

Dernière mise-à-jour : 2020/08/06 15:21

# LDF205 - Gestion des Tâches

## Contenu du Module

- **LDF205 - Gestion des Tâches**

- Contenu du Module
- cron
  - Le Fichier /etc/crontab
  - Les Champs Temporels
  - Les crontabs Utilisateurs
  - Les Fichiers /etc/cron.allow et /etc/cron.deny
- anacron
  - Le Fichier /etc/anacrontab
  - Les Fichiers cron.daily, cron.weekly et cron.monthly
- at

## cron

Le service crond est normalement lancé au démarrage de la machine. Ce service est chargé de faire exécuter des tâches ( commandes et scripts ) à des moments précis.



Le service crond présume que la machine est allumée en permanence.

Le service crond lit toutes les minutes le fichier **/etc/crontab**, les crontabs se trouvant dans **/etc/cron.d** ainsi que les fichiers propres à chaque utilisateur.

Si une commande produit une sortie, celle-ci est dirigée vers la messagerie.

L'utilisation de `crond` est réservé à `root`. Cependant, vous pouvez établir une liste d'utilisateurs qui ont la permission d'utiliser `crond` en créant un fichier nommé **`cron.allow`** dans **`/etc`**. A l'inverse, un fichier **`cron.deny`** peut contenir une liste d'utilisateurs qui n'ont pas la permission d'utiliser `crond`.

Les crontabs des utilisateurs ont comme nom, le nom de l'utilisateur qui les crée et se trouvent dans le répertoire **`/var/spool/cron/crontabs/`**. Par exemple, le fichier propre à l'utilisateur `trainee` est le fichier **`/var/spool/cron/crontabs/trainee`**. Le service `crond` exécute des tâches en rajoutant une ligne dans son fichier de journalisation **`/var/log/syslog`**.

## Le Fichier `/etc/crontab`

Sous Debian **`/etc/crontab`** prend la forme suivante :

```
root@debian8:~# cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 * * * * root    cd / && run-parts --report /etc/cron.hourly
25 6 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
```



**Important** : Notez ici que le crontab appelle un binaire, nommé **`run-parts`**, afin d'exécuter les scripts et commandes se trouvant dans les répertoires `/etc/cron.hourly`, `/etc/cron.daily`, `/etc/cron.weekly` et `/etc/cron.monthly`. Dans ce fichier, on peut constater la définition de



variables dont SHELL et PATH. Ces variables sont différentes des variables système.

## Les Champs Temporels

Chaque ligne dans un fichier `/etc/crontab` contient 5 champs temporels qui décrivent le périodicité de l'exécution de la tâche concernée.

Les 5 champs sont :

Minutes	Heures	Jour du mois	Mois de l'année	Jour de la sem.
(0-59)	(0-23)	(1-31)	(1-12)	(0-6)*

\* le 0 correspond à dimanche.

Les champs temporels peuvent contenir des valeurs différentes :

Exemple	Description
Une valeur absolue telle 10	Dans le champs minutes = 10 minutes après l'heure
Une série de valeurs telle 2,6,8	Dans le champs mois = février, juin et août
Une intervalle telle 1-5	Dans le champs Jour de la Semaine = du lundi au vendredi
Le joker *	Dans le champs minutes = toutes les minutes
Une périodicité telle 0-23/2	Dans le champs heures = toutes les deux heures

## Les crontabs Utilisateurs

Chaque utilisateur peut avoir sa propre version d'un fichier crontab. Pour vérifier s'il existe une version de crontab pour lui, l'utilisateur lancera la commande **crontab -l**. Afin de créer son fichier crontab, l'utilisateur lance la commande crontab avec l'option **-e**. Cette commande lance l'interface de l'éditeur **vi**.

```
trainee@debian8:~$ crontab -l
no crontab for trainee
```

Afin de créer votre fichier crontab, lancez la commande crontab avec l'option -e en tant que votre nom d'utilisateur et non en tant que root :

```
$ crontab -e [Entrée]
```

Cette commande lance l'interface de l'éditeur **vi**. Tapez ensuite le texte suivant :

```
* * * * * /bin/pwd > pwd.txt
```

Sauvegardez et quittez vi.

Vous obtiendrez un résultat similaire à celui-ci :

```
trainee@debian8:~$ crontab -e
no crontab for trainee - using an empty one

Select an editor. To change later, run 'select-editor'.
 1. /bin/nano          <---- easiest
 2. /usr/bin/vim.tiny

Choose 1-2 [1]: 2
crontab: installing new crontab
```



**Important** : Ce fichier crontab va être exécuté toutes les minutes. La commande va inscrire le répertoire courant dans le fichier pwd.txt.

Le fichier crontab que vous venez de créer, au nom de votre utilisateur, a été écrit sur le disque dans le répertoire **/var/spool/cron/crontabs/**.

Malgré le fait que vous venez de l'éditer, vous ne pouvez pas visualiser votre fichier en tant que simple utilisateur avec la commande cat. Seule la commande crontab -e vous permettra de consulter ou modifier votre fichier crontab.

Passez donc en mode super-utilisateur et visualisez le fichier en utilisant la commande cat :

```
root@debian8:~# cat /var/spool/cron/crontabs/trainee
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (/tmp/crontab.raNAVf/crontab installed on Mon Oct 26 13:20:25 2015)
# (Cron version -- $Id: crontab.c,v 2.13 1994/01/17 03:20:37 vixie Exp $)
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
* * * * * /bin/pwd > pwd.txt
```

## Les Fichiers `/etc/cron.allow` et `/etc/cron.deny`

Afin d'accorder ou non le droit d'éditer son propre fichier crontab, root a la possibilité d'inscrire les noms d'utilisateurs dans un des deux fichiers :

- **`/etc/cron.allow`** pour accorder le droit d'éditer son propre fichier crontab

- **/etc/cron.deny** pour interdire le droit d'éditer son propre fichier crontab

Par contre, si root inscrit le nom d'un utilisateur dans /etc/cron.deny et cet utilisateur a déjà planifié des tâches avec son propre fichier crontab, les tâches seront exécutées. Pour détruire les tâches planifiées de l'utilisateur, il faut détruire le fichier crontab de l'utilisateur.

## anacron

L'inconvénient avec crond est qu'afin que celui-ci puisse exécuter des tâches automatiquement, la machine doit rester allumée en permanence.

Il existe un alternatif très efficace pour des machines qui ne sont pas constamment allumée. Cette solution est le logiciel Anacron.

Ouvrez le fichier /etc/anacrontab :

```
root@debian8:~# cat /etc/anacrontab
# /etc/anacrontab: configuration file for anacron

# See anacron(8) and anacrontab(5) for details.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
HOME=/root
LOGNAME=root

# These replace cron's entries
1 5 cron.daily run-parts --report /etc/cron.daily
7 10 cron.weekly run-parts --report /etc/cron.weekly
@monthly 15 cron.monthly run-parts --report /etc/cron.monthly
```

### Le Fichier /etc/anacrontab

Vous noterez que ce fichier fait référence aux répertoires cron.daily, cron.weekly et cron.monthly.

Les deux premiers champs sont des champs temporels. Dans le cas d'anacron ceux-ci correspondent à:

Période	Délai
La périodicité en jours	Le délai en minutes, après le démarrage d'anacron et avant l'exécution des tâches dans le répertoire concerné

## Les Fichiers cron.daily, cron.weekly et cron.monthly

Anacron mémorise dans les fichiers qui se trouvent dans **/var/spool/anacron/** la date à laquelle il a exécuté les commandes qui se trouvent dans anacrontab pour la dernière fois. Ces fichiers sont appelés cron.daily, cron.weekly et cron.monthly.

```
root@debian8:~# ls /var/spool/anacron/
cron.daily  cron.monthly  cron.weekly
root@debian8:~# cat /var/spool/anacron/cron.daily
20151026
root@debian8:~# cat /var/spool/anacron/cron.monthly
20151026
root@debian8:~# cat /var/spool/anacron/cron.weekly
20151023
```

Si cron exécute les tâches normalement, les dates d'exécution des tâches planifiées sont mises-à-jour à la date du jour dans les fichiers de contrôle par un script appelé **0anacron**. Dans le cas contraire anacron va prendre le relai lors du prochain démarrage.

Sous Debian, il existe trois fichiers **0anacron** :

```
root@debian8:~# updatedb
root@debian8:~# locate 0anacron
/etc/cron.daily/0anacron
/etc/cron.monthly/0anacron
/etc/cron.weekly/0anacron
```

A savoir :

```
root@debian8:~# cat /etc/cron.daily/0anacron
```

```
#!/bin/sh
#
# anacron's cron script
#
# This script updates anacron time stamps. It is called through run-parts
# either by anacron itself or by cron.
#
# The script is called "0anacron" to assure that it will be executed
# _before_ all other scripts.

test -x /usr/sbin/anacron || exit 0
anacron -u cron.daily

root@debian8:~# cat /etc/cron.weekly/0anacron
#!/bin/sh
#
# anacron's cron script
#
# This script updates anacron time stamps. It is called through run-parts
# either by anacron itself or by cron.
#
# The script is called "0anacron" to assure that it will be executed
# _before_ all other scripts.

test -x /usr/sbin/anacron || exit 0
anacron -u cron.weekly

root@debian8:~# cat /etc/cron.monthly/0anacron
#!/bin/sh
#
# anacron's cron script
#
# This script updates anacron time stamps. It is called through run-parts
# either by anacron itself or by cron.
```



```
#  
# The script is called "0anacron" to assure that it will be executed  
# _before_ all other scripts.  
  
test -x /usr/sbin/anacron || exit 0  
anacron -u cron.monthly
```

Le but de ces trois scripts est de mettre à jour la date inscrite dans les trois fichiers **/var/spool/anacron/cron.daily**, **/var/spool/anacron/cron.weekly** et **/var/spool/anacron/cron.monthly** respectivement.

## at

Tout comme avec la commande cron, root a la possibilité d'accorder ou d'interdire aux utilisateurs le droit d'exécuter des tâches avec at en utilisant les deux fichiers suivants :

- **/etc/at.allow**
- **/etc/at.deny**



Si le fichier **at.allow** existe, seuls les utilisateurs dans ce fichier pourront exécuter at.

En tant que root, mettez en place maintenant deux tâches pour le 31/12/2018 à 13h00 et 14h00 respectivement :

```
root@debian8:~# at 13:00 12/31/2018  
warning: commands will be executed using /bin/sh  
at> pwd > /tmp/test13.atd  
at> [^D]  
at> <EOT>  
job 1 at Sat Dec 31 13:00:00 2018  
  
root@debian8:~# at 14:00 12/31/2018
```

```
warning: commands will be executed using /bin/sh
at> free > /tmp/test14.atd
at> [^D]
at> <EOT>
job 2 at Sat Dec 31 14:00:00 2018
```

Sous Debian, les fichiers concernant ces deux tâches sont stockés dans le répertoire **/var/spool/cron/atjobs/** :

```
root@debian8:~# ls /var/spool/cron/atjobs
a0000101793190  a00002017931cc
```

A l'examen du deuxième fichier, vous constaterez un résultat similaire à celui-ci :

```
root@debian8:~# cat /var/spool/cron/atjobs/a0000101793190
#!/bin/sh
# atrun uid=0 gid=0
# mail trainee 0
umask 22
XDG_VTNR=7; export XDG_VTNR
XDG_SESSION_ID=2; export XDG_SESSION_ID
USER=root; export USER
MAIL=/var/mail/root; export MAIL
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin; export PATH
PWD=/root; export PWD
LANG=en_US.UTF-8; export LANG
SHLVL=1; export SHLVL
HOME=/root; export HOME
XDG_SEAT=seat0; export XDG_SEAT
LANGUAGE=en_US:en; export LANGUAGE
LOGNAME=root; export LOGNAME
XAUTHORITY=/home/trainee/.Xauthority; export XAUTHORITY
cd /root || {
    echo 'Execution directory inaccessible' >&2
    exit 1
}
```

```
}  
pwd > /tmp/test13.atd
```

Pour supprimer cette tâche il convient d'utiliser la commande suivante :

```
# at -d N° de tâche [Entrée]
```

Par exemple :

```
root@debian8:~# at -l  
2    Sat Dec 31 14:00:00 2018 a root  
1    Sat Dec 31 13:00:00 2018 a root  
root@debian8:~# at -d 2  
root@debian8:~# at -l  
1    Sat Dec 31 13:00:00 2018 a root
```

Finalement, pour exécuter plusieurs commandes à la même heure d'une manière séquentielle, vous pouvez les insérer dans un fichier :

```
# at 14:30 12/31/2018 < todo.list [Entrée]
```

Par exemple :

```
root@debian8:~# touch todo.list  
root@debian8:~# echo pwd > todo.list  
root@debian8:~# echo free >> todo.list  
root@debian8:~# echo who >> todo.list  
root@debian8:~# cat todo.list  
pwd  
free  
who  
root@debian8:~# at 14:30 12/31/2018 < todo.list  
warning: commands will be executed using /bin/sh  
job 3 at Thu Dec 31 14:30:00 2018
```

<html>

Copyright © 2020 Hugh Norris.<br><br>

</html>

From:

<https://www.ittraining.team/> - **www.ittraining.team**

Permanent link:

<https://www.ittraining.team/doku.php?id=elearning:workbooks:debian:6:junior:l110>

Last update: **2020/08/06 15:21**

