GETTING STARTED
WITH
USERS & GROUPS

➢ MANAGING USERS:

- The control of users and groups is a core element of **RHEL** system administration.
- Each RHEL user has distinct login credentials and can be assigned to various groups to customize their system privileges.
- Each user and group have a unique number called a **userid (UID)** and Each group is associated with a **group ID (GID),** respectively.
- RHEL reserves **UID & GIDs** below **1000** for system users & groups, above **1000** for normal users & Groups. The root user id value is always
- To view reserved user and group IDs, use:

```
#cat   /usr/share/doc/setup*/uidgid
```

TYPES OF USERS

SUPER USER:

- The **root user** account is the equivalent of the administrator or enterprise admin account in the windows.
- The root user ID value is **"0"**.
- Login prompt is **"#".**

NORMAL USERS:

- Normal users are created by the root user, like Raju, Sara, max…etc.
- These accounts have **no write access** to anything on the system except their home directory.
- Normal user id values are above **"1000".**
- Login prompt is **"$"**.

SYSTEM USERS:

- System users normally don't have a home directory and can't login the way normal users do.
- These are created with **applications or service** to help run them more securely.
- System users' id values are **"1-999".**

USER PRIVATE GROUP

- RHEL uses the **user private group (UPG)** system configuration, which makes UNIX groups easier to manage.
- A user private group is created whenever a new user is added to the system. The user private group has the same name as the user for which it was created and that user is the only member of the user private group.

WHAT THINGS ARE CREATED BY DEFAULT

- A home directory is created **(/home/username)**
- A mail box is created **(/var/spool/mail/username)**
- Login shell is created **(/bin/bash)**
- Unique **UID & GID** values are given to a user

FILES CONTROLLING USERS

/ETC/PASSWD FILE:

- This file is world-readable and contains a list of users, each on a separate line.

→ To verify the user fields:

```
#cat /etc/passwd
```

/ETC/SHADOW FILE:

- This file must be world-readable, there is a risk involved in storing everyone's password in a file.
- True, the passwords are **encrypted.**

→ To verify the password fields:

```
#cat /etc/shadow
```

/ETC/LOGIN.DEFS FILE:

- The **/etc/login.defs** file provides default configuration information for several user and group account parameters.

USERADD / ADDUSER:

- **useradd / adduser** utility creates new users and adds them to the system.

SYNTAX: **#useradd [options] login**

→ Create a new user with default options:

```
#useradd raju
```

→ To verify the user details:

```
#cat /etc/passwd | grep -i raju
#id raju
```

→ To login a user with su:

```
#su – raju
#pwd
#ls -a
#echo $0
#exit
```

→ Create a user with non-default options:

```
#useradd -u 1010 -c "Developer" -d /opt/jai -s /bin/sh jai
#cat /etc/passwd | grep -i jai
```

→ Creating a User while Copying Contents to the Home Directory:

```
#useradd -m -k /opt/jai ram
#su – ram
#ls -a
```

→ Setting up the Account Expiration Date [YYYY-MM-DD]:

```
#useradd -e 2024-12-31 max
```

→ Creates a system account:

```
#useradd -r smith
```

→ If you do not want to create the home directory for the user at all:

```
#useradd -M john
#cat /etc/passwd | grep -i john
```

NOTE: We can’t login user’s home directory here.

USERMOD:

- It is used to modify user account settings or options.

SYNTAX: **#usermod [options] login**

→ Modify uid (u), comment(c), login shell (s):

```
#usermod -u 1005 -c “Tester” -s /bin/bash raju
#cat /etc/passwd | grep -i jai
```

→ Changing User's Login:

```
#usermod -l ramu jai
```

→ Changing user home directory:

```
#usermod -d /opt/ramu ramu
```

NOTE: With the usermod command you can also move the content of the user's home directory to a new location, or lock the account by locking its password.

```
#usermod -m -d /home/ramu -L ramu
```

PASSWD:

- It will update user's authentication tokens.
- When running the basic useradd username command, the password is automatically set to never expire

SYNTAX: **# passwd [options] login**

→ To setting up a new / update password:

```
#passwd raju
```

NOTE: Type the password twice when the program prompts you to. Password should be strong.

→ To verify the password details:

```
#cat /etc/shadow | grep -i raju
#passwd -S raju
```

→ To locks raju's account password:

```
#passwd -l raju
#cat /etc/shadow | grep -i raju
#passwd -S raju
```

NOTE: When the user password has been locked, the **/etc/shadow** file shows **exclamation mark (!)** in the beginning of encrypted password field.

→ To unlocks raju's account password successfully:

```
#passwd -u raju
#passwd -S raju
#cat /etc/shadow | grep -i raju
```

→ If you want a password for an account to expire:

```
#passwd -e raju
```

→ Adjusting Aging Data for User Passwords:

```
#passwd -n 10 -x 60 -w 3 raju
```

→ To delete a password:

```
#passwd -d raju
#passwd -S raju
```

CHAGE:

- It is used to change user password expiry information

SYNTAX: **#chage [options] login**

→ To verify user password aging information:

```
#chage -l raju
```

→ To set expire date for the user account [YYYY-MM-DD]:

```
#chage -E 2024-12-31 raju
```

→ Setting minimum (m), maximum(M), and warning(W) days:

```
#chage -m 30 -M 45 -W 3 raju
#chage -l raju
```

→ To change last password change:

```
#chage -d 2023-12-01 raju
```

→ To specify the number of days the account should be inactive after its expiry:

```
#chage -I 5 raju
```

→ To change user's password at first login:

```
#chage -d 0 raju
```

USERDEL:

- It will delete a user account from the system.

SYNTAX: **#userdel [options] login**

→ To delete a user account:

```
#userdel raju
```

→ To delete user account along with user's home directory:

```
#userdel -r raju
#id raju
```

USER ACCOUNT INITILIZATION

- When a user is created, everything from the **“/etc/skel”** directory is copied to the user’s newly created home directory (usually **/home/username**).
- The customizable files are broken down into two different sections.

USER SPECIFIC FILES:

- **.bashrc** : Defines functions and aliases.
- **.bash_profile:** : Sets environment variables.
- **.bash_logout:** : Any commands that should be executed before users log out.

GLOBAL USER CONFIGURATION:

- **/etc/bashrc** : Defines functions and aliases.
- **/etc/profile:** : Sets environment variables.
- **/etc/profile.d:** : It contains scripts that are called by the **/etc/profile** file.

COMMAND LINE UTILITIES:

- **id** : Displays user and group IDs.
- **useradd, usermod, userdel** : adding, modifying, and deleting user ac’s.
- **passwd** : Update user password authentications.
- **pwck** : Verification of the password.
- **chage** : Change user password expiry information
- **pwconv, pwunconv** : Conversion of passwords to shadow passwords, or back from shadow passwords to standard passwords.

➢ MANAGING GROUPS:

- A group is an entity which ties together multiple user accounts for a common purpose, such as granting access to particular files.
- On Linux, user groups can act as primary or supplementary. Primary and supplementary groups have the following properties:

PRIMARY GROUP:

- Every user has just one primary group at all times.
- You can change the user's primary group.

SUPPLEMENTARY GROUPS:

- Add an existing user to an existing supplementary group to manage users with the same security and access privileges within the group.
- Users can be members of zero or multiple supplementary groups.

FILES CONTROLLING GROUPS:

/ETC/GROUP FILE:

- This file is world-readable and contains a list of groups, each on a separate line.

```
#cat /etc/group
```

/ETC/GSHADOW FILE:

- It is a readable only by the root user & contains an encrypted password for each group, as well as group membership & administrator information

```
#cat /etc/gshadow
```

COMMAND LINE UTILITIES:

- **groupadd, groupmod, groupdel** **:** Adding, modifying, and deleting groups.
- **gpasswd** : For modification of group password in the **/etc/gshadow** file
- **grpck** **:** verification of the password, group, and associated
- **grpconv, grpunconv** **:** Conversion of shadowed info for group accounts.

GROUPADD:

- It is used to create a new group.

  SYNTAX: **#groupadd [options] groupname**

  → Creating a Group with Default Settings:

  ```
  #groupadd friends
  #cat /etc/group | grep -i friends
  ```

  → Create a group with own gid:

  ```
  #groupadd -g 1015 schoolmates
  #cat /etc/group | grep -i schoolmates
  ```

  → To create a system group:

  ```
  #groupadd -r group-name
  ```

GROUPMOD:

- It is modifying group settings.

  SYNTAX: **#groupmod [options] groupname**

  → To change the group id:

  ```
  #groupmod -g 1012 friends
  #cat /etc/group | grep -i friends
  ```

  → To change a group name:

  ```
  #groupmod -n new_groupname groupname
  #groupmod -n sports friends
  #cat /etc/group | grep -i sports
  ```

  → Setting up primary group and secondary group for the new user:

  ```
  #useradd -g friends -G schoolmates sachin
  ```

  → To verify the details:

  ```
  #id -Gn sachin
  ```

GPASSWD:

- It is used to update group authentication and attaching, removing users to groups.

  SYNTAX: **#gpasswd [options] groupname**

  → Updating Group Authentication:

  ```
  #gpasswd sports
  #cat /etc/gshadow | grep -i sports
  ```

  → Remove password from the group:

  ```
  #gpasswd -r sports
  ```

  → Adding existing user into the group:

  ```
  #gpasswd -a raju sports
  ```

  → Adding multiple users into the group:

  ```
  #gpasswd -M raju,ram,jai,max sports
  #cat /etc/gshadow | grep -i sports
  ```

  → User raju to make an admin of the group:

  ```
  #gpasswd -A raju sports
  ```

  → To remove a user from the named group:

  ```
  #gpasswd -d max sports
  ```

GROUPDEL:

- The groupdel command modifies the system account files, deleting all entries that see the group.

  SYNTAX: **#groupdel groupname**

  ```
  #groupdel sports
  ```

  → To verify the group details:

  ```
  #cat /etc/group | grep -i sports
  ```

CONFIGURING SUDO ACCESS

- System administrators can grant **sudo** access to allow non-root users to execute administrative commands that are normally reserved for the root user.
- As a result, non-root users can execute such commands without logging in to the root user account.
- Run the **visudo** to edit the **/etc/sudoers** file.

→ User raju to run any commands anywhere:

```
#visudo

raju   ALL=(ALL)        ALL
```

→ To verify from the user raju:

```
#su – raju

$useradd rahul

        Enter raju user password

$cat /etc/passwd | grep -i rahul
```

→ User raju to run any commands anywhere without password:

```
raju   ALL=(ALL)        NOPASSWD: ALL
```

→ User jai to run only specific commands:

```
jai     ALL=/usr/sbin/useradd, /usr/sbin/groupadd
```

→ Allows people in group friends to run all commands:

```
%friends    ALL=(ALL)                  ALL
```

→ Allows people in group friends to run commands without a password:

```
%friends    ALL=(ALL)        NOPASSWD: ALL
```