

# JOB SCHEDULING IN LINUX

Linux is an open-source kernel for operating systems. It is widely used in operating systems such as Ubuntu, Debian, etc. One of the unique features of Linux is that it allows the user to have a higher degree of control over the machine. We can execute and program a vast number of things into our computer using simple commands.

Scheduling processes and commands is one of these things. If we wish to schedule a particular script to run next Monday, how would we do so? This is what we will look at in our article. There are 2 basic commands that we can use for this - at and crontab.

## At Command

The **at** command can be used to schedule a job to run at a specific time. We can send reminders, execute system updates, and take backups at our desired time. We can specify a specific time and date (such as 12th October at 12 AM) or an interval (such as after 3 hours).

### Installation:

**at** needs to be installed first. We can use **apt-get** to install it. At is not always installed by default on all Unix-like systems. Run the command:

```
sudo apt-get install at
```

**atq** : To list all the jobs currently pending

**atrm** :atrm is used to delete a pending job.

Once it is installed, we must start it and enable it. We can do so by running the commands:

```
sudo systemctl start atd
```

```
sudo systemctl enable atd
```

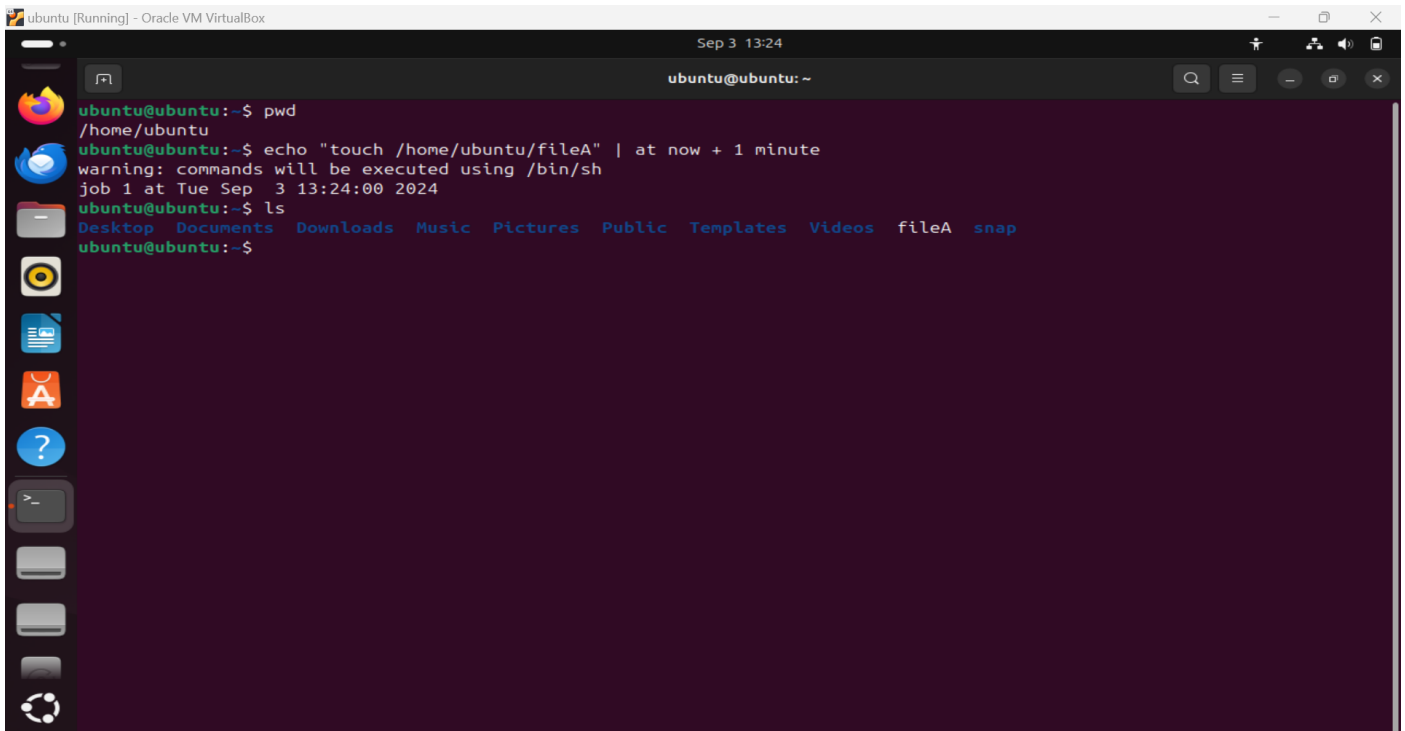
```
ubuntu [Running] - Oracle VM VirtualBox
Sep 3 13:15
ubuntu@ubuntu: ~
ubuntu@ubuntu:~$ at -v
Command 'at' not found, but can be installed with:
sudo apt install at
ubuntu@ubuntu:~$ sudo apt install at
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  default-mta | mail-transport-agent
The following NEW packages will be installed:
  at
0 upgraded, 1 newly installed, 0 to remove and 108 not upgraded.
Need to get 40.6 kB of archives.
After this operation, 159 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu noble/universe amd64 at amd64 3.2.5-2.1ubuntu3 [40.6 kB]
Fetched 40.6 kB in 1s (64.4 kB/s)
Selecting previously unselected package at.
(Reading database ... 212173 files and directories currently installed.)
Preparing to unpack .../at_3.2.5-2.1ubuntu3_amd64.deb ...
Unpacking at (3.2.5-2.1ubuntu3) ...
Setting up at (3.2.5-2.1ubuntu3) ...
Created symlink /etc/systemd/system/multi-user.target.wants/atd.service → /usr/lib/systemd/system/atd.service.
Processing triggers for man-db (2.12.0-4build2) ...
ubuntu@ubuntu:~$
```

```
ubuntu [Running] - Oracle VM VirtualBox
Sep 3 13:16
ubuntu@ubuntu: ~
ubuntu@ubuntu:~$ sudo systemctl start atd
ubuntu@ubuntu:~$ sudo systemctl enable atd
Synchronizing state of atd.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable atd
ubuntu@ubuntu:~$ sudo systemctl status atd
● atd.service - Deferred execution scheduler
   Loaded: loaded (/usr/lib/systemd/system/atd.service; enabled; preset: enabled)
   Active: active (running) since Tue 2024-09-03 13:14:56 UTC; 1min 34s ago
     Docs: man:atd(8)
    Main PID: 12825 (atd)
      Tasks: 1 (limit: 5747)
     Memory: 260.0K (peak: 1.5M)
        CPU: 10ms
     CGroup: /system.slice/atd.service
            └─12825 /usr/sbin/atd -f

Sep 03 13:14:56 ubuntu systemd[1]: Starting atd.service - Deferred execution scheduler...
Sep 03 13:14:56 ubuntu systemd[1]: Started atd.service - Deferred execution scheduler.
ubuntu@ubuntu:~$
```

To schedule the creation of an empty file, use the at command. For example, to create the file one minute from now:

- `echo "touch /path/to/yourfile.txt" | at now + 1 minute`
- “ls” check the file has been created.

A screenshot of a terminal window titled 'ubuntu [Running] - Oracle VM VirtualBox'. The terminal shows a user named 'ubuntu' at the prompt 'ubuntu@ubuntu:~'. The user runs 'pwd' and gets '/home/ubuntu'. Then, they run 'echo "touch /home/ubuntu/fileA" | at now + 1 minute'. A warning message appears: 'warning: commands will be executed using /bin/sh'. Below that, it says 'job 1 at Tue Sep 3 13:24:00 2024'. Finally, the user runs 'ls' and the output shows 'Desktop Documents Downloads Music Pictures Public Templates Videos fileA snap'. The terminal has a dark purple background and a sidebar on the left with various application icons.

```
ubuntu@ubuntu:~$ pwd
/home/ubuntu
ubuntu@ubuntu:~$ echo "touch /home/ubuntu/fileA" | at now + 1 minute
warning: commands will be executed using /bin/sh
job 1 at Tue Sep 3 13:24:00 2024
ubuntu@ubuntu:~$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos fileA snap
ubuntu@ubuntu:~$
```

## Cron

One limitation of the `at` command is that we can't use it to run recurring tasks. For example, if I want to run a task every Monday, we can't use `at` to accomplish this. For this, we use `crontab`. Before we see what `crontab` is, let us look at the `cron` daemon first.

### Cron Daemon

The `cron` daemon is a background process that is always running on a Linux machine. It is used to schedule jobs at regular time intervals, at specific recurring intervals, or on a particular date and time.

Cron Tables - known as `crontab` for short, are where the commands to be run, and the times to run them are stored.

### Crontab

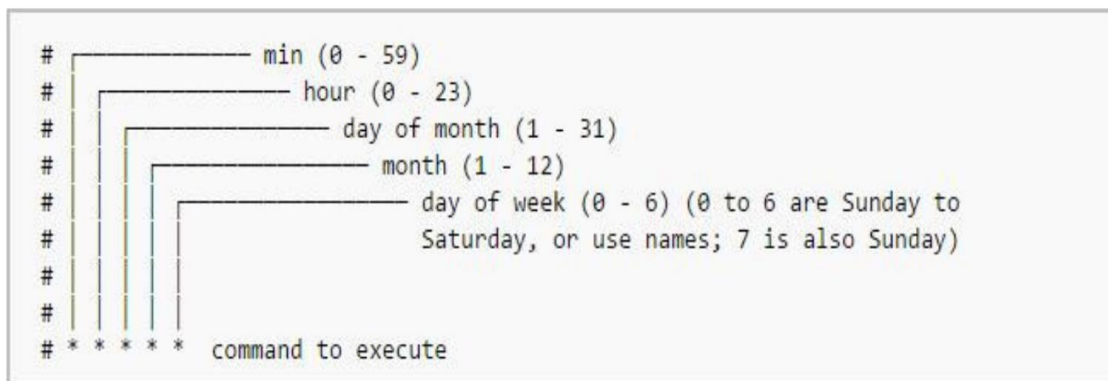
A `crontab` file is a simple plain text file in which each line represents a job. They are usually in the `/etc` folder (or its subdirectories), and each user has their own `crontab` file. `Crontab` files contain all of the scheduled jobs of a particular user. It is a text file that each user can edit and add their tasks to.

A `cron` job is a scheduled task that runs automatically at specified intervals. To set up a `cron` job that runs at 9 a.m. every day, you need to edit your `crontab` file.

`Cron` is installed by default, so we do not have to install it. However, we may have to create a `crontab` file for the current user.

## Format for Crontab

Each line must have five fields, separated by a space - followed by the command or the script to execute. It looks like this:



Open the crontab file by using the command **crontab -e**

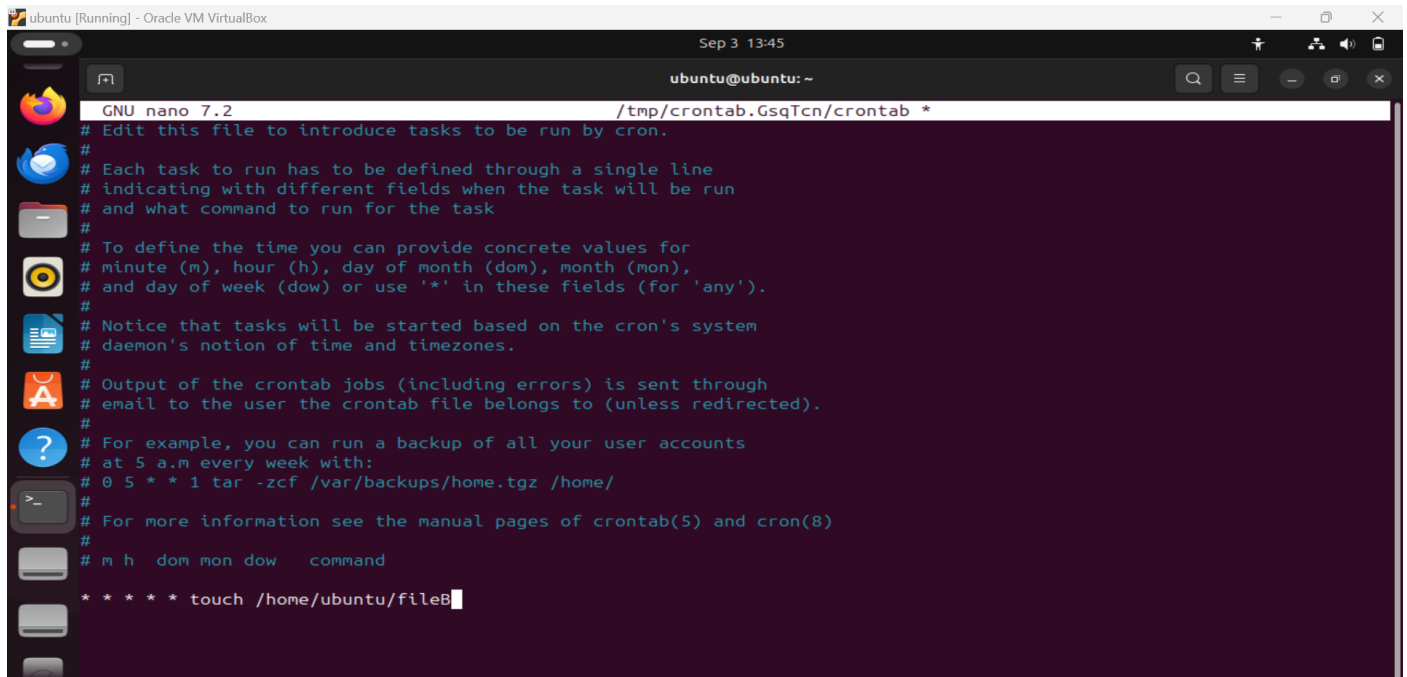
After using the crontab -e it will ask for **editor** to use. Select the editor as per your need.

```
no crontab for admin - using an empty one

Select an editor. To change later, run 'select-editor'.
 1. /bin/nano          <---- easiest
 2. /usr/bin/vim.basic
 3. /usr/bin/vim.tiny

Choose 1-3 [1]: 1|
```

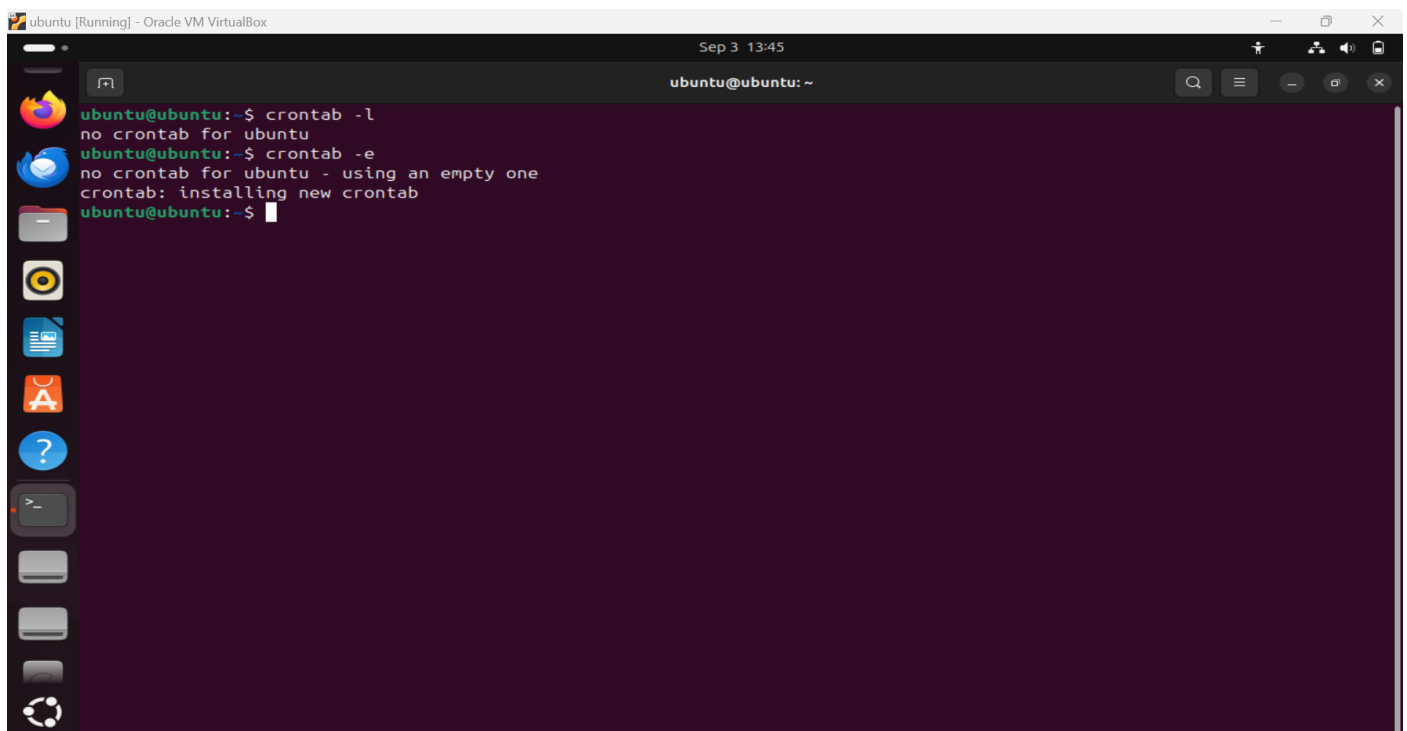
**\* \* \* \* \*** **touch /home/ubuntu/fileB** this command will run every minute of every hour of every day of every month, and every day of the week to create the file.



The screenshot shows a terminal window titled 'ubuntu [Running] - Oracle VM VirtualBox' with a timestamp of 'Sep 3 13:45'. The terminal is running the nano text editor, editing the file '/tmp/crontab.GsqTcn/crontab \*'. The editor's status bar at the top indicates 'GNU nano 7.2'. The content of the file being edited is as follows:

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# 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
* * * * * touch /home/ubuntu/fileB
```

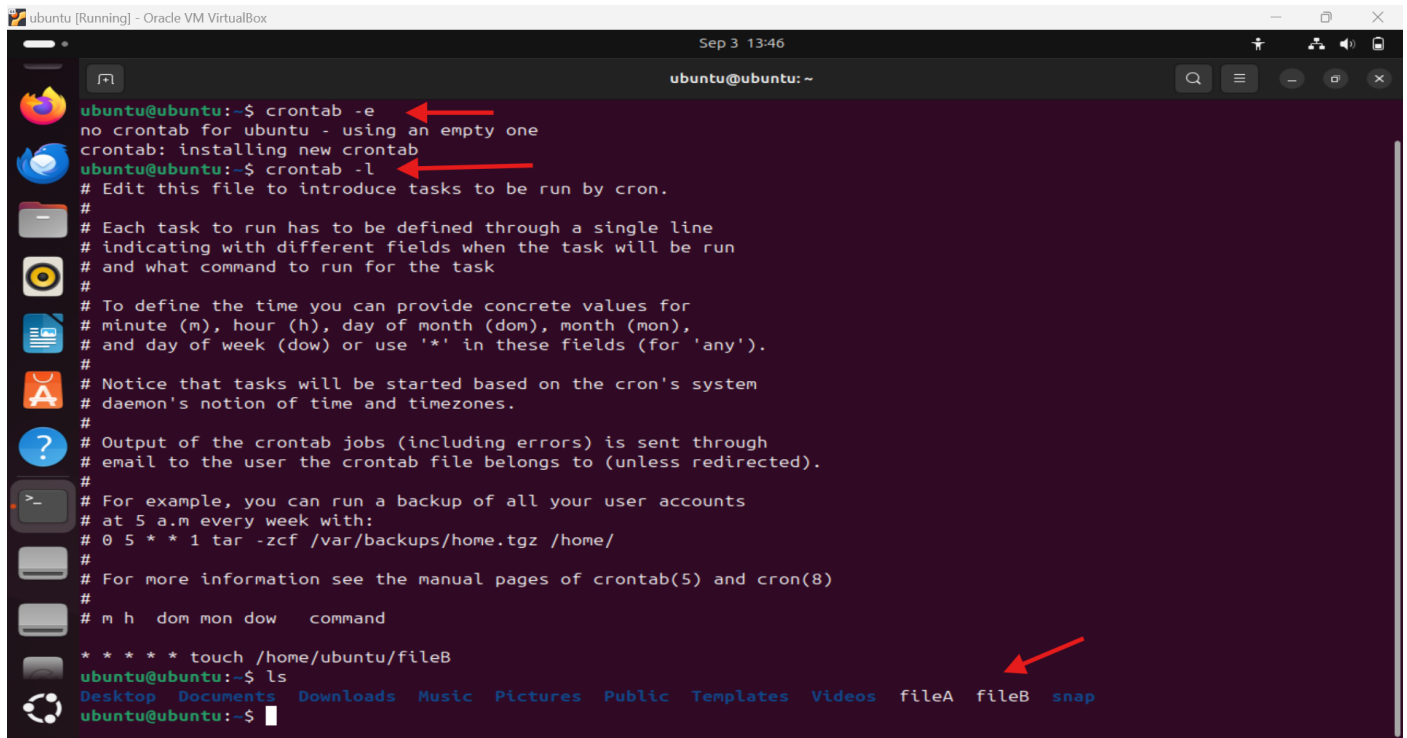
This will create a file called fileB in our home directory.



The screenshot shows the same terminal window as before, but now the nano editor is closed. The user has entered the following commands and received the following output:

```
ubuntu@ubuntu:~$ crontab -l
no crontab for ubuntu
ubuntu@ubuntu:~$ crontab -e
no crontab for ubuntu - using an empty one
crontab: installing new crontab
ubuntu@ubuntu:~$
```

The file just created should be visible. In home directory, check if the file has been created.

A terminal window titled 'ubuntu [Running] - Oracle VM VirtualBox' with a timestamp of 'Sep 3 13:46'. The prompt is 'ubuntu@ubuntu: ~'. The user enters 'crontab -e', which results in 'no crontab for ubuntu - using an empty one' and 'crontab: installing new crontab'. Then, the user enters 'crontab -l', which shows a sample crontab file with various comments and a task: '\* \* \* \* \* touch /home/ubuntu/fileB'. Finally, the user enters 'ls', and the output shows 'Desktop Documents Downloads Music Pictures Public Templates Videos fileA fileB snap'. Red arrows point to the 'crontab -e' command, the 'crontab -l' command, and the 'fileB' in the 'ls' output.

```
ubuntu@ubuntu:~$ crontab -e
no crontab for ubuntu - using an empty one
crontab: installing new crontab
ubuntu@ubuntu:~$ crontab -l
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# 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
* * * * * touch /home/ubuntu/fileB
ubuntu@ubuntu:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos  fileA  fileB  snap
ubuntu@ubuntu:~$
```

Whether you're managing server tasks or developing scripts, mastering cron and at can significantly enhance your productivity.