11 Cron Command Examples in Linux [Schedule Cron Jobs]

In this article, we are going to review and see how we can schedule and run Linux tasks in the background automatically at regular intervals using the **Crontab** command.

Dealing with a frequent job manually is a daunting task for system administrators and such tasks can be scheduled and run automatically in the background without human intervention using cron daemon in Linux or Unix-like operating system.

[You might also like: How to Create and Manage Cron Jobs on Linux **]**

For instance, you can automate Linux system backup, **schedule updates,** and synchronization of files, and many more using **Cron** daemon, which is used to run scheduled tasks from the command line or use online tools to generate cron jobs.

Cron wakes up every minute and checks scheduled tasks in countable – **Crontab** (**CRON TABle**) is a table where we can schedule such kinds of repeated tasks.

Tips: Each user can have their own crontab to create, modify and delete tasks. By default **cron** is enabled for users, however, we can restrict users by adding an entry in **/etc/cron.deny** file.

Crontab in Linux

The **Crontab** file consists of commands per line and has six fields actually and separated either by space or tab. The beginning five fields represent the time to run tasks and the last field is for command.

- Minute (hold values between **0-59**)
- Hour (hold values between **0-23**)
- Day of Month (hold values between **1-31**)
- The month of the year (hold values between **1-12** or **Jan-Dec**, you can use the first three letters of each month's name i.e **Jan or Jun**.)
- Day of week (hold values between **0-6** or **Sun-Sat**, Here also you can use the first three letters of each day's name i.e **Sun or Wed**.)
- Command – The /path/to/command or script you want to schedule.

1. List Crontab Entries

List or manage the task with the crontab command with `-l` option for the current user.

```
# crontab -l




00 10 * * * /bin/ls >/ls.txt
```

2. Edit Crontab Entries

To edit the crontab entry, use `-e` the option shown below. The below example will open scheduled jobs in the **VI** editor. Make necessary changes and quit pressing `:wq` keys that save the setting automatically.

```
# crontab -e
```

3. List Scheduled Cron Jobs of User

To list scheduled jobs of a particular user called **tecmint** using the option as `-u` (**User**) and `-l` (**List**).

```
# crontab -u tecmint -l




no crontab for tecmint
```

Note: Only **root** user have complete privileges to see other users' crontab entries. Normal users can't view others.

4. Remove Crontab Entry

Caution: Crontab with `-r` the parameter will remove complete scheduled jobs without confirmation from Crontab. Use `-i` option before deleting the user's crontab.

```
# crontab -r
```

5. Prompt Before Deleting Crontab

crontab with `-i` the option will prompt you confirmation from the user before deleting the user's crontab.

```
# crontab -i -r




crontab: really delete root's crontab?
```

6. Allowed Special Characters (*, -, /, ?, #)

- **Asterisk(*)** – Match all values in the field or any possible value.
- **Hyphen(-)** – To define a range.
- **Slash (/)** – 1st field /10 meaning every ten minutes or increment of range.
- The **Comma (,)** – To separate items.

7. System-Wide Cron Schedule

A system administrator can use the predefined cron directory as shown below.

- /etc/cron.d
- /etc/cron.daily
- /etc/cron.hourly
- /etc/cron.monthly
- /etc/cron.weekly

8. Schedule a Job for a Specific Time

The below jobs delete empty files and directories from **/tmp** at **12:30** am daily. You need to mention the user name to perform the crontab command. In the below example, **root** user is performing a cron job.

```
# crontab -e




30 0 * * *   root   find /tmp -type f -empty -delete
```

9. Special Strings for Common Schedule

Strings	Meanings
@reboot	The command will run when the system reboots.
@daily	Once per day or may use @midnight.
@weekly	Once per week.
@yearly	Once per year. we can use the @annually keyword also.

Need to replace five fields of the cron command with keywords if you want to use the same.

10. Multiple Commands with Double ampersand(&&)

In the below example, command1 and command2 run daily.

```
# crontab -e




@daily <command1> && <command2>
```

11. Disable Email Notifications.

By default, cron sends mail to the user account executing cronjob. If you want to disable it add your cron job similar to the below example. Using the **>/dev/null 2>&1** option at the end of the file will redirect all the output of the cron results under **/dev/null**.

```
[root@tecmint ~]# crontab -e



* * * * * >/dev/null 2>&1
```

conclusion: Automation of tasks may help us to perform our tasks in better ways, error-free, and efficient. You may refer to a manual page of crontab for more information by typing the '**man crontab**' command in your terminal.