Planifier une tache avec Cron

Cron est un outil simple et par conséquent fantastique pour planifier des tâches sous Linux. Rien de moins. Tout administrateur système connait l'importance de pouvoir gérer des tâches régulières, automatiquement, en arrière-plan. L'utilitaire Linux Cron est un moyen efficace pour planifier des tâches à des moments spécifiques.

Pour commencer, voici la syntaxe de Cron :

```
m h dom mon dow user cmd
```

Champ	Description	Valeur acceptée
m	minutes	0-59
h	heure	0-23
dom	jour du mois	1-31
mon	mois	1-12
dow	jour de la semaine	0-6
user	utilisateur	n'importe quel utilisateur valide
command	commande	n'importe quelle commande valide

- **1 – Planifier une tâche à un moment spécifique**

L'utilisation basique de cron est d'exécuter une tâche à un moment spécifique comme dans cet exemple :

L'utilisateur root exécutera le script « full-backup » le 10 juin à 08:30 du matin

```
30 08 10 06 * root /root/scripts/full-backup
```

30 – 30ème minute
08 – 8 heures du matin (cron fonctionne en 24 heures)
10 – 10ème jour
06 – 6ème mois (juin)
* – n'importe quel jour de la semaine

- **2 – Planifier une tâche plus d'une fois (2 fois par jour par exemple)**

L'utilisateur backupuser exécutera le script incremental-backup 2 fois par jour, à heures fixes. Comme pour l'exemple précédent, mais des valeurs séparées par des virgules signalent plusieurs exécutions

```
00 11,16 * * * backupuser /home/backupuser/scripts/incremental-backup
```

00 – 0ème minute (début de l'heure)
11,16 – 11 et 16 heures
* – chaque jour
* – chaque mois
* – n'importe quel jour de la semaine

- **3 – Planifier une tâche dans une plage de temps spécifique**

Vous pouvez planifier une tâche chaque heure pendant une tranche horaire spécifique grâce au signe '-'

Chaque jour, pendant les heures de bureau
Cet exemple vérifie le statut d'une base de données chaque jour, même les weekends, pendant les heures de bureau (de 9 à 18h)

```
00 09-18 * * * sqluser /home/sqluser/bin/check-db-status
```

00 – 0ème minute (début de l'heure)
09-18 – 9, 10, 11, 12, 13, 14, 15, 16, 17, 18 heures
* – chaque jour
* – chaque mois
* – n'importe quel jour de la semaine

Chaque jour *de travail*, pendant les heures de bureau
Cet exemple vérifie le statut d'une base de données chaque jour sauf les weekends, pendant les heures de bureau (de 9 à 18h)

```
00 09-18 * * 1-5 sqluser /home/sqluser/bin/check-db-status
```

00 – 0ème minute (début de l'heure)
09-18 – 9, 10, 11, 12, 13, 14, 15, 16, 17, 18 heures
* – chaque jour
* – chaque mois
1-5 – lundi, mardi, mercredi, jeudi, vendredi

- **4 – Planifier une tâche chaque minute**

Normalement, vous ne devriez pas avoir de tâche planifiée toutes les minutes. Cependant, comprendre cet exemple vous permettra de mieux appréhender les exemples suivants.

```
* * * * user command
```

le signe * signifie 'tout ce qui est possible'. Donc ici, chaque minute, de chaque heure, de chaque jour de l'année.

D'autre signes sont possibles, particulièrement utiles :

/5 dans le champ minute signifie 'toutes les 5 minutes'
0-10/2 dans le champ minute signifie 'toutes les 2 minutes pendant les 10 premières minutes de l'heure'.

Cette sytaxe peut bien sûr être employées dans les 4 autres champs **h dom mon dow**

- **5- Planifier une tâche toutes les 10 minutes**

Vous pourriez faire la chose suivante si vous vouliez vérifier un espace disque toutes les 10 minutes.

```
*/10 * * * * user /home/user/check-disk-space
```

Vous pourriez cependant avoir besoin d'exécuter cette tâche seulement pendant les heures de bureau, par exemple. Les exemples au dessus vous montrent la marche à suivre. Ce serait :

```
*/10 09-18 * * 1-5 user /home/user/check-disk-space
```

***/10** – toutes les 10 minutes, toutes les heures
09-18 – 9, 10, 11, 12, 13, 14, 15, 16, 17, 18 heures
***** – chaque jour
***** – chaque mois
1-5 – lundi, mardi, mercredi, jeudi, vendredi

Il existe certains cas spéciaux où vous pouvez utiliser des mots-clés précédés du signe '@' en remplacement des 5 champs :

Mot clé	**Equivalent**
@yearly	0 0 1 1 *
@daily	0 0 * * *
@hourly	0 * * * *
@reboot	Exécuter au démarrage

- **6 – Planifier une tâche à la première minute de l'année en utilisant @yearly**

Si vous vouliez qu'une tâche s'exécute à la première minute de l'année, vous pourriez utiliser le mot-clé @yearly

```
@yearly user /home/user/annual-maintenance
```

- **7 – Planifier une tâche toutes les secondes**

C'est impossible : l'unité de temps minimale reconnue par cron est une minute.

- **8 – Spécifier une variable PATH dans crontab**

Jusqu'ici nous avons utilisé les chemins absolus des scripts.
Nous pourrions mettre ces chemins dans une variable PATH pour simplifier l'écriture des tâches :

```
PATH=/bin:/sbin:/usr/bin:/usr/sbin:/home/user

@yearly annual-maintenance
*/10 * * * * check-disk-space
```