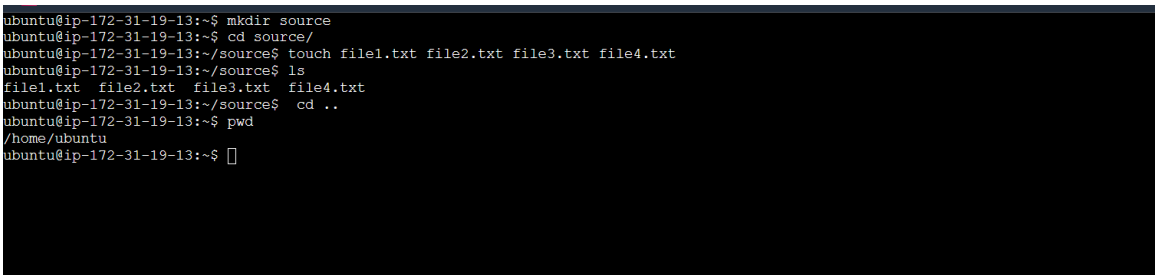
Real Time Shell Scripts Example ::

1 . Take backup of a particular folder

Create two folders as the **source** and **destination**. Then, inside the source folder, create some files. This source folder will be used as a backup folder, and the destination folder where we will take the backup.



Create a script file as **backup.sh** and set the **source folder path** in the script **[for me its /home/ubuntu/source]** and also mention the **destination [/home/ubuntu/destination]** folder where the backup will be taken .

Explanation

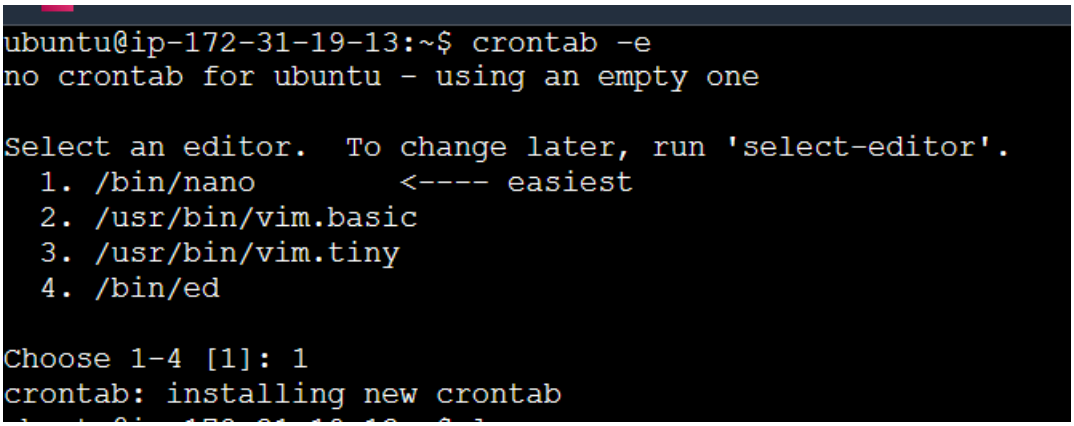
- **SOURCE: The directory to be backed up.**
- **DESTINATION: The directory where the backup will be stored.**
- **DATE: Captures the current date and time to create a unique backup folder.**
- **mkdir -p $DESTINATION/$DATE: Creates the backup directory if it does not exist.**
- **cp -r $SOURCE $DESTINATION/$DATE: Copies the contents of the source directory to the backup directory.**
- **echo "Backup completed on $DATE": Outputs a message indicating the completion of the backup.**

```
#!/bin/bash

SOURCE="/home/ubuntu/source"

DESTINATION="/home/ubuntu/destination"

DATE=$(date +%Y-%m-%d_%H-%M-%S)

# Create backup directory and copy files
mkdir -p $DESTINATION/$DATE

cp -r $SOURCE $DESTINATION/$DATE

echo "Backup completed on $DATE"
```

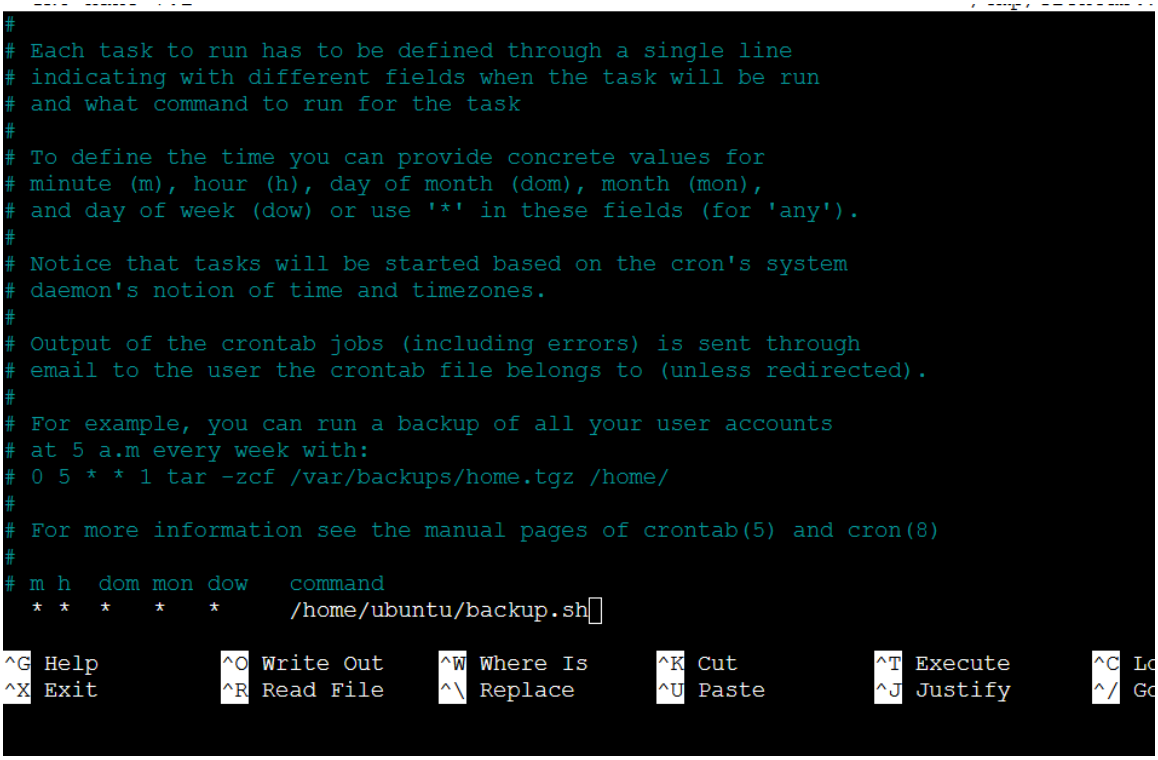
Now give the **executable permission** to the **backup file** and execute the script as **/backup.sh** . Then check the destination folder, and you will see that one backup is there.

```
ubuntu@ip-172-31-19-13:~$ chmod +x backup.sh
ubuntu@ip-172-31-19-13:~$ ./backup.sh
Backup completed on 2024-05-23_13-44-09
ubuntu@ip-172-31-19-13:~$ ls
backup.sh  destination  source
ubuntu@ip-172-31-19-13:~$ cd destination/
ubuntu@ip-172-31-19-13:~/destination$ ls
2024-05-23_13-44-09
ubuntu@ip-172-31-19-13:~/destination$ cd 2024-05-23_13-44-09/
ubuntu@ip-172-31-19-13:~/destination/2024-05-23_13-44-09$ ls
source
ubuntu@ip-172-31-19-13:~/destination/2024-05-23_13-44-09$ cd source/
ubuntu@ip-172-31-19-13:~/destination/2024-05-23_13-44-09/source$ ls
file1.txt  file2.txt  file3.txt  file4.txt
ubuntu@ip-172-31-19-13:~/destination/2024-05-23_13-44-09/source$
```

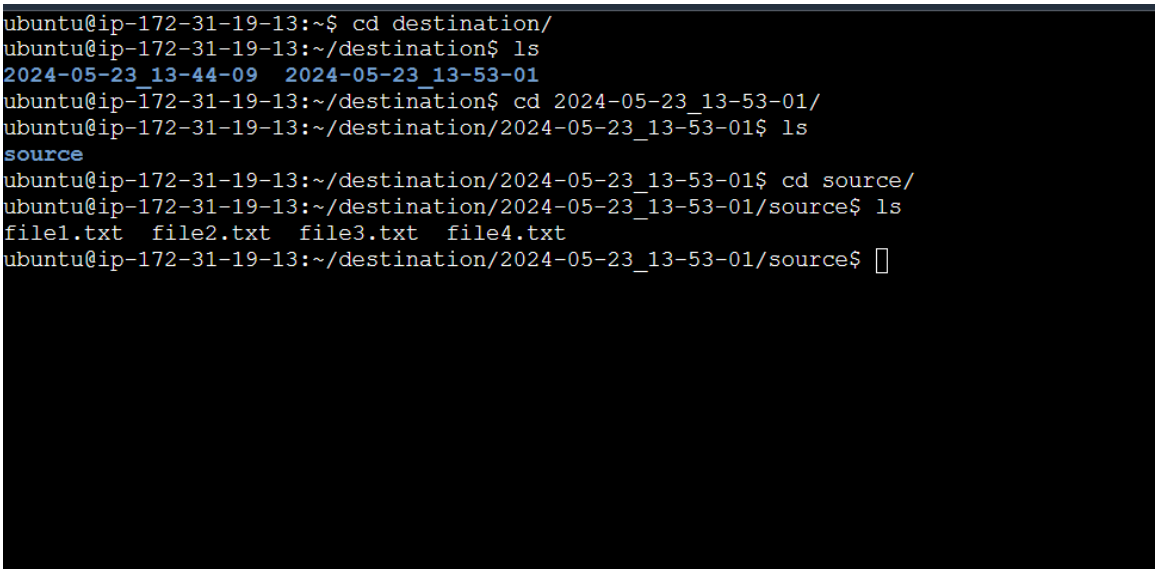
Another process to execute the script by using the **cron job** . Open the **crontab -e** file in editing mode.



Now, you can set the cron job as per your requirements. If I want to execute the script **every minute**, the cron job will be as follows:



After some time, if you check the **destination folder**, you will see another backup of the source file inside the date folder.



2. To check whether a Service is running or not in your system .

Create a script file called as **Service-health-check.sh** and this is the script.

In my case, I have installed the nginx service to check its status.

sudo apt-get install nginx -y

service nginx status ;; manually check whether its in running state or not

Explanation

- **SERVICE: The name of the service to check.**
- **systemctl is-active --quiet $SERVICE: Checks if the service is running.**
- **echo "$SERVICE is running": Prints a message if the service is running.**
- **systemctl start $SERVICE: Starts the service if it is not running.**

```
#!/bin/bash

SERVICE="nginx"   ## put any service name that you want to check

# Check if the service is running, if not, start it

if systemctl is-active --quiet $SERVICE; then

echo "$SERVICE is running"

else

echo "$SERVICE is not running"

systemctl start $SERVICE

fi
~
~
~
~
~
```

Give the executable permission to the script file and execute it manually, or you can set the cron job as well.

```
ubuntu@ip-172-31-19-13:~$ ./service-health-check.sh
nginx is running
ubuntu@ip-172-31-19-13:~$
```