Schedule Tasks on Linux Using Crontab

4 minute read

If you've got a website that's heavy on your web server, you might want to run some processes like generating thumbnails or enriching data in the background. This way it can not interfere with the user interface. Linux has a great program for this called cron. It allows tasks to be automatically run in the background at regular intervals. You could also use it to automatically create backups, synchronize files, schedule updates, and much more. Welcome to the wonderful world of crontab.

Crontab

The crontab (cron derives from *chronos*, Greek for time; tab stands for *table*) command, found in Unix and Unix-like operating systems, is used to schedule commands to be executed periodically. To see what crontabs are currently running on your system, you can open a terminal and run:

```
$ sudo crontab -l
```

To edit the list of *cronjobs* you can run:

```
$ sudo crontab -e
```

This wil open a the default editor (could be vi or pico, if you want you can change the default editor) to let us manipulate the crontab. If you save and exit the editor, all your cronjobs are saved into crontab. Cronjobs are written in the following format:

```
* * * * * /bin/execute/this/script.sh
```

Scheduling explained

As you can see there are 5 stars. The stars represent different date parts in the following order:

- minute (from 0 to 59)
- hour (from 0 to 23)
- day of month (from 1 to 31)
- month (from 1 to 12)
- day of week (from 0 to 6) (0=Sunday)

Execute every minute

If you leave the star, or asterisk, it means **every**. Maybe that's a bit unclear. Let's use the the previous example again:

```
* * * * * /bin/execute/this/script.sh
```

</>

They are all still asterisks! So this means execute `/bin/execute/this/script.sh`:

- **every** minute
- of **every** hour
- of **every** day of the month
- of **every** month
- and **every** day in the week.

In short: This script is being executed every minute. Without exception.

Execute every Friday 1AM

So if we want to schedule the script to run at 1AM every Friday, we would need the following cronjob:

```
0 1 * * 5 /bin/execute/this/script.sh
```

Get it? The script is now being executed when the system clock hits:

- minute: `0`
- of hour: `1`
- of day of month: `*` (every day of month)
- of month: `*` (every month)
- and weekday: `5` (=Friday)

Execute on workdays 1AM

So if we want to schedule the script to Monday till Friday at 1 AM, we would need the following cronjob:

```
0 1 * * 1-5 /bin/execute/this/script.sh
```

Get it? The script is now being executed when the system clock hits:

- minute: `0`
- of hour: `1`
- of day of month: `*` (every day of month)
- of month: `*` (every month)
- and weekday: `1-5` (=Monday til Friday)

Execute 10 past after every hour on the 1st of every month

Here's another one, just for practicing

```
10 * 1 * * /bin/execute/this/script.sh
```

Fair enough, it takes some getting used to, but it offers great flexibility.

Neat scheduling tricks

What if you'd want to run something every 10 minutes? Well you could do this:

```
0,10,20,30,40,50 * * * * /bin/execute/this/script.sh
```

But crontab allows you to do this as well:

```
*/10 * * * * /bin/execute/this/script.sh
```

Which will do exactly the same. Can you do the the math? ;)

Special words

For the first (minute) field, you can also put in a keyword instead of a number:

```
@reboot     Run once, at startup
@yearly     Run once  a year     "0 0 1 1 *"
@annually   (same as  @yearly)
@monthly    Run once  a month    "0 0 1 * *"
@weekly     Run once  a week     "0 0 * * 0"
@daily      Run once  a day      "0 0 * * *"
@midnight   (same as  @daily)
@hourly     Run once  an hour    "0 * * * *"
```

Leaving the rest of the fields empty, this would be valid:

```
@daily /bin/execute/this/script.sh
```

Storing the crontab output

By default cron saves the output of `/bin/execute/this/script.sh` in the user's mailbox (root in this case). But it's prettier if the output is saved in a separate logfile. Here's how:

```
*/10 * * * * /bin/execute/this/script.sh >> /var/log/script_output.log 2>&1
```

Explained

Linux can report on different levels. There's standard output (STDOUT) and standard errors (STDERR). STDOUT is marked 1, STDERR is marked 2. So the following statement tells Linux to store STDERR in STDOUT as well, creating one datastream for messages & errors:

```
2>&1
```

Now that we have 1 output stream, we can pour it into a file. Where `>` will overwrite the file, `>>` will append to the file. In this case we'd like to to append:

```
>> /var/log/script_output.log
```

Mailing the crontab output

By default cron saves the output in the user's mailbox (root in this case) on the local system. But you can also configure crontab to forward all output to a real email address by starting your crontab with the following line:

```
MAILTO="yourname@yourdomain.com"
```

Mailing the crontab output of just one cronjob

If you'd rather receive only one cronjob's output in your mail, make sure this package is installed:

```
$ aptitude install mailx
```

And change the cronjob like this:

```
*/10 * * * * /bin/execute/this/script.sh 2>&1 | mail -s "Cronjob ouput" yourname@yourdomain.com
```

Trashing the crontab output

Now that's easy:

```
*/10 * * * * /bin/execute/this/script.sh > /dev/null 2>&1
```

Just pipe all the output to the null device, also known as the black hole. On Unix-like operating systems, `/dev/null` is a special file that discards all data written to it.

Caveats

Many scripts are tested in a Bash environment with the `PATH` variable set. This way it's possible your scripts work in your shell, but when run from cron (where the `PATH` variable is different), the script cannot find referenced executables, and fails.

It's not the job of the script to set `PATH`, it's the responsibility of the caller, so it can help to `echo $PATH`, and put `PATH=<the result>` at the top of your cron files (right below `MAILTO`).

Categories: crontab linux

Updated: July 29, 2007

LEAVE A COMMENT RIGHT HERE

We were unable to load Disqus. If you are a moderator please see our troubleshooting guide.