Tutoriel : Cron (crontabs)

Dans ce tutoriel nous allons découvrir ce qu'est un cron et un crontab, à quoi ça sert et comment le mettre en place.

I. Qu'est-ce qu'un Cron ?

Cron est le diminutif de crontab qui est le diminutif de chrono table qui signifie table de planification.

Cron est un *daemon* utilisé pour planifier/programmer des tâches devant être exécutées à un moment précis. Chaque utilisateur a un fichier crontab, lui permettant d'indiquer les actions et à quelles périodes, elles devront être exécutées. Il y a également une crontab pour le système, permettant les tâches techniques, pour la mise à jour des différents programmes ou autres besoins périodiques. Par exemple, un cron peut permettre le lancement automatique d'un script bash.

II. Comment fonctionne un Cron ?

Pour utiliser cron, ajoutez simplement les entrées à votre fichier crontab (situé dans le répertoire /etc). Une entrée dans crontab contient deux parties et un retour à la ligne (« \n »). :

- La première partie de l'entrée de crontab décrit quand l'action sera effectuée. Il y a cinq champs, séparés par un espace ou une tabulation, et contenant un nombre, une étoile (*), ou le texte approprié. Les champs sont, dans l'ordre) :
  - *minute*,
  - *heure*,
  - *jour du mois*,
  - *mois*.
  - *jour de la semaine*

- La seconde partie indique la commande à lancer.

Si *jour du mois* et *mois* sont définis, *jour de la semaine* n'est pas nécessaire. Cependant, si le champ est indiqué en plus, la commande sera exécutée à la date jour du mois et mois mais AUSSI tous les jours de la semaine définis. Exemple avec :

```
0 0 13 1 5 tâche
```

La tâche sera exécutée le 13 janvier ET tous les vendredis.

—- L'exemple ci-dessous exécutera */usr/bin/apt-get update*, chaque jour, de chaque mois à 03:05 (le **cron** fonctionne sur 24 h) avec les droits de l'utilisateur **nomdutilisateur**.

```
5 3  * *  * nomdutilisateur /usr/bin/apt-get update
```

—- Vous pouvez faire tourner **cron** toutes les 5 minutes tout au long de la journée de travail (9am-5pm) avec un message :

```
*/5 9-17 * * mon,tue,wed,thu,fri wall "Où en es tu ?"
```

ou vous rappeler un anniversaire à 9h du matin le 10 janvier chaque année :

```
0 9 10 jan  * echo "C'est l'anniversaire de ta Maman aujourd'hui !" >>~/readme
```

Il existe des raccourcis intéressants :

- @reboot # se lance au reboot avec les droits utilisateurs, bien commode
- @yearly
- @annually
- @monthly
- @weekly
- @daily
- @midnight
- @hourly

Cf aussi :

```
man 5 crontab
```

Pour exécuter des applications graphiques, il faut tout d'abord être sûr que l'usager root a accès au *display* si jamais le contrôle d'accès est actif (cf. **xhost**), par exemple en exécutant (soi-même, ou en rajoutant la ligne dans un script de démarrage comme *rc.local*) :

```
xhost + local:root
```

Puis il faut préciser quel *display* utiliser lors de l'exécution de la commande à cron en ajoutant DISPLAY=*nom_du_display* au début de la commande à exécuter ; par exemple :

```
0 8 * * * DISPLAY=:0.0 totem "mon_fichier_son.mp3"
```

III. Commande pour Cron

Pour regarder le contenu de votre **crontab**, tapez :

```
crontab -l
```

Pour éditer le fichier de votre **crontab**, tapez :

```
crontab -e
```

Pour supprimer votre **crontab**, tapez :

```
crontab -r
```

Quand vous sortez de l'éditeur, le nouveau fichier **crontab** sera installé. Le fichier est stocké dans */var/spool/cron/crontabs/<user>* mais doit seulement être édité par l'intermédiaire de la commande **crontab**.

Note : sur xubuntu, il faut auparavant indiquer que l'utilisateur a le droit d'utiliser crontab. Pour cela il faut créer un fichier */etc/cron.allow* et y saisir le nom des utilisateurs autorisés à utiliser crontab.
L'éditeur utilisé pour modifier la crontab peut être modifié par la commande :

```
sudo update-alternatives --config editor
```

ou par :

```
select-editor
```