Scheduled Tasks

The Totara 'cron' process is a PHP script (part of the standard Totara installation) that must be run regularly in the background. It's strongly recommended that the frequency of cron is set to be at least once per minute. The Totara cron script runs different tasks at differently scheduled intervals, which can be configured by admin.

> Do not skip setting up the cron process on your server for your Totara. Your site will not work properly without it.

A special program, 'cron', is used to run the Totara cron script at a regular interval. The Totara cron script runs tasks including sending mail, updating Totara reports, RSS feeds, activity completions, posting forum messages and other tasks. Since different tasks have different schedules, not every task will run in Totara when the cron script is triggered.

The cron program (which runs the Totara script) is a core part of Unix based systems (including Linux and OSX) being used to run many time-dependent services. On Windows the simplest solution is to create a task in the Windows Task Scheduler and set it to run at regular intervals. On shared hosting, you should find the documentation (or ask support) how cron is configured.

The task involves adding a single command to the list of cron activities on your system. On Unix based systems this list is a file called a 'crontab' which all users have.

On this page

Cron command

Totara has two different ways to deploy cron which use different scripts within the Totara install. These are as follows:

- The CLI (command line interpreter) script: This will be at the path **/path/to/Totara/admin/cli/cron.php**. If in doubt, this is the correct script to use. This needs to be run by a 'PHP CLI' program on your computer. So the final command may look something like **/usr/bin/php /path/to/Totara/admin/cli/cron.php**. You can (and should) try this on your command line to see if it works.
- The web based script: This needs to be run from a web browser and will be accessed via a web URL something like **http://your.totara.site/admin/cron.php**. You can find command line based web browser (e.g. wget) so the final command may look like **/usr/bin/wget http://your.totara.site/admin/cron.php**. This has the advantage that it can be run from anywhere. If you can't get cron to work on your machine it can be run somewhere else.

Cron scheduler UI for Site Administrators

Another important feature is the cron scheduler UI for administrators. This enables administrators to know which tasks the cron runs and to troubleshoot when each part was last executed.

Site administrators can take control of the background processes running for their Totara site. This feature brings visibility to the Totara cron making it easier to see what's going on, debug slow or failing processes, and to take control of and optimise cron for a particular site.

Use the scheduler feature located under *Site administration* > *Server* > *Scheduled tasks* to display a list of tasks that get executed by cron. This includes displaying when the task last ran, and when it is expected to next be run.

Each task can also be configured to run at the desired time on any desired day.

System Administrators with command line access can also manually run a single task if required by executing **admin/tool/task/cli/schedule_task.php** with the required arguments.

> Those running their site in a hosted environment where changes to system cron cannot be made or must be put through a request and approval process also benefit from in-product configuration.

Finding the right place to put the command

This depends on the system you are using and you should find and read the documentation for your platform or hosting. In most cases getting the Totara cron to run consists of establishing the correct command (above) and then adding it, and the time to run the command, to a file. This might be either through a specific user interface or by editing the file directly.

If using the CLI version you also need to make sure that the cron process is run as the correct user. This is not an issue with the web version.

Example: Installing cron on Ubuntu/Debian Linux. Assuming logged in as root:

Use the crontab command to open a crontab editor window for the www-data user. This is the user that Apache (the web server) runs as on Debian based systems.

```
$ crontab -u www-data -e
```

This will open an editor window. To run the cli cron script every minute, add the line:

```
*/1 * * * * /usr/bin/php  /path/to/Totara/admin/cli/cron.php >/dev/null
```

Note that the final ">/dev/null" sends all the output to the 'bin' and stops you getting an email every minute.

Recommended settings

Generally speaking, those tasks that run daily are the ones that you are more likely to need to pay close attention to, particularly on larger sites. Performance impacts are going to be dependent on how much information needs to be processed each time this task runs and what timing you're changing the task to e.g. changing an hourly task to one that runs every 30 minutes could be fine, but reducing it to run each and every minute could have an impact.

Its advisable to monitor these task timings for any issues. The cron logs will give an understanding of the time it takes to execute a particular task and based on this adjustments can be made to improve site performance.

As all sites differ and cron performance is dependent on the usage of various features it is difficult to offer a general or 'ideal' configuration.

Logging cron

The cron script output can be logged as it executes. You need to specify the path to PHP and the path to where you want the log file to be written. For Windows you need to set up a scheduled task and include the command. Below are examples of the commands to use for each operating system:

- **Unix/ Linus example:** */1 * * * * /usr/bin/php /path/to/Totara/admin/cli/cron.php >> /var/log/cron.log 2>&1
- **Windows example:** c:/php/php.exe -f c:/totara/admin/cli/cron.php > c:/totara/admin/cron.log

For Windows, if you have problems using the above commends in a text file for logging the output of **cron.php**. Should this be the case you can alternatively try using backslashes like this:

```
@echo off c:\php\php.exe -f c:\path\to\totara\admin\cli\cron.php >> c:
\totara\admin\cron.log
```

Run this in a batch file, for example **c:\totara\cron.cmd**, and then specify the path of this batch file as the task to run.

Note that in the last example we used a double greater than/ arrow symbol (**>>**) rather than the single one in the first example (**>**). The double symbol means that all the cron logs will be kept in the file, rather than being overwritten. This can however mean that the file can grow quite large, so it is important to carry our regular maintenance and clean out unwanted data.

Scheduled tasks purpose

You can find details on the purpose of each scheduled task on the associated component help page (as listed below):

- Programs
- Enrolments