Set up automatic mode in Linux with Crontab

Cron is a powerful scheduler that allows users to schedule scripts or commands and run them on a regular basis. Users schedule scripts to run in crontab file.

Cron is a powerful timer that allows you to schedule commands and scripts and run them on a regular basis. The script is scheduled to run in the **crontab** file **.** This file may be a bit "scary" when first seen, but once you get used to it, you can catch it quite easily.

In some Linux distributions, like Arch Linux, cron is not installed by default because they have used **systemd** instead of **cron.** In addition, there are different methods of using cron and each option has a way of editing or scheduling own work. This tutorial will focus on cronie, the default option in most Linux distributions, like Ubuntu.

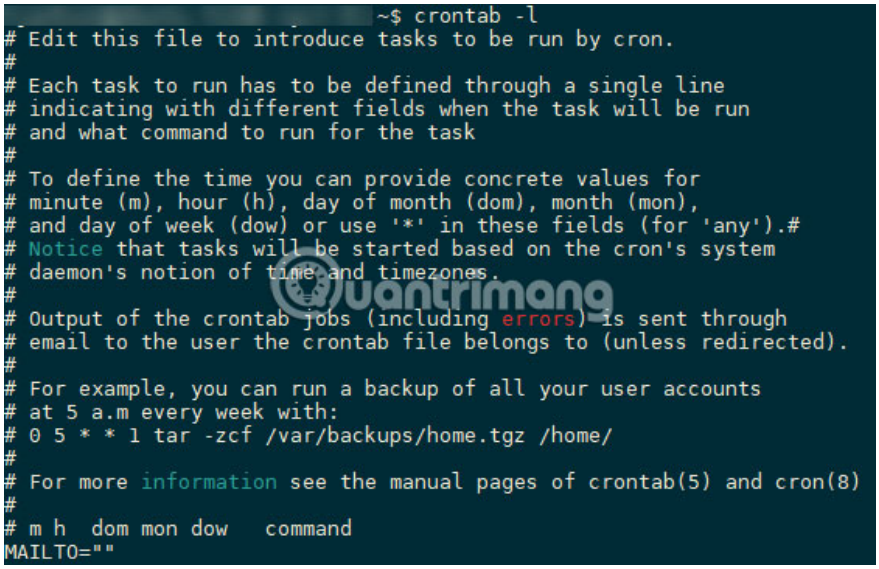
Use Cron to schedule tasks in Linux

1. See cron items
2. Edit cron entries
3. Short codes
4. Edit cron entries as another user
5. Review cron activity

See cron items

See crontab allows users to view scheduled tasks that will run in the account.

```
crontab -l
```



Users can also view scheduled tasks for the root account using **sudo.**

```
sudo crontab -l
```

Edit cron entries

If you have never set up a task in cron, this file will be empty or have some comments. To add an item, open the crontab file using the **-e** option **.**

```
crontab -e
```

Crontab entries will always follow the same syntax, allowing users to check items on any machine as well as create them programmatically.

The basic syntax for crontab input expression is:

```
mhd mon dow command
```

1. **m** = Minutes (between 0 and 59)
2. **h** = Hour (from 0 to 23)
3. **d** = Day of the month (from 1 to 31)
4. **mon** = Month (from 1 to 12)
5. **dow** = Day of the week (from 0 to 6). 0 is Sunday.
6. **Command** = Run command.

For example, you need to run a regular command to check if mysql server is running well. To execute the command to run for hours, use the following syntax:

```
0 * * * * mysqlcheck --all-databases --check-only-changed --silent
```

This means the command will run every hour. If you only want it to run at 1:42 am every day, the command will become:

```
42 1 * * * mysqlcheck --all-databases --check-only-changed --silent
```

In addition, commas can be used to list many items (such as 1, 3, 5) and hyphens can be used for ranges. The use of a special asterisk in the position of an item in the expression indicates that the item will run for all values ??of the field. (An asterisk in the hour field means that the command will run every hour, from 0 - 23.) The pound sign (#) on a line indicates this is a comment (meaning that the code line will not be executed).

Finally, a forward slash will indicate a step value (value increases after each iteration). For example, every 5 minutes will be indicated as * / 5 (since the crontab is saved and the recurring task is created). If only '5' is used, the entry will only run at 5 minutes.

For example:

```
34 3 * * 0 mysqlcheck --all-databases --check-only-changed --silent
```

1. **34** : Minutes to run between 0 and 59
2. **3** : Hour, from 0 to 23
3. ***** : Two middle asterisks indicate the command will run every day (1 to 31) and each month (1 to 12)
4. **0** : The last number 0 indicates the date the command will run on Sunday

This task will run at 3:34 am every month, every week on Sunday.

Short codes

Crontab allows users to use some short codes to make cron entries easier to read.

1. **@reboot** - Run once, on startup
2. **@yearly** - Run once a year, '0 0 1 1 *'.
3. **@annual** - Like @yearly
4. **@monthly** - Run once a month, '0 0 1 * *'
5. **@weekly** - Run once a week, '0 0 * * 0'
6. **@daily** - Run once a day, '0 0 * * *'
7. **@midnight** - like @daily
8. **@hourly** - Run every hour, '0 * * * *'

For example:

```
@hourly mysqlcheck --all-databases --check-only-changed --silent
```

Edit cron entries as another user

There are times when users need to add a crontab with root privileges. For example, to automatically apply Let's Encrypt SSL certificate, run the renewal script with root privileges. To edit crontab as root, simply add **sudo** before the command:

```
sudo crontab -e
```

Similarly, it is possible to schedule jobs in cron with the role of another user, using the **-u** flag **:**

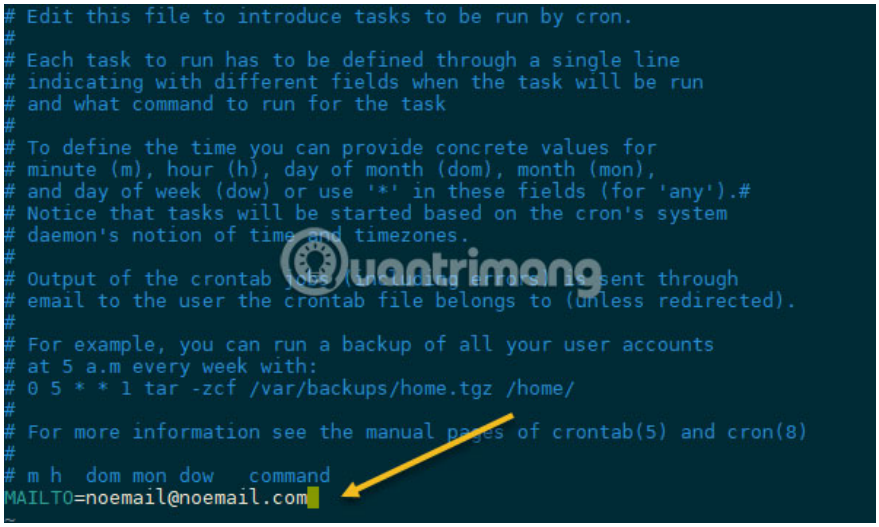
```
sudo crontab -u username -e
```

For example, to run a recurring job as a **'www-data'** user, use the following command:

```
sudo crontab -u www-data -e
```

Review cron activity

Now specific commands or scripts have been set up, but users need to make sure they run. Integration into cron means that once the command is run, cron will email the owner. This can be changed with **MAILTO** variable **.**



Adding **MAILTO=youremail@yourdomain.com** will send all recurring work reports to a specified email. This variable can often be found at the top of the crontab editing screen. However, if it is not there, users can add variables themselves and it will work as expected.

Many email addresses can be separated by commas. If you need an email sent to another location, add the **MAILTO** command directly above the command. The following commands **MAILTO** will be sent to the new address. If **MAILTO =** left blank, the message will be sent to the owner of the cron section.

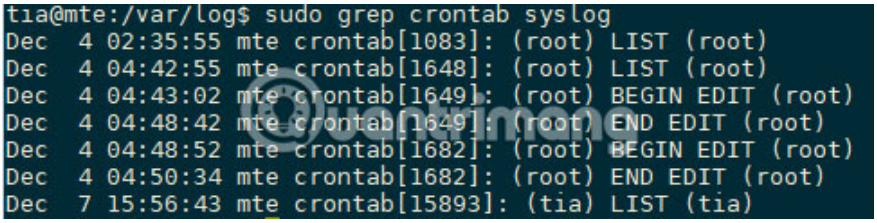
Alternatively, you can use the redirection operator (>) to send output when needed.

```
34 3 * * 0 mysqlcheck --all-databases --check-only-changed --silent > /dev/null
```

The above command will redirect the output to **'/ dev / null',** without emailing and removing the output.

If you want to ensure a job has been done but do not want to receive email, users can also check the cron log. On most systems, accessing the cron log will require **superuser** permissions **.** Cron logs can be found in **'/ var / log'.** The cron or syslog file will display the logs of the cron entries.

```
sudo grep crontab syslog
```



Cron can be further narrowed to allow specific uses and default options can be set. In summary, cron is a tool that provides users with the ability to run tasks comfortably and to ensure that system components are regularly maintained at the discretion of the user.