Unit 1.1 Graded Assignment: RSync

Group members

Ifra Saleem (2303.khi.deg.003)
Muhammad Khan (2303.khi.deg.027)

Unit 1.1 Graded Assignment: RSync

Question:

On a linux server setup a cron job for copying example data with *rsync* periodically. Ensure the copying is handled in the background and independently of the user session.

Explanation of the question:

According to the above we have to setup a cron job which can copy a file from one location to other using rsync (rsync is a linux command-line tool that lets you transfer files and directories to local and remote destinations) and the copying should be done automatically according to the given timestamp.

Syntax:

rsync -options SOURCE_PATH DESTINATION_PATH

Solution:

<u>Step #01</u>:

First of all, I created a bash script file (sh file is a shell script which can be executed in a terminal). In this data.sh file I write a script which is using rsync to copy a new.txt file from one location to other.

Command:

rsync -avz /home/all/Documents/new.txt /home/all/Documents/New

- rsync command is used to synchronize the files or directories between two locations, locally or remotely.
- -a is for archive files and it used to preserve everything like file permission, ownership, timestamps etc.
- -v is for verbose output and it contains information about the file being transferred.
- -z is used to compress the files or folders being transferred, reducing the file size and the time required for the file transfer.

- After rsync -avz we have the source path (the path of the file which we want to copy) and then we have the destination path (the path where we want to copy that file).

```
Open   data.sh ~/Documents   Save
1 rsync -avz /home/all/Documents/new.txt /home/all/Documents/New
```

Step #02:

After creating the data.sh file I have to run a command in the terminal to make the file executable.

Command:

chmod +x /home/all/Documents/data.sh

- chmod is used to modify the permission for the file.
- +x adds the execute permission to the file so that it can run the script.
- And then I mention the path of the data.sh file.

Step #03:

After making the data.sh file executable I open the crontab configuration file by using the following command:

- **crontab -e**

`Cron` is a system daemon used to execute tasks at designated times.

```
  GNU nano 6.2                    /tmp/crontab.RMPzDu/crontab
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
33 09 * * * /home/all/Documents/data.sh
```

In the crontab configuration file I mentioned the time of the rsync command by using the following command: The time could be given using the following syntax:

- **minute hour day_of_month month day_of_week command_to_run**

Command:

*33 09 * * * /home/all/Documents/data.sh*

- 33 09 * * * is the time at which we want the crontab to perform the task mentioned in the data.sh file. I only mentioned minute and hour (33-minute, 09 hour). It means that it will copy the file at 09:33.
- Then there is a path of the data.sh file.
- Save the command and exit the crontab.

Final Result: