

# EXECUTER AUTOMATIQUEMENT DES SCRIPTS PHP AVEC CRON ET WGET

Version 1.0 Patrick Brunswyck

un manuel édité par

*all2all*

Moving Art Studio a.s.b.l.

Copyright 2009 © Moving Art Studio

GNU Free Documentation Licence

<http://www.gnu.org/copyleft/fdl.html>

*all2all* .beagent



## Table des matières

Exécuter automatiquement des scripts PHP avec cron et wget.....	3
Qu'est-ce que cron.....	3
Syntaxe de cron.....	3
La comande wget.....	3
Employer cron avec Virtualmin.....	4
Créer un petit script de test.....	4
Créer une tâche programmée avec cron.....	5
Lancer une tâche PHP dans cron.....	8
Versions.....	9

# Exécuter automatiquement des scripts PHP avec cron et wget

## Qu'est-ce que cron ?

Cron permet de **programmer dans le temps des tâches** à effectuer automatiquement sur un système d'exploitation de type Unix (comme Linux). **Cron** est une abréviation de Chronograph.

Cron donne aux utilisateurs la possibilité de lancer des commandes, des scripts ou des programmes à une certaine date ou heure prédéfinie.

## Syntaxe de cron

```
.----- minute (0 - 59)
| .----- heure (0 - 23)
| | .----- jour du mois (1 - 31)
| | | .----- mois (1 - 12) OU jan, feb, mar, apr ...
| | | | .---- jour de la semaine (0 - 6) (Dimanche = 0 ou 7) OU sun, mon, tue, wed, thu, fri, sat
| | | | |
* * * * * commande à exécuter
```

Pour plus d'informations sur la syntaxe de cron : <http://fr.wikipedia.org/wiki/Crontab>

## La commande wget

Exécuter un script PHP peut être réalisé en utilisant cron et wget en faisant abstraction du résultat; juste lancer la tâche et la laisser mourir. Pour réaliser cela, inscrivez ce qui suit dans /etc/crontab:

➤ \* \* \* \* \* **wget -q http://mon\_site.be/index.php >/dev/null 2>&1**

L'option **q** ou **-quiet** de Wget désactive le résultat de wget. C'est exactement ce que nous voulons puisque nous n'avons pas l'intention de produire du contenu mais d'**exécuter** seulement le script PHP.

Il est habituel lorsqu'on ajoute une entrée dans crontab de terminer la commande comme suit :

➤ **>/dev/null 2>&1**

Le but de la manoeuvre est de supprimer le produit de la commande elle-même qui ne nous intéresse pas.

➤ La première partie **>/dev/null**:

signifie : redirige STDOUT (standard output stream) vers /dev/null (qui est essentiellement un trou noir qui avale les bits).

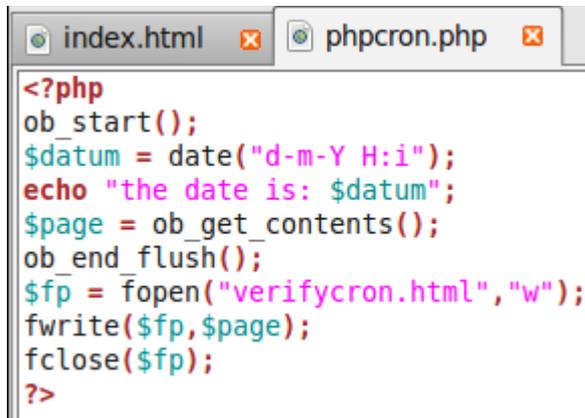
➤ La seconde partie **2>&1**:

signifie : redirige STDERR (standard error stream) vers le même endroit que STDOUT (qui vient d'être spécifié). À STDOUT a été assigné le numéro 1, tandis que le numéro 2 a été assigné à STDERR. Ainsi, tant STDOUT (1) que STDERR (2) sont dirigés vers /dev/null et **tous les produits des commande cron sont supprimés.**

# Employer cron avec Virtualmin

## Créer un petit script de test

Nous pouvons créer un petit script PHP que nous appellerons **phpcron.php**. Ce script va générer un fichier **verifycron.html** pour vérifier que le cron a exécuté la tâche planifiée telle que configurée :



```
<?php
ob_start();
$datum = date("d-m-Y H:i");
echo "the date is: $datum";
$page = ob_get_contents();
ob_end_flush();
$fp = fopen("verifycron.html","w");
fwrite($fp,$page);
fclose($fp);
?>
```

Ce script saisira tous les résultats et les enregistrera dans un nouveau dossier appelé **verifycron.html**. En employant la fonction de date, nous pouvons vérifier la date exacte à laquelle la page verifycron.html a été créée (vous pouvez naturellement juste créer un simple fichier et vérifier son empreinte temporelle pour voir quand il a été créé). De cette façon, nous pouvons être sûrs que les travaux programmés dans le cron sont bien effectués.

Confirmez juste que votre script peut écrire dans le dossier où vous allez sauver votre fichier verifycron.html (vérifiez les droits en écriture) et est exécutable (comme pour tout script).

### But :

- Pour que le cron exécute cette page et qu'il crée le fichier verifycron.html (suivant cet exemple dans le dossier : /var/www/htdocs/patrick/public). Nous employons cet modèle comme cas d'espèce pour des buts plus utiles.
- Ainsi dans cet exemple la commande qui doit figurer dans cron est :  
**wget -q http://patrick.all2all.org/phpcron.php 2>&1**
- Ceci va **exécuter** le fichier phpcron.php, exécuter le script qui s'y trouve et créer une page verifycron.html qui va indiquer sa date et heure de création exacte et ainsi confirmer que le cron a bien exécuté le script comme prévu.

# Créer une tâche programmée avec cron

Allez dans **Virtualmin**, cliquez sur **Webmin Modules** et puis sur **Scheduled Cron Jobs**

Maintenant cliquez sur **Create a new scheduled cron job**

**Job Details**

Execute cron job as

Active?  Yes  No

Command

Input to command

Description

**When to execute**

Simple schedule ..   Times and dates selected below ..

Minutes	Hours	Days	Months	Weekdays
<input type="radio"/> All	<input type="radio"/> All	<input type="radio"/> All	<input type="radio"/> All	<input type="radio"/> All
<input checked="" type="radio"/> Selected ..	<input checked="" type="radio"/> Selected ..	<input checked="" type="radio"/> Selected ..	<input checked="" type="radio"/> Selected ..	<input checked="" type="radio"/> Selected ..
0 12 24 36 48 1 13 25 37 49 2 14 26 38 50 3 15 27 39 51 4 16 28 40 52 5 17 29 41 53 6 18 30 42 54 7 19 31 43 55 8 20 32 44 56 9 21 33 45 57 10 22 34 46 58 11 23 35 47 59	0 12 1 13 2 14 3 15 4 16 5 17 6 18 7 19 8 20 9 21 10 22 11 23	1 13 25 2 14 26 3 15 27 4 16 28 5 17 29 6 18 30 7 19 31 8 20 9 21 10 22 11 23 12 24	January February March April May June July August September October November December	Sunday Monday Tuesday Wednesday Thursday Friday Saturday

Note: Ctrl-click (or command-click on the Mac) to select and de-select minutes, hours, days and months.

**Date range to execute**

Run on any date

Only run from  /  /  ... to  /  /  ...

[Return to cron list](#)

Introduisez à présent la ligne de commande, programmez-la et puis cliquez sur le bouton **create**.

## Scheduled Cron Jobs

Find Cron jobs matching

Select all. | Invert selection. | Create a new scheduled cron job. | Create a new environment variable.

Active?	Command	Move
<input checked="" type="checkbox"/> Yes	wget -q http://patrick.all2all.org/phpcron.php 2>&1	

Select all. | Invert selection. | Create a new scheduled cron job. | Create a new environment variable.

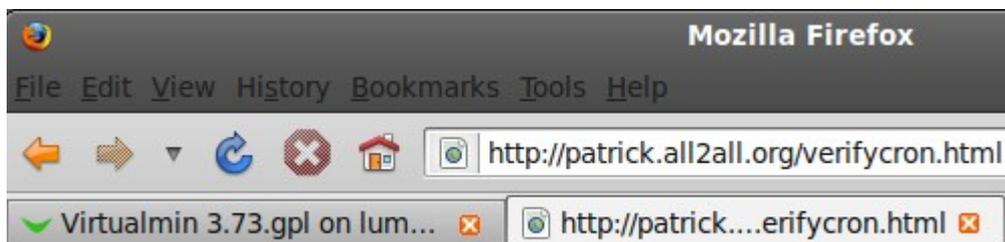
Vous pouvez maintenant voir la tâche programmée ainsi que les autres qui ont été créées. Vérifiez que vous avez bien coché “Yes” dans la colonne **active** et cliquez sur **Enable Selected Jobs**, le système inscrit à présent la tâche dans le cron.

Dans cet exemple, la commande **wget -q http://patrick.all2all.org/phpcron.php 2>&1** va être exécutée le mardi 27 octobre à 13 heures 47 minutes.

Syntaxe du crontab : **47 13 27 10 2 wget -q http://patrick.all2all.org/phpcron.php 2>&1**

Vous devriez donc voir maintenant qu'un nouveau fichier `verifycron.html` a été créé dans le dossier public à 13h47 !

En effet, lorsqu'on visite `http://patrick.all2all.org/verifycron.html`, la page créée par PHP montre la date et l'heure correctes, comme spécifié dans le cron :



the date is: 27-10-2009 13:47

Le Virtualmin filemanager confirme aussi que la date et l'heure son correctes :

Name	Size	User	Group	/ Date
..				
xmlrpc.php	352 B	patrick	patrick	Dec/05
logs	4 kB	patrick	patrick	30/Jul
joomla	4 kB	patrick	patrick	10/Sep
wordpress	4 kB	patrick	patrick	10/Sep
drupal-6.14	4 kB	patrick	patrick	18/Sep
upload	4 kB	patrick	patrick	15/Oct
index.html	490 B	patrick	patrick	26/Oct
stats	4 kB	patrick	patrick	26/Oct
phpcron.php	354 B	patrick	patrick	11:29
verifycron.html	29 B	www-data	www-data	13:47

Notez que vous pouvez tester vos lignes de commande avant de les programmer dans le cron. Pour ce faire, cliquez sur **Virtualmin**, puis sur **Webmin Modules** et finalement sur **Running Processes** :

Entrons par exemple une commande pour appeler de l'aide à propos du paquet PHP :

[Help..](#)

## Running Processes

**Display :** [PID](#) | [User](#) | [Memory](#) | [CPU](#) | [Search](#) | [Run..](#)

**Command to run**

**Run mode**  Run in background  Wait until complete

**Run as user**

**Input to command**

Le résultat de la commande est :

[Module Index](#)

## Command Output

Output from /usr/bin/php --help..

```
PHP Warning:  Cannot load module 'PDO ODBC' because required module 'pdo' is not loaded
Usage: php [options] [-f] <file> [--] [args...]
       php [options] -r <code> [--] [args...]
       php [options] [-B <begin_code>] -R <code> [-E <end_code>] [--] [args...]
       php [options] [-B <begin_code>] -F <file> [-E <end_code>] [--] [args...]
       php [options] -- [args...]
       php [options] -a

-a          Run interactively
-c <path>|<file> Look for php.ini file in this directory
-n          No php.ini file will be used
-d foo[=bar] Define INI entry foo with value 'bar'
-e          Generate extended information for debugger/profiler
-f <file>   Parse and execute <file>.
-h          This help
-i          PHP information
-l          Syntax check only (lint)
-m          Show compiled in modules
-r <code>   Run PHP <code> without using script tags <?..?>
-B <begin_code> Run PHP <begin_code> before processing input lines
-R <code>   Run PHP <code> for every input line
-F <file>   Parse and execute <file> for every input line
-E <end_code> Run PHP <end_code> after processing all input lines
-H          Hide any passed arguments from external tools.
-s          Display colour syntax highlighted source.
-v          Version number
-w          Display source with stripped comments and whitespace.
-z <file>   Load Zend extension <file>.

args...    Arguments passed to script. Use -- args when first argument
           starts with - or script is read from stdin

--ini      Show configuration file names

--rf <name> Show information about function <name>.
--rc <name> Show information about class <name>.
--re <name> Show information about extension <name>.
--ri <name> Show configuration for extension <name>.
```

## Lancer une tâche PHP dans cron

Les tâches programmées sont un trait commun des applications Web modernes. De nettoyer le cache toutes les 24 heures à vérifier les périodes d'abonnement et même produire des rapports d'activité, de plus en plus d'applications Web vivent avec une horloge intégrée.

- Vous pouvez appeler vos scripts PHP via cron en employant le paquet **PHP binary**. Disons que vos scripts se trouvent dans le répertoire `/var/www/htdocs/mysite/scripts` et votre paquet PHP se situe dans `/usr/bin/php`, la commande pour lancer votre script devrait être celle-ci :

**`/usr/bin/php /var/www/htdocs/mysite/scripts/runmyscript.php`**

Nous allons à nouveau utiliser le fichier `phpcron.php` avec la fonction de date pour montrer comment lancer une tâche PHP avec cron :

- Testez votre commande dans **Virtualmin=>Webmin Modules=>Running Processes** :

`/usr/bin/php /var/www/htdocs/patrick/phpcron.php`

[Help..](#)

### Running Processes

Display : [PID](#) | [User](#) | [Memory](#) | [CPU](#) | [Search](#) | [Run..](#)

<b>Command to run</b>	<input type="text" value="/usr/bin/php -a /var/www/htdocs/patrick/phpcron.php"/>	<input type="button" value="Run"/>
<b>Run mode</b>	<input type="radio"/> Run in background <input checked="" type="radio"/> Wait until complete	
<b>Run as user</b>	<input type="text" value="patrick"/> <input type="button" value="..."/>	
<b>Input to command</b>	<input type="text"/>	

[Module Index](#)

### Command Output

```
Output from /usr/bin/php -a /var/www/htdocs/patrick/
PHP Warning: Cannot load module 'PDO_ODBC' becaus
Interactive mode enabled
the date is: 26-10-2009 18:00
```

Le système indique la date correcte. Aussi pouvons-nous poursuivre en toute sécurité et configurer la tâche PHP dans le cron, sachant que la syntaxe PHP est correcte.

Créez ensuite la tâche dans cron pour exécuter votre commande comme indiqué [plus haut](#).

## Versions

Version number	Modifications	Author
1.0 EN	Original version	Patrick Brunswyck
1.0 FR	Traduction	Frédéric Jadoul