?
4. Display detailed information about the student user, including their UID, GID, and group memberships?
5. Remove the student user and delete their home directory?
6. Create a User named rbu with user id as 123, group id as 1003, having home directory /home/rbu/rbu, login shell as /bin/sh and password as rbu using a single command.

### **Part 2: Group Management**

1. Create a new group named developers?
2. Create a new user named devuser and add them to the developers group?
3. Which command would you use to verify if devuser is a member of the developers group?
4. Rename the developers group to devteam?

**Part 3: Ownership**

1. Create a file “file1.txt” in user rbu
2. Create a User named rbu2 without password
3. Change the owner of file1.txt to rbu2 and group to devteam