# SHAIK. RAHILA – 192311013

1. Illustrate the various File Access Permission and different types of users in Linux.

# Aim

To understand and demonstrate file access permissions and the types of users in Linux.

# File Access Permissions

In Linux, file permissions define how files and directories are accessed by users. These permissions are represented as:

* + **Read (r)**: Allows viewing the content of a file or directory.
  + **Write (w)**: Allows modifying the content of a file or adding/deleting files in a directory.
  + **Execute (x)**: Allows running a file as a program or accessing a directory.

# Permission Categories

1. **Owner (u)**: The user who owns the file.
2. **Group (g)**: A group of users with shared access.
3. **Others (o)**: All other users on the system.

Permissions are displayed using the **ls -l** command, where: diff

Copy code

-rwxr-xr--

* + **First character**: File type (- for a file, d for a directory).
  + **Next 3 characters**: Permissions for the owner (e.g., rwx).
  + **Next 3 characters**: Permissions for the group (e.g., r-x).
  + **Last 3 characters**: Permissions for others (e.g., r--).

# Algorithm

1. Open a terminal and create a file/directory using touch or mkdir.
2. Check the current permissions using the ls -l command.
3. Modify permissions using the chmod command.
   * chmod [permissions] [filename]
   * Permissions can be set symbolically (u, g, o) or numerically (e.g., 777).
4. Validate the changes by checking permissions again with ls -l.

# Code

Below is an example script that demonstrates file permission changes: bash

Copy code #!/bin/bash

# Step 1: Create a file

echo "Creating a file named 'example.txt'..." touch example.txt

# Step 2: Display default permissions

echo "Default permissions for 'example.txt':" ls -l example.txt

# Step 3: Modify permissions to give full access to the owner, read/execute for group, and no access to others

chmod u=rwx,g=rx,o= example.txt

echo "Modified permissions for 'example.txt':" ls -l example.txt

# Step 4: Modify permissions numerically to 777 (full access for everyone) chmod 777 example.txt

echo "Permissions after setting to 777:" ls -l example.txt

# Clean up

rm example.txt

echo "File 'example.txt' deleted."

# Output

When the above script is executed, the output will resemble the following: plaintext

Copy code

Creating a file named 'example.txt'... Default permissions for 'example.txt':

-rw-r--r-- 1 user user 0 Dec 16 14:35 example.txt Modified permissions for 'example.txt':

-rwxr-x--- 1 user user 0 Dec 16 14:35 example.txt Permissions after setting to 777:

-rwxrwxrwx 1 user user 0 Dec 16 14:35 example.txt File 'example.txt' deleted.

# Result

* Demonstrated the default file permissions in Linux.
* Successfully modified file permissions using both symbolic and numeric modes.
* Observed how permissions affect access for the owner, group, and others.