Home Help Linux and Unix

Linux crontab command

Updated: 12/29/2017 by Computer Hope

ComputerHope.com

About crontab

The crontab (short for "cron table") is a list of commands that are scheduled to run at regular time intervals on your computer system.
The **crontab**command opens the crontab for editing, and lets you add, remove, or modify scheduled tasks.

The daemon which reads the crontab and executes the commands at the right time is called cron. It's named after Kronos, the Greek god of time.

Command syntax

```
crontab [-u user] file
```

```
crontab [-u user] [-l | -r | -e] [-i] [-s]
```

Options

file	Load the crontab data from the specified file. If *file* is a dash ("**-**"), the crontab data is read from standard input.
-u *user*	Specifies the user whose **crontab** is to be viewed or modified. If this option is not given, **crontab** opens the crontab of the user who ran **crontab**. Note: using **su** to switch users can confuse **crontab**, so if you are running it inside of **su**, always use the **-u** option to avoid ambiguity.
-l	Display the current crontab.
-r	Remove the current crontab.

-e	Edit the current crontab, using the editor specified in the environment variable **VISUAL** or **EDITOR**.
-i	Same as **-r**, but gives the user a yes/no confirmation prompt before removing the crontab.
-s	SELinux only: appends the current SELinux security context string as an **MLS_LEVEL**setting to the crontab file before editing or replacement occurs. See your SELinux documentation for detailed information.

Overview

The **crontab** command is used to view or edit the table of commands to be run by **cron**.

Each user on your system can have a personal crontab.

Crontab files are located in **/var/spool/** (or a subdirectory such as **/var/spool/cron/crontabs**), but they are not intended to be edited directly. Instead, they are edited by running **crontab**.

Cron command entries

Each **cron** command entry in the crontab file has five time and date fields (followed by a user name, only if it is the system crontab file), followed by a command.

Commands are executed by **cron** when the minute, hour, and month fields match the current time, and at least one of the two day fields (day of month, or day of week) match the current day.

The **cron** daemon checks the crontab once every minute.

Note: Nonexistent times, such as "missing hours" during daylight savings "Spring forward" days, will never match. This causes jobs scheduled during the "missing times" not to run during those times. For the same reason, times that occur more than once during daylight savings (in the Autumn) will cause matching jobs to run twice.

Time and date fields

field	allowed values
minute	**0**-**59**
hour	**0**-**23**
day of month	**1**-**31**
month	**1**-**12** (or names; see example below)
day of week	**0**-**7** (**0** or **7** is Sunday, or use names; see below)

Any of these fields can be set to an asterisk (*****), which stands for "first through last". For instance, to run a job every month, put ***** in the Month field.

Ranges of numbers are allowed. Ranges are two numbers separated with a hyphen. The specified range is inclusive; for example, **8-11** for an "hours" entry specifies execution at hours 8, 9, 10 and 11.

Lists are allowed. A list is a set of numbers (or ranges) separated by commas. Examples: "**1,2,5,9**", "**0-4,8-12**".

Step values can be used in conjunction with ranges. For example, "**0-23/2**" in the Hours field means "every other hour." Steps are also permitted after an asterisk, so if you want to say "every two hours", you can use "***/2**".

Names can also be used for the "month" and "day of week" fields. Use the first three letters of the particular day or month (case doesn't matter). Ranges or lists of names are not allowed.

The "sixth" field (the rest of the line) specifies the command to be run. The entire command portion of the line, up to a newline or **%** character, will be executed by **/bin/sh** or by the shell specified in the SHELL variable of the cronfile. Percent-signs (**%**) in the command, unless escaped with backslash (\), will be changed into newline characters, and all data after the first **%** will be sent to the command as standard input.

Note that the day of a command's execution can be specified by two fields: day of month, and day of week. If both fields are restricted (in other words, they aren't *****), the command will be run when either field matches the current time. For example, "**30 4 1,15 * 5**" would cause a command to be run at 4:30 am on the 1st and 15th of each month, plus every Friday.

The crontab file

Each line of a crontab file is either "active" or "inactive". An "active" line is an environment setting, or a cron command entry. An "inactive" line is anything ignored, including comments.

Blank lines and leading spaces and tabs are ignored. Lines whose first non-space character is a pound-sign (**#**) are interpreted as comments, and are ignored. Comments are not allowed on the same line as **cron** commands, because they will be interpreted as part of the command. For the same reason, comments are not allowed on the same line as environment variable settings.

Environment settings

An environment setting line in the crontab can set environment variables for whenever cron runs a job.

Tip: Not every system's crontab can include environment settings. On Ubuntu and Debian, and systems that use GNU **mcron**, environment settings can be made in the crontab. On other systems, such as Arch Linux and Fedora, environment settings in the crontab are not allowed. Check your distribution's cron documentation for more information.

An environment setting in the crontab is formatted as:

```
name = value
```

The spaces around the equal sign (**=**) are optional, and any subsequent non-leading spaces in *value* will be part of the value assigned to *name*. The value string may be placed in quotes (single or double, but matching) to preserve leading or trailing blanks.

Some environment variables are set automatically by **cron**:

> **SHELL** is set to **/bin/sh**.
>
> **LOGNAME** and **HOME** are set from the **/etc/passwd** line of the crontab's owner. **HOME** and **SHELL** may be overridden at runtime by settings in the crontab; **LOGNAME** may not.
>
> The **LOGNAME** variable is sometimes called **USER** on BSD systems. On these systems, **USER** will be set also.

Configuration

Permitting users to run cron jobs

Cron jobs can be allowed or disallowed for individual users, as defined in the files **/etc/cron.allow** and **/etc/cron.deny**. If **cron.allow** exists, a user must be listed there to be allowed to use a given command. If the **cron.allow** file does not exist but the **cron.deny** file does, then a user must *not* be listed there to use a given command. If neither of these files exists, only the superuserwill be allowed to use a given command.

Cron permissions can also be defined using PAM (pluggable authentication module) authentication to set up users who may or may not use **crontab** and system **cron** jobs. PAM configuration is located in **/etc/cron.d/**.

Configuring the temp directory

The temporary directory for cron jobs can be set in environment variables listed below. If these variables are not defined, the default temporary directory **/tmp** is used.

Configuration files

File	Description
/etc/cron.allow	If this file exists, users must be listed in this file to be able to run cron jobs.
/etc/cron.deny	If this file exists, users must *not* be listed in this file to be able to run cron jobs.

If neither configuration file exists, only the superuser may run cron jobs.

Examples

Running crontab

```
crontab -e
```

Edit your crontab.

```
crontab -l
```

Display ("list") the contents of your crontab.

```
crontab -r
```

Remove your crontab, effectively un-scheduling all crontab jobs.

```
sudo crontab -u charles -e
```

Edit the crontab of the user named **charles**. The **-u** option requires administrator privileges, so the command is executed using **sudo**.

```
sudo crontab -l -u jeff
```

View the crontab of user **jeff**.

```
sudo crontab -r -u sandy
```

Remove the crontab of user **sandy**.

Crontab entries

The following are examples of entries which could be included in a crontab.

Run the shell script **/home/melissa/backup.sh** on January 2 at 6:15 A.M:

```
15 6 2 1 * /home/melissa/backup.sh
```

Days and months can be listed by name (**Monday**) or abbreviation (**Jan**). Zeroes at the beginning of a number are valid, which can help you make multiple entries line up visually.

ee
For instance, the next example runs the same script as above, at 12:01 AM, every Monday in January:

```
01 00 * Jan Monday /home/melissa/backup.sh
```

Run **/home/carl/hourly-archive.sh** every hour, on the hour, from 9 A.M. (**09**:**00**) through 6 P.M. (**18**:**00**), every day:

```
00 09-18 * * * /home/carl/hourly-archive.sh
```

Run **/home/wendy/script.sh** every Monday, at 9 A.M. and 6 P.M:

```
0 9,18 * * Mon /home/wendy/script.sh
```

Run **/usr/local/bin/backup** at 10:30 P.M., every weekday:

```
30 22 * * Mon,Tue,Wed,Thu,Fri /usr/local/bin/backup
```

Related commands

at — Schedule a command to be run at a certain time.