# RedHat Enterprise Linux Essential

## Unit 4: Users, Groups, and Permissions

# Objectives

- Upon completion of this unit, you should be able to:
  - Explain the Linux security model
  - Explain the purpose of user and group accounts
  - Read and set file permissions

# Users

- Add user : **useradd *student* ; passwd *student***
- Every user of the system is assigned with a unique User ID number (the uid)
  - *UID 0 identifies root*
- Users' names and uids are stored in */etc/passwd*
- Users are assigned to a home directory and a program that is run when they log in (usually a *shell*)
- Users cannot read, write or excute each others' files without permission

# Groups

- Users are assigned to groups with unique group ID numbers (the *gid*)
- *gids* are stored in */etc/group*
- Each user is given their own private group
  - They can also be added to other groups to gain additional access
  - The primary group can temporarily be changed by running:

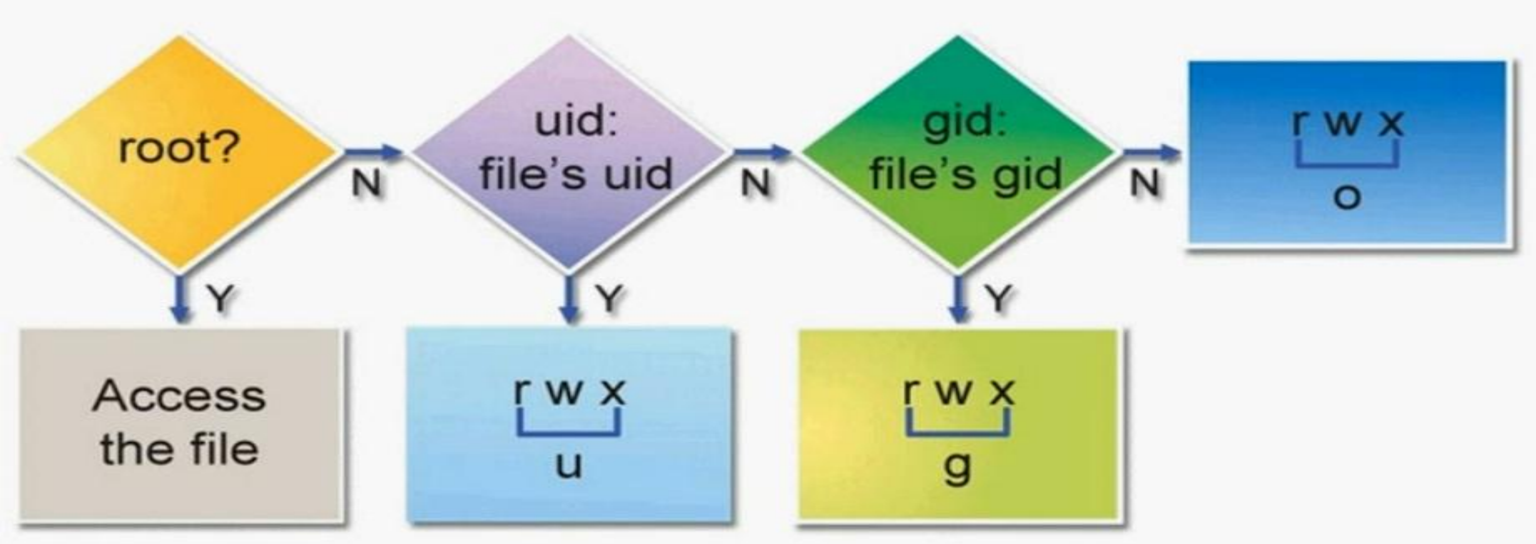
    **newgrp *groupname***
- All users in a group can share files that belong to the group

# Linux File Security

- Every file is owned by a UID and a GID
- Every process runs as a UID and one or more GIDs
  - Usually determined by who runs the process
- Three access categories:
  - Processes running with the same UID as the file (*user*)
  - Processes running with the same GID as the file (*group*)
  - All other processes (*other*)

# Processing Linux Security

- When a process accesses a file the user and group of the process are compared with the user and group of the file
  - If the user matches the user permissions apply
  - If the group matches, but the user doesn't, the group permissions apply
  - If neither match, the other permissions apply



# Permission Types

- Four symbols are used when displaying permissions:
  - **r** permission to read a file or list a directory's contents (**ls** )
  - **W** permission to write to a file or create and remove files from a directory (**touch, rm**)
  - **x** permission to excute a program or change into a directory and do along listing of the directory (**cd && ls –l**)
  - **-** no permission (in place of the r, w, or x)

# Examining Permissions

- File permissions may be viewed using **ls –l**

```
$ ls -l /bin/login

-rwxr-xr-x 1 root root 19080 Apr 1 18:26 /bin/login
```

- File type and file access permissions are symbolized by a 10 character string

# Interpreting Permissions

- Characters 2,3 and 4 identify permissions for owner
- Characters 5,6 and 7 identify permissions for members of the group
- Characters 8,9 and 10 identify permissions for all other

# Changing File Ownership

- Only root can change a file's owner
- Only root or the owner can change a file's group
- Ownership is changed with **chown**:

```
chown [-R] user_name file|directory
```

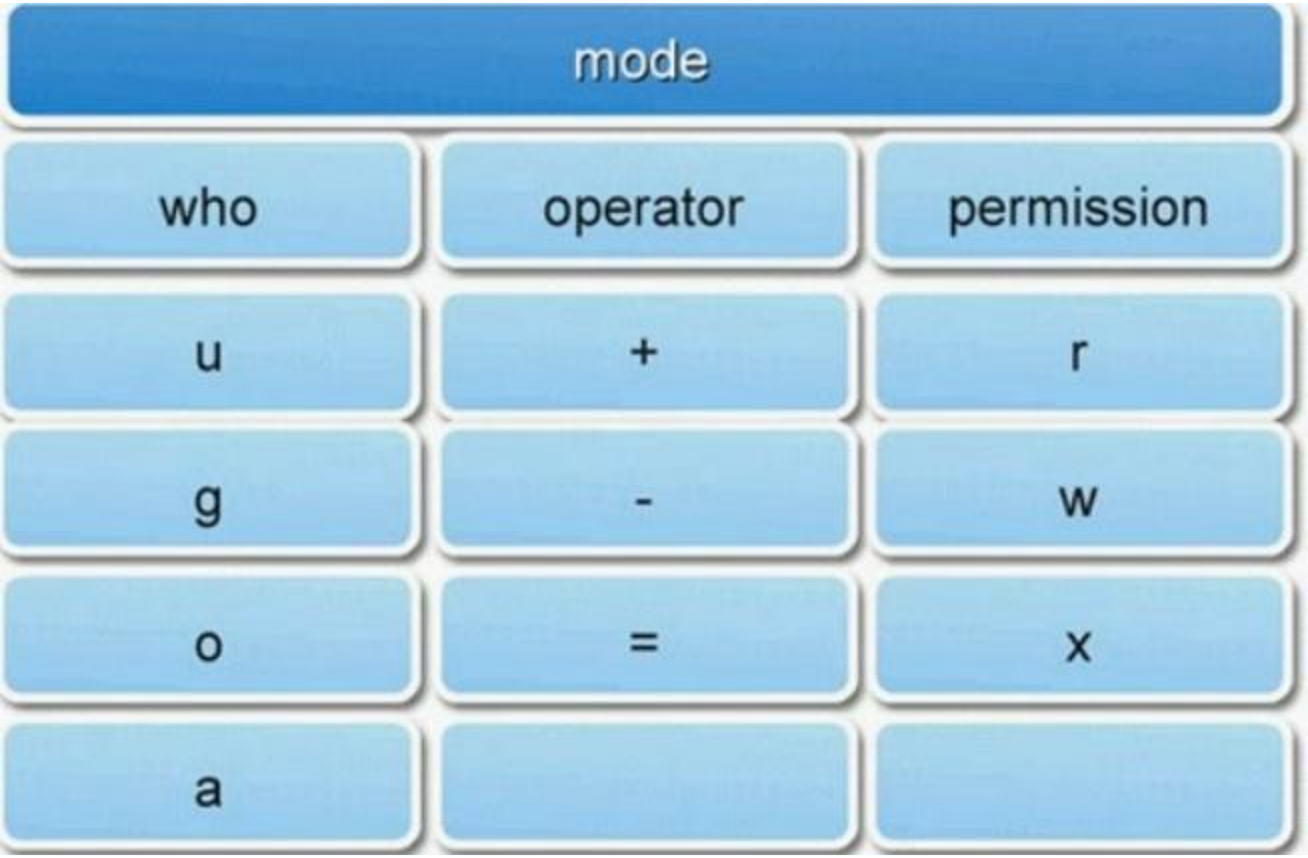
- Group-Ownership is changed with **chgrp**:

```
chgrp [-R] group_name file|directory
```

# Changing Permissions - Symbolic Method

- To change access modes:

```
chmod [-R] mode file
```



| mode | | |
|---|---|---|
| who | operator | permission |
| u | + | r |
| g | - | w |
| o | = | x |
| a | | |

# Changing Permissions- Numeric Method

- Uses a three-digit mode number
  - First digit specifies owner's permissions
  - Second digit specifies group permissions
  - Third digit represents others' permissions
- Permissions are calculated by adding:
  - **4** (for read)
  - **2** (for write)
  - **1** (for execute)
  - **0** (for no permission)
- **Example: chmod 640 myfile**

# Thank You !