Module: 5- Linux server - Deploy, configure, and maintain systems Assignment

1. Schedule tasks using cron or at.

➔ 1. Cron

- `cron` is a tool used to schedule recurring tasks on Unix-like systems (like Linux or macOS). It runs tasks at specified times, dates, or intervals.
- To schedule a task with cron, we use a crontab (cron table). Each user has their own crontab where we can specify the command to run and the time it should run.

Syntax of a cron job:

lua

```
* * * * * command_to_run

- - - - -

| | | | |

| | | | +---- Day of week (0 - 6) (Sunday=0)

| | | +------ Month (1 - 12)

| | +-------- Day of month (1 - 31)

| +---------- Hour (0 - 23)

+------------ Minute (0 - 59)
```

Example:

To run a task every day at 5 PM:

pgsql

```
0 17 * * * /path/to/script.sh
```

This means:

- `0` minute
- 17 hour (5 PM)
- Every day of the month
- Every month
- Every day of the week

To edit your cron jobs:

1. Type `crontab -e` in the terminal.
2. Add your job with the desired timing.

To see your scheduled jobs, type:

```
crontab -l
```

2. At

The at command is used to schedule a one-time task to run at a specific time in the future. Unlike cron, it's for tasks you only want to run once.

Syntax of an at command:

```
at TIME

at> command_to_run

at> <press Ctrl+D to save>
```

Example:

To run a script at 3 PM today:

```
at 3pm

at> /path/to/script.sh

at> <press Ctrl+D>
```

This will execute the script once at 3 PM.

To view the scheduled tasks for at, you can use:

```
Atq
```

And to remove a task:

```
atrm JOB_ID
```

2. Use apt or yum (depending on your Linux distribution) to install, update, and remove software packages.

- ➔ apt and yum are tools used on Linux systems to manage software. The commands help us install, update, and remove software packages.

1. To Install Software:

- For Ubuntu/Debian (uses apt):
  ```
  sudo apt install package_name
  ```
- For Red Hat/CentOS/Fedora (uses yum):
  ```
  sudo yum install package_name
  ```

2. To Update Software:

- For Ubuntu/Debian (uses apt):
  ```
  sudo apt update       # Updates the list of available packages

   sudo apt upgrade      # Upgrades installed packages
  ```
- For Red Hat/CentOS/Fedora (uses yum):
  ```
  sudo yum update
  ```

3. To Remove Software:

- For Ubuntu/Debian (uses apt):
  ```
  sudo apt remove package_name
  ```
- For Red Hat/CentOS/Fedora (uses yum):
  ```
  sudo yum remove package_name
  ```

3.Install all httpd package

- ➔ To install the httpd package (which is the Apache web server) on our Linux system, follow these steps:

1. For Ubuntu/Debian (uses apt):

- Open a terminal and run the following command:
  ```
  sudo apt install apache2
  ```
- This will install the Apache web server on our system. (In Ubuntu/Debian, the package is called apache2 instead of httpd.)

2. For Red Hat/CentOS/Fedora (uses yum):

- Open a terminal and run the following command:
  ```
  sudo yum install httpd
  ```
- This will install the Apache web server (httpd) on our system.

Once installed, we can start the web server using:

For Ubuntu/Debian:
```
sudo systemctl start apache2
```

For Red Hat/CentOS/Fedora:
```
sudo systemctl start httpd
```

That's it! The httpd (Apache) web server should now be installed and running on our system.

4. Open kickstart configuration graphically

➔ To open the Kickstart configuration file graphically, the process depends on the Linux distribution we're using. Kickstart is used to automate the installation of Red Hat-based systems like RHEL, CentOS, or Fedora. While there's no specific graphical tool for Kickstart by default, we can use text editors or specialized graphical tools to manage it.

Here are the steps to open and edit the Kickstart configuration file graphically:

1. Using a Graphical Text Editor (like gedit or gedit):

If we have a GUI (graphical user interface) available, we can open the Kickstart configuration file using a text editor. Here’s how:

For Ubuntu/Debian:

- Open a terminal and use a command to open the Kickstart file:
  ```
  sudo gedit /path/to/our/kickstart.cfg
  ```

Replace /path/to/our/kickstart.cfg with the actual location of the Kickstart file.

For RHEL/CentOS/Fedora:

- Open a terminal and use:
  ```
  sudo gedit /path/to/your/kickstart.cfg
  ```

5.Configure new kickstart file

- ➔ A Kickstart file is used to automate the installation of Red Hat-based Linux systems (like CentOS, RHEL, Fedora). It contains instructions that the system follows during the installation process.

Here's we can create a basic Kickstart file step-by-step:

1. Create a New Kickstart File

First, we need to create a new file using a text editor. Let's say we are using **gedit** or **nano**.

Open our terminal and run:

sudo gedit /path/to/kickstart.cfg

or, if we're using **nano**:

sudo nano /path/to/kickstart.cfg

2. Basic Kickstart Configuration Example

Now, in the text editor, write the following basic Kickstart configuration:

- ➔ # Basic Kickstart Configuration Example
- ➔ # Install the system

```
install
```

- ➔ # Choose the installation source (URL of the repo)

```
url --url="http://mirror.centos.org/centos/7/os/x86_64/"
```

- ➔ # Set language (optional)

```
lang en_US.UTF-8
```

- ➔ # Set keyboard layout

```
  keyboard us

➔ # Set timezone (we can change this based on our location)

  timezone India/delhi

➔
➔ # Set root password (use an encrypted password)

    rootpw --iscrypted $1$abcdef$Our_encrypted_password_here

➔ # Network configuration

  network --device eth0 --bootproto dhcp --hostname
  myserver.localdomain

➔ # Disk partitioning (automatically partition)

   autopart

➔ # Select packages to install

  %packages

  @core           # Minimal install

  vim             # Text editor (we can add more packages)

  httpd           # Apache web server

  %end

➔ # Post-installation script (optional)

  %post

  echo "Post installation complete!" > /tmp/post_install.log

  %end
```

3. Explanation of Sections

- install: This tells the system to start installation.
- url: Specifies where to get the installation files (in this case, from a CentOS mirror).

- lang: Sets the system language (English here).
- keyboard: Defines the keyboard layout (US English here).
- timezone: Specifies our timezone (we can change it based on our location).
- rootpw: Sets the root password. we can generate an encrypted password using the openssl command (explained below).
- network: Configures the network interface. In this example, it uses DHCP to get an IP address.
- autopart: This automatically handles disk partitioning.
- %packages: Specifies which packages to install (in this case, @core, vim, and httpd).
- %post: This is a section for post-installation actions. we can put additional commands here to run after the system is installed.

4. Generate Encrypted Password

To make our root password secure, it is recommended to encrypt it. we can generate an encrypted password using this command:

```
openssl passwd -1 "ourpassword"
```

This will output an encrypted password that we can copy and paste into our Kickstart file.

5. Save the Kickstart File

After we've written our Kickstart configuration, save the file. we can now use it during the installation process to automate the setup.

6. Using the Kickstart File

Once our Kickstart file is ready, you can use it during the installation. If you're installing from a bootable media (like a DVD or USB), we can specify the Kickstart file by typing:

➔ linux ks=http://example.com/kickstart.cfg

Replace the URL with the location of your Kickstart file.