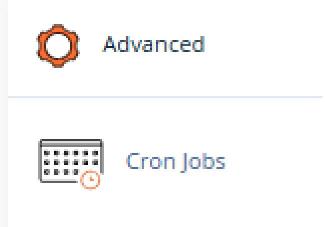
About Cron jobs

Cron jobs in cPanel allow you to automate tasks and execute commands or scripts on your website at scheduled intervals. This can be useful for a variety of administrative tasks, such as clearing temporary files, sending regular reports, or executing database backups. The cron job feature helps you automate processes without manual intervention, ensuring that tasks run on time.

1. Accessing Cron Jobs

To access the Cron Jobs feature:

1. Log in to your cPanel account.
2. Navigate to the **Advanced** section.
3. Click on **Cron Jobs**.



This will open the Cron Jobs interface where you can set up, manage, and monitor automated tasks for your site.

Understanding Cron Jobs

Cron jobs are scheduled tasks that run automatically at specified intervals. You can define the frequency and the exact time when these tasks should be executed. Cron jobs are typically used for running commands, executing scripts, or performing server maintenance tasks at specific times.

How Cron Jobs Work

Cron jobs are scheduled by defining time intervals using five fields:

- **Minute**: The minute when the job should run (0-59).
- **Hour**: The hour when the job should run (0-23).
- **Day**: The day of the month when the job should run (1-31).
- **Month**: The month when the job should run (1-12).
- **Weekday**: The day of the week when the job should run (0-7) where both 0 and 7 represent Sunday.

For example, if you want a script to run at 2:00 AM every day, you would enter:

- **Minute**: 0
- **Hour**: 2
- **Day**: *
- **Month**: *
- **Weekday**: *

Common Settings for Cron Jobs

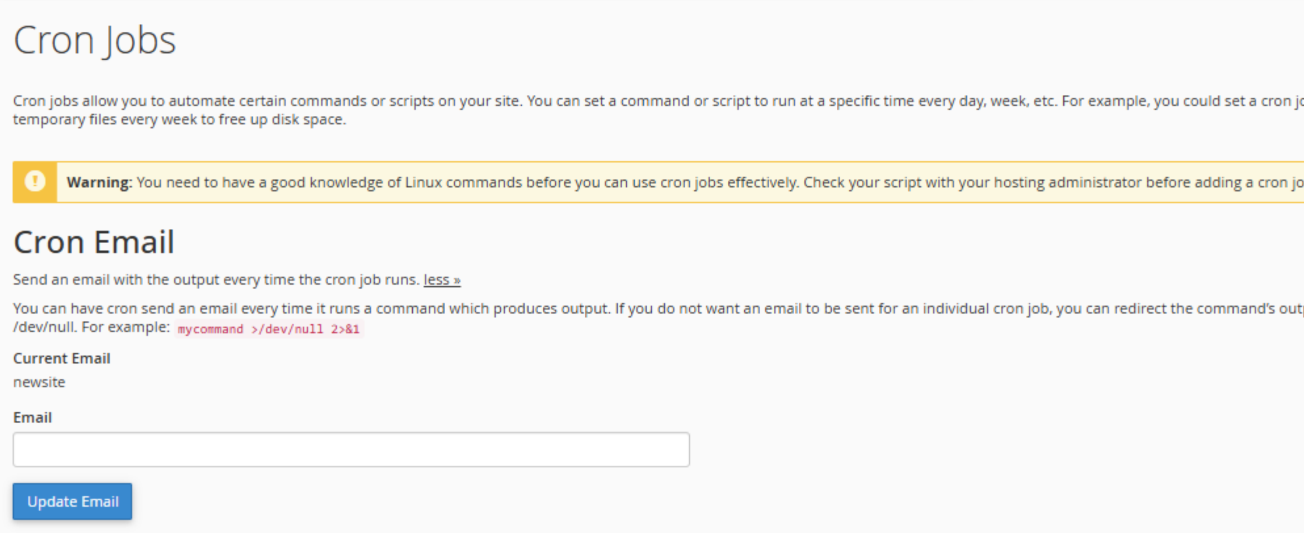
- **Once per day**: If you want to schedule a job to run daily at a specific time, enter the following settings:
    - **Minute**: 0
    - **Hour**: (the hour you want)
    - **Day**: *
    - **Month**: *
    - **Weekday**: *
- **Hourly Cron Job**: If you want the task to run every hour, set:
    - **Minute**: 0
    - **Hour**: *
    - **Day**: *
    - **Month**: *
    - **Weekday**: *
- **Weekly Cron Job**: If you want the task to run on a specific day of the week (e.g., every Monday at 3:00 AM):
    - **Minute**: 0
    - **Hour**: 3
    - **Day**: *
    - **Month**: *
    - **Weekday**: 1 (for Monday)

2. Adding a New Cron Job

To add a new cron job, follow these steps:

1. **Email Settings**:
    - If you want an email sent to you every time the cron job runs, enter your email address in the **Current Email** section. For example, you may see **newsite@example.com** in this field.

- If you don’t want to receive emails, use the command `>/dev/null 2>&1` to suppress email notifications. For example: `/usr/local/bin/php /home/newsite/public_html/path/to/cron/script >/dev/null 2>&1`



Cron Jobs

Cron jobs allow you to automate certain commands or scripts on your site. You can set a command or script to run at a specific time every day, week, etc. For example, you could set a cron jo temporary files every week to free up disk space.

Warning: You need to have a good knowledge of Linux commands before you can use cron jobs effectively. Check your script with your hosting administrator before adding a cron jo

Cron Email

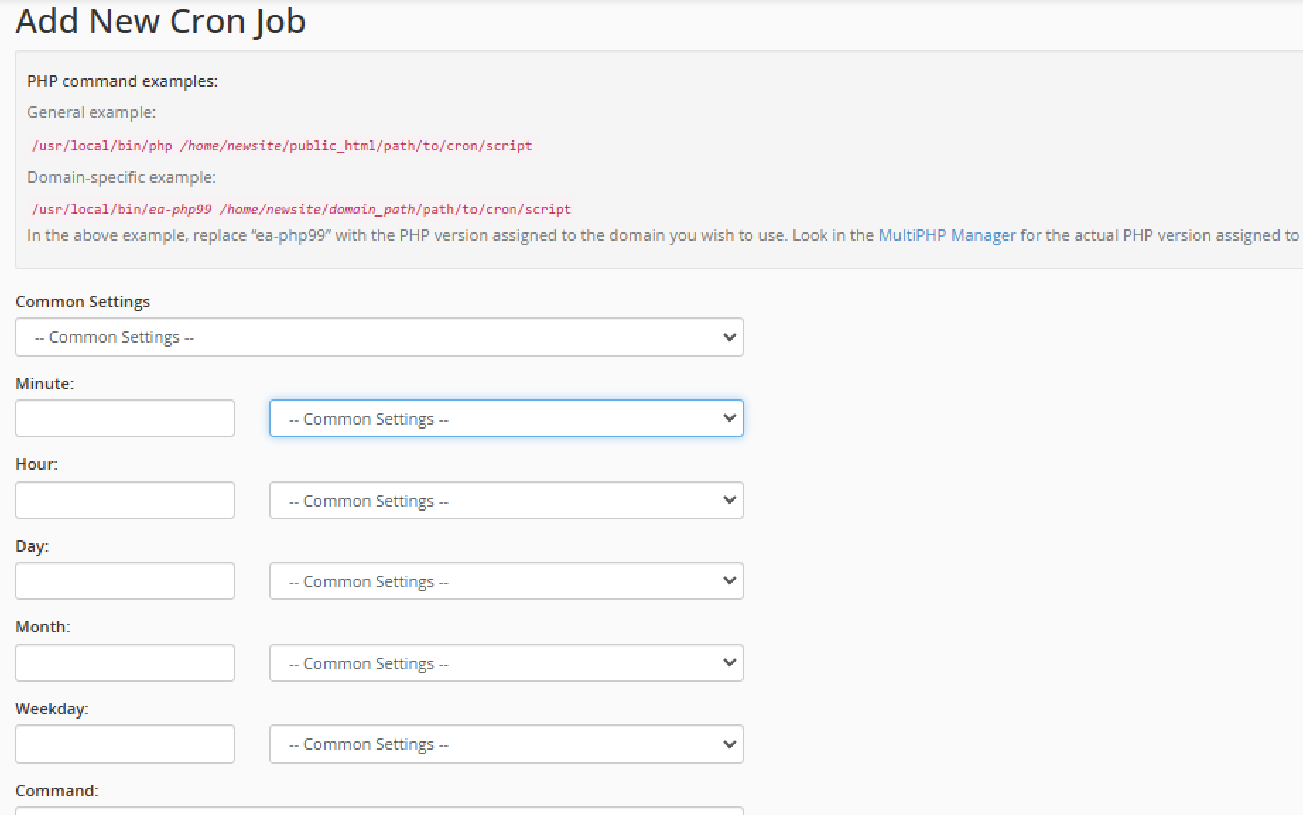
Send an email with the output every time the cron job runs. less »

You can have cron send an email every time it runs a command which produces output. If you do not want an email to be sent for an individual cron job, you can redirect the command’s out /dev/null. For example: `mycommand >/dev/null 2>&1`

Current Email
newsite

Email

2. **Set the Time Intervals**:
   - Use the dropdown menus for **Minute**, **Hour**, **Day**, **Month**, and **Weekday** to specify when the cron job should run.



Add New Cron Job

PHP command examples:

General example:

`/usr/local/bin/php /home/newsite/public_html/path/to/cron/script`

Domain-specific example:

`/usr/local/bin/ea-php99 /home/newsite/domain_path/path/to/cron/script`

In the above example, replace “ea-php99” with the PHP version assigned to the domain you wish to use. Look in the MultiPHP Manager for the actual PHP version assigned to

3. **Command**:
   - In the **Command** field, you’ll enter the command or script that you want to run. Below are some common examples:

General PHP Command:

swift

`/usr/local/bin/php /home/newsite/public_html/path/to/cron/script`

This will run a PHP script located at `/home/newsite/public_html/path/to/cron/script`.

Domain-Specific PHP Command:

swift

```
/usr/local/bin/ea-php99 /home/newsite/domain_path/path/to/cron/script
```

In this example, replace `ea-php99` with the PHP version assigned to the domain. To find the PHP version assigned to the domain, you can check in the **MultiPHP Manager** in cPanel.

4. **Add the Cron Job**: Once you have configured the cron job settings, click **Add New Cron Job** to schedule the task.

Cron Job Email Notifications

- **Cron Email**: When you set up a cron job, you can choose to receive an email each time the cron job runs and produces output. This is useful for monitoring the task's execution.
  - If you don't need an email notification for a specific cron job, you can redirect its output to `/dev/null` to suppress the email notification. Example:

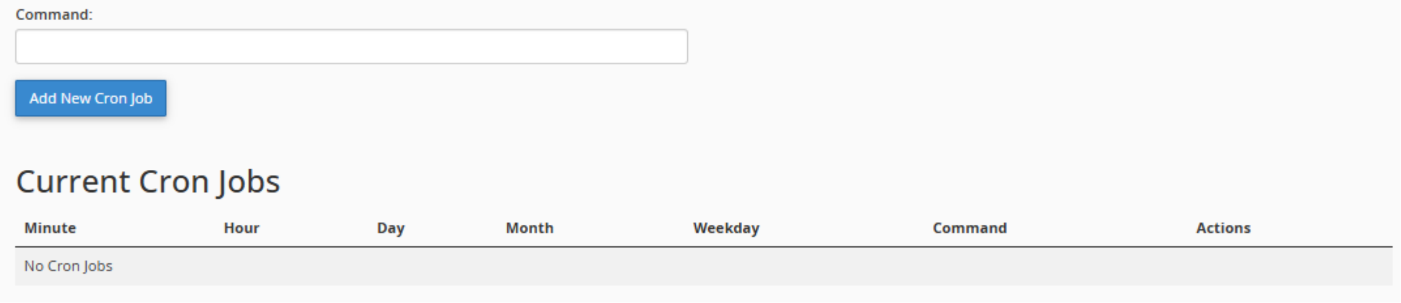
javascript

```
mycommand >/dev/null 2>&1
```

- **Current Email**: The system will send an email to the email address listed in the **Current Email** section of your cPanel. You can change the email address if needed.

3. Managing Cron Jobs

Once you have set up cron jobs, they will appear in the **Current Cron Jobs** table.



Command:

Current Cron Jobs

Minute	Hour	Day	Month	Weekday	Command	Actions
No Cron Jobs						

This table displays the following information:

Minute	Hour	Day	Month	Weekday	Command	Actions
*	2	*	*	*	/usr/local/bin/php /home/newsite/public_html/path/to/cron/script	[Edit] [Delete]

- **Actions**: You can **Edit** or **Delete** an existing cron job by clicking the corresponding link in the **Actions** column. Editing allows you to adjust the time intervals or the command to be executed.

Important Notes About Cron Jobs

- **Linux Knowledge Required**: To use cron jobs effectively, you need to have a basic understanding of Linux commands. Always ensure that the script or command you are

scheduling is correct and tested.
- **Testing Scripts**: Before adding a cron job, you should test the script to make sure it performs as expected when run manually. This helps avoid errors when the job runs automatically.
- **Cron Job Limits**: Some hosting providers may limit the number of cron jobs you can set up or impose restrictions on the frequency at which they can run. Be sure to check with your hosting provider for any limitations.

Cron jobs are an invaluable tool for automating tasks on your website. By setting up cron jobs, you can ensure that tasks like cleaning temporary files, backing up databases, and running scripts happen automatically and on schedule. Always verify your commands and test scripts before scheduling them, and be mindful of your email notifications to keep track of your cron job activities.

Revision #2
Created 4 March 2025 07:54:37 by Admin
Updated 4 March 2025 08:12:25 by Admin