User Management and Groups in Linux

Linux provides robust tools for managing users and groups, essential for controlling access and ensuring security in multi-user environments.

User Management Commands

Command	Description
`whoami`	Display the current logged-in username.
`id <username>`	Show user ID (UID), group ID (GID), and group memberships for a user.
`adduser <username>`	Add a new user with default settings (interactive).
`passwd <username>`	Set or change the password for a user.
`deluser <username>`	Remove a user (but keep their files).
`userdel -r <username>`	Remove a user and their home directory.
`w`	Display logged-in users and their activities.
`last`	Show login history of users.

Adding a User

Using `adduser` (interactive):

```
sudo adduser alice
```

1.
    - Prompts for details like password, full name, etc.

Deleting a User

Remove the user but keep their files:

```
sudo deluser alice
```

1.

Remove the user and their home directory:

```
sudo userdel -r alice
```

2.

Group Management Commands

Command	Description
`groups <username>`	List groups a user belongs to.
`groupadd <groupname>`	Add a new group.
`groupdel <groupname>`	Delete a group.
`usermod -aG <groupname> <username>`	Add a user to a group.
`id`	Show the current user's groups.

Adding a Group

Create a new group:

```
sudo groupadd developers
```

1.

Add a user to the group:

```
sudo usermod -aG developers alice
```

2.
    - `-aG`: Appends the user to the group.

Deleting a Group

Remove a group:

```
sudo groupdel developers
```

1.

Remove a user from a group:

```
sudo gpasswd -d alice developers
```

2.

Default User and Group IDs

- **UID (User ID):**
  - `0`: Reserved for the `root` user.
  - `1-99`: Reserved for system users.
  - `1000+`: Assigned to regular(General) users.
- **GID (Group ID):**
  - Functions similarly to UID, with unique group IDs for each group.

Switching Users

Switch to another user:

```
su - alice
```

-

Return to the previous user:

```
exit
```

-

User and Group Configuration Files

1. **User Information:**

    - `/etc/passwd`: Stores user account information.
        - Format: `username:x:UID:GID:FullName:HomeDirectory:Shell`
    - `/etc/shadow`: Stores encrypted passwords and password policies.
2. **Group Information:**
    - `/etc/group`: Stores group names and their members.

What is Shell Scripting?

A **shell script** is a program written using a shell language (e.g., Bash) that allows you to automate tasks in a Linux or Unix-like operating system. The shell acts as a command-line interpreter, and shell scripts contain sequences of commands, loops, and conditions that the shell executes.

Why Use Shell Scripting?

1. **Automation**: Perform repetitive tasks automatically (e.g., backups, deployments).
2. **Efficiency**: Execute multiple commands in one script.
3. **Custom Tools**: Create tailored solutions for specific problems.
4. **Integration**: Combine existing commands and tools for complex workflows.

Features of Shell Scripting

- Combines **commands** and **logic** (e.g., conditions, loops).
- Portable across Unix/Linux systems.
- Extensible with utilities (e.g., `grep`, `awk`, `sed`).
- Used in automation, system administration, and software development.

Structure of a Shell Script

Shebang (`#!`): Specifies the shell to interpret the script.

```
#!/bin/bash
```

1.

Commands: Shell commands to be executed.

```
echo "Hello, World!"
```

2.

Variables: Store data.

```
name="Alice"
echo "Welcome, $name!"
```

3.

Example Shell Script

Hello, World!

```
#!/bin/bash
echo "Hello, World!"
```

Basic User Interaction

```
#!/bin/bash
read -p "Enter your name: " name
echo "Hello, $name! Welcome to shell scripting."
```

Steps to Create and Run a Shell Script

Create the Script: Use a text editor like `nano`, `vim`, or `gedit` to write the script.

```
nano script.sh
```

1.

Make it Executable: Grant execute permissions to the script.

```
chmod +x script.sh
```

2.

Run the Script: Execute the script from the terminal.

```
./script.sh
```

3.

Applications of Shell Scripting

- **System Administration**: Manage users, groups, and processes.
- **Automation**: Automate deployments and backups.
- **Monitoring**: Log system activities or resource usage.
- **Development**: Build, test, and deploy applications.