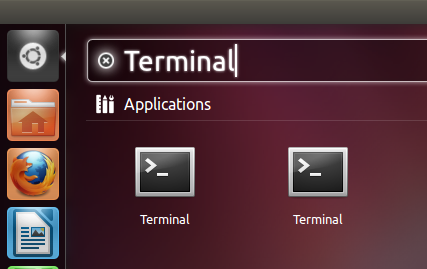
**Crontab**

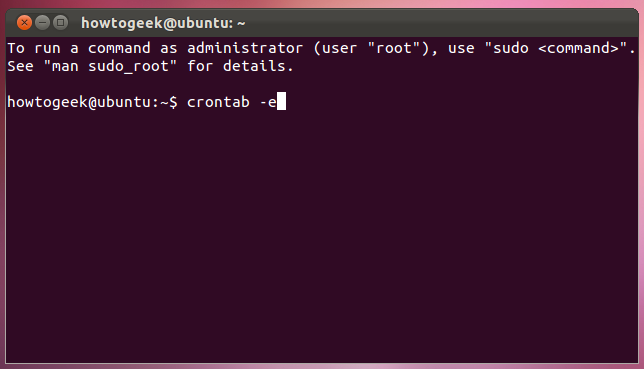
The cron daemon on Linux runs tasks in the background at specific times; it’s like the Task Scheduler on Windows. Add tasks to your system’s crontab files using the appropriate syntax and cron will automatically run them for you.

Crontab files can be used to automate backups, system maintenance and other repetitive tasks. The syntax is powerful and flexible, so you can have a task run every fifteen minutes or at a specific minute on a specific day every year.

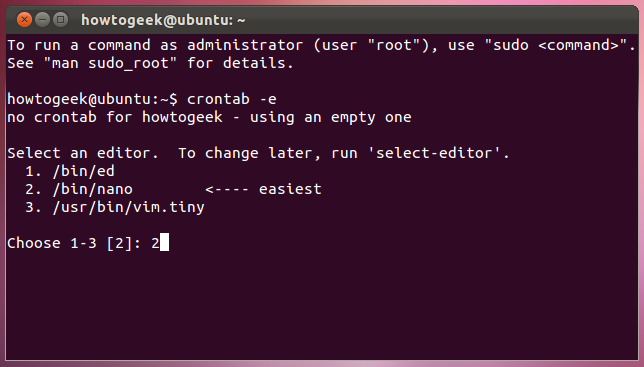
**Opening Crontab**



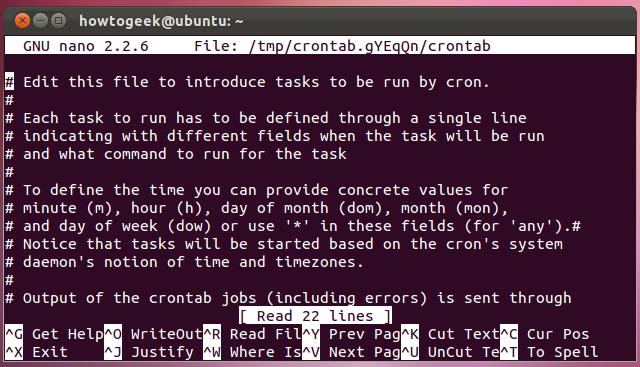
First, open a terminal window from your Linux desktop’s applications menu. You can click the Dash icon, type Terminal and press Enter to open one if you’re using Ubuntu.



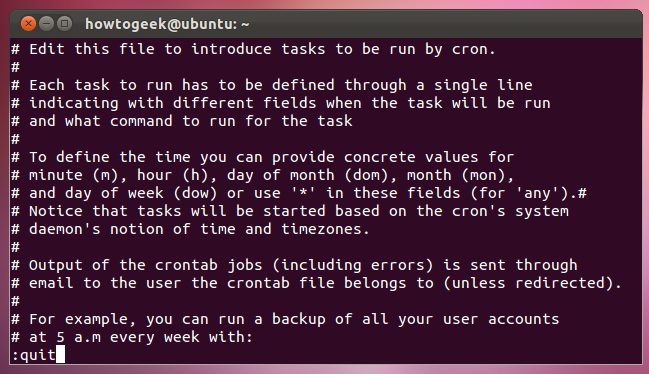
Use the **crontab -e** command  to open your user account’s crontab file. Commands in this file run with your user account’s permissions. If you want a command to run with system permissions, use the **sudo crontab -e** command to open the root account’s crontab file. Use the **su -c “crontab -e”** command instead if your Linux distribution doesn’t use sudo.



You may be asked to select an editor. Select Nano if it’s available by typing its number and pressing Enter. Vi and other more advanced editors may be preferred by advanced users, but Nano is an easy editor to get started with.



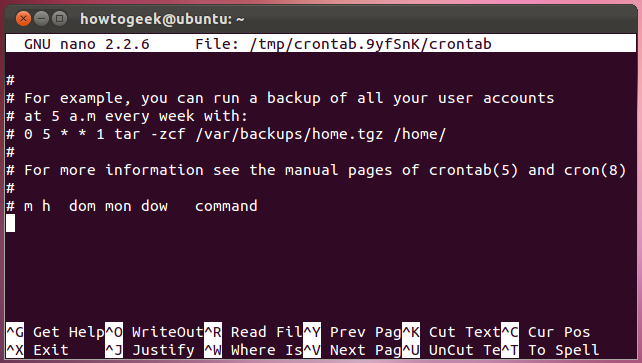
You’ll see the Nano text editor, identified by the “GNU nano” header at the top of your terminal window. If you don’t, crontab probably opened in the vi text editor.



If you’re not comfortable using vi, you can type **:quit** into vi and press Enter to close it. Run the **export EDITOR=nano** command, then run **crontab -e** again to open the crontab file in Nano.

**Adding New Tasks**

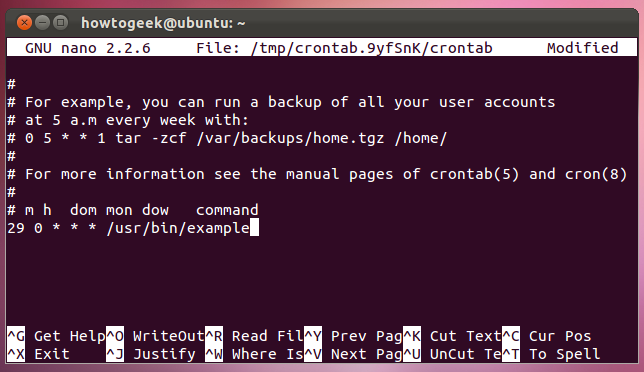
Use the arrow keys or the page down key to scroll to the bottom of the crontab file in Nano. The lines starting with # are comment lines, which means that cron ignores them. Comments just provide information to people editing the file.



Lines in the crontab file are written in the following sequence, with the following acceptable values:

minute(0-59) hour(0-23) day(1-31) month(1-12) weekday(0-6) command

You can use an asterisk (\*) character to match any value. For example, using a asterisk for the month would cause the command to run every month.



For example, let’s say we want to run the command /usr/bin/example at 12:30 a.m. every day. We’d type:

29 0 \* \* \* /usr/bin/example

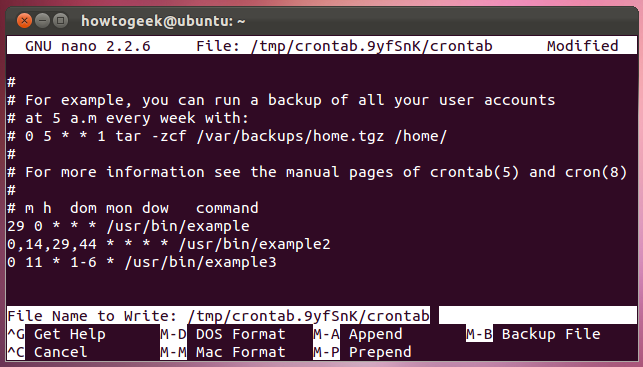
We use 29 for the 30-minute mark and 0 for 12 a.m. because the minute, hour and weekday values start at 0. Note that the day and month values start at 1 instead of 0.

**Multiple Values and Ranges**

Use comma-separated values to specific multiple times. For example, the line

0,14,29,44 \* \* \* \* /usr/bin/example2

runs /usr/bin/example2 at the 15-minute mark on every hour, every day. Make sure you add each new task on a new line.



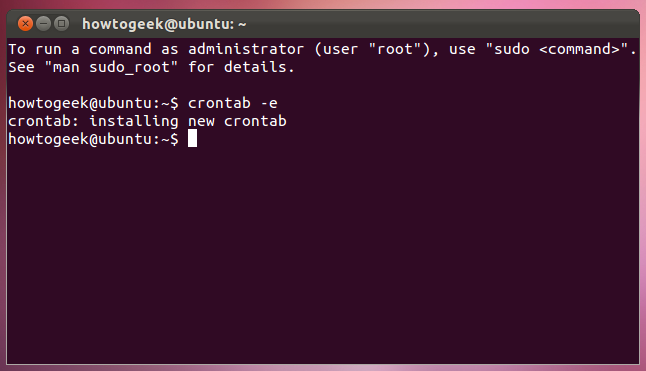
Use dash-separated values to specify a range of values. For example, the line

0 11 \* 1-6 \* /usr/bin/example3

runs /usr/bin/example3 at noon every day, but only in the first six months of the year.

**Saving the File**

Press Ctrl-O and press Enter to save the crontab file in Nano. Use the Ctrl-X shortcut to close Nano after you’ve saved the file.



You’ll see the “crontab: installing new crontab” message, indicating that your new crontab file was installed successfully.