

Dernière mise-à-jour : 2020/01/30 03:28

LSF110 - Gestion des Tâches

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.

Les crontabs des utilisateurs ont comme nom, le nom de l'utilisateur qui les crée et se trouvent dans le répertoire **/var/spool/cron/tabs/**.

Par exemple, le fichier propre à l'utilisateur trainee est le fichier **/var/spool/cron/tabs/trainee**.

Le service crond exécute des tâches en rajoutant une ligne dans son fichier de journalisation **/var/log/messages**.

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.

Le Fichier /etc/crontab

Sous SLES **/etc/crontab** prend la forme suivante :

```
SLES12SP1:~ # cat /etc/crontab
SHELL=/bin/sh
PATH=/usr/bin:/usr/sbin:/sbin:/bin:/usr/lib/news/bin
MAILTO=root
```

```
#  
# check scripts in cron.hourly, cron.daily, cron.weekly, and cron.monthly  
#  
-*/15 * * * * root test -x /usr/lib/cron/run-crons && /usr/lib/cron/run-crons >/dev/null 2>&1
```



Important : Notez ici que le crontab appelle un script appelé **run-crons** afin d'exécuter les scripts et commandes se trouvant dans /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 la 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@SLES12SP1:~> 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@SLES12SP1:~> crontab -e  
no crontab for trainee - using an empty one  
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/tabs/**.

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` :

```
trainee@SLES12SP1:~> cat /var/spool/cron/tabs/trainee
cat: /var/spool/cron/tabs/trainee: Permission non accordée
trainee@SLES12SP1:~> su -
Mot de passe :
SLES12SP1:~ # cat /var/spool/cron/tabs/trainee
# DO NOT EDIT THIS FILE - edit the master and reinstall.
# (/tmp/crontab.5oXTvX installed on Fri Oct 13 15:57:34 2017)
# (Cronie version 4.2)
* * * * * /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 allumées. Cette solution est le logiciel `Anacron`.

Sous `SUSE`, `anacron` n'est pas installé par défaut. Il convient donc de l'installer à l'aide de la commande **`zypper`** à partir du dépôt d'`OpenSUSE` :

```
SLES12SP1:~ # zypper ar http://ftp.uni-erlangen.de/opensuse/distribution/openSUSE-current/repo/oss/ oss
Adding repository 'oss' .....[done]
Repository 'oss' successfully added
Enabled      : Yes
Autorefresh  : No
GPG Check    : Yes
URI          : http://ftp.uni-erlangen.de/opensuse/distribution/openSUSE-current/repo/oss/
```

```
SLES12SP1:~ # zypper install cronie-anacron
```

```
Refreshing service 'SUSE_Linux_Enterprise_Server_12_SP1_x86_64'.
```

```
New repository or package signing key received:
```

```
Repository:      oss
Key Name:        openSUSE Project Signing Key <opensuse@opensuse.org>
Key Fingerprint: 22C07BA5 34178CD0 2EFE22AA B88B2FD4 3DBDC284
Key Created:     Mon May  5 10:37:40 2014
Key Expires:     Thu May  2 10:37:40 2024
Rpm Name:        gpg-pubkey-3dbdc284-53674dd4
```

```
Do you want to reject the key, trust temporarily, or trust always? [r/t/a/? shows all options] (r): a
```

```
Building repository 'oss' cache .....[done]
```

```
Loading repository data...
```

```
Reading installed packages...
```

```
Resolving package dependencies...
```

```
The following NEW package is going to be installed:
```

```
  cronie-anacron
```

```
The following package is not supported by its vendor:
```

```
  cronie-anacron
```

```
1 new package to install.  
Overall download size: 30.3 KiB. Already cached: 0 B. After the operation,  
additional 39.2 KiB will be used.  
Continue? [y/n/? shows all options] (y): y
```

Le Fichier `/etc/anacrontab`

Ouvrez le fichier `/etc/anacrontab` :

```
SLES12SP1:~ # cat /etc/anacrontab  
# /etc/anacrontab: configuration file for anacron  
  
# See anacron(8) and anacrontab(5) for details.  
  
SHELL=/bin/sh  
PATH=/sbin:/bin:/usr/sbin:/usr/bin  
MAILTO=root  
# the maximal random delay added to the base delay of the jobs  
RANDOM_DELAY=45  
# the jobs will be started during the following hours only  
START_HOURS_RANGE=3-22  
  
#period in days   delay in minutes   job-identifier   command  
1    5    cron.daily      nice /usr/lib/cron/run-crons /etc/cron.daily  
7    25    cron.weekly    nice /usr/lib/cron/run-crons /etc/cron.weekly  
@monthly 45 cron.monthly    nice /usr/lib/cron/run-crons /etc/cron.monthly
```

Vous noterez que ce fichier fait référence aux répertoires `/etc/cron.daily`, `/etc/cron.weekly` et `/etc/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.

```
SLES12SP1:~ # ls /var/spool/anacron/  
cron.daily  cron.monthly  cron.weekly
```

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 relais lors du prochain démarrage :

```
SLES12SP1:~ # updatedb  
SLES12SP1:~ # locate 0anacron  
/etc/cron.hourly/0anacron  
SLES12SP1:~ # cat /etc/cron.hourly/0anacron  
#!/bin/sh  
# Check whether 0anacron was run today already  
if test -r /var/spool/anacron/cron.daily; then  
    day=`cat /var/spool/anacron/cron.daily`  
fi  
if [ `date +%Y%m%d` = "$day" ]; then  
    exit 0;  
fi  
  
# Do not run jobs when on battery power  
if test -x /usr/bin/on_ac_power; then  
    /usr/bin/on_ac_power >/dev/null 2>&1  
    if test $? -eq 1; then  
        exit 0  
    fi  
fi  
  
/usr/sbin/anacron -s
```

Ce script a pour but d'exécuter la commande `/usr/sbin/anacron -s`. L'option `-s` indique à anacron d'attendre la fin d'exécution d'un job avant d'exécuter le suivant.

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`**



Important - 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 31/12/2020 à 13h00 et 14h00 respectivement :

```
SLES12SP1:~ # at 13:00 12/31/2020
warning: commands will be executed using /bin/sh
at> pwd > /tmp/test13.atd
[^D]
at> <EOT>
job 1 at Thu Dec 31 13:00:00 2020
Can't open /run/atd.pid to signal atd. No atd running?
SLES12SP1:~ # systemctl status atd
atd.service - Deferred execution scheduler
   Loaded: loaded (/usr/lib/systemd/system/atd.service; disabled)
   Active: inactive (dead)

SLES12SP1:~ # systemctl enable atd
SLES12SP1:~ # systemctl start atd
SLES12SP1:~ # at 14:00 12/31/2020
```



```
warning: commands will be executed using /bin/sh
at> free > /tmp/test14.atd
[^D]
at> <EOT>
job 2 at Thu Dec 31 14:00:00 2020
```

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

```
SLES12SP1:~ # ls /var/spool/atjobs
.SEQ  a0000101994bb0  a0000201994bec
```

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

```
SLES12SP1:~ # cat /var/spool/atjobs/a0000201994bec
#!/bin/sh
# atrun uid=0 gid=0
# mail trainee 0
umask 22
LESSKEY=/etc/lesskey.bin; export LESSKEY
NNTPSERVER=news; export NNTPSERVER
MANPATH=/usr/share/man:/usr/local/man; export MANPATH
XDG_SESSION_ID=8; export XDG_SESSION_ID
HOSTNAME=SLES12SP1.fenestros.loc; export HOSTNAME
XKEYSYMDB=/usr/X11R6/lib/X11/XKeysymDB; export XKEYSYMDB
HOST=SLES12SP1.fenestros.loc; export HOST
PROFILEREAD=true; export PROFILEREAD
HISTSIZE=1000; export HISTSIZE
MORE=-sl; export MORE
USER=root; export USER
LS_COLORS=no=00:fi=00:di=01\;34:ln=00\;36:pi=40\;33:so=01\;35:do=01\;35:bd=40\;33\;01:cd=40\;33\;01:or=41\;33\;01
:ex=00\;32:\*.cmd=00\;32:\*.exe=01\;32:\*.com=01\;32:\*.bat=01\;32:\*.btm=01\;32:\*.dll=01\;32:\*.tar=00\;31:\*.t
bz=00\;31:\*.tgz=00\;31:\*.rpm=00\;31:\*.deb=00\;31:\*.arj=00\;31:\*.taz=00\;31:\*.lzh=00\;31:\*.lzma=00\;31:\*.z
ip=00\;31:\*.zoo=00\;31:\*.z=00\;31:\*.Z=00\;31:\*.gz=00\;31:\*.bz2=00\;31:\*.tb2=00\;31:\*.tz2=00\;31:\*.tbz2=00
\;31:\*.xz=00\;31:\*.avi=01\;35:\*.bmp=01\;35:\*.fli=01\;35:\*.gif=01\;35:\*.jpg=01\;35:\*.jpeg=01\;35:\*.mng=01\;
```

```
;35:\*.mov=01\;35:\*.mpg=01\;35:\*.pcx=01\;35:\*.pbm=01\;35:\*.pgm=01\;35:\*.png=01\;35:\*.ppm=01\;35:\*.tga=01\;  
35:\*.tif=01\;35:\*.xbm=01\;35:\*.xpm=01\;35:\*.dl=01\;35:\*.gl=01\;35:\*.wmv=01\;35:\*.aiff=00\;32:\*.au=00\;32:  
\*.mid=00\;32:\*.mp3=00\;32:\*.ogg=00\;32:\*.voc=00\;32:\*.wav=00\;32;; export LS_COLORS  
XNLSPATH=/usr/share/X11/nls; export XNLSPATH  
QEMU_AUDIO_DRV=pa; export QEMU_AUDIO_DRV  
HOSTTYPE=x86_64; export HOSTTYPE  
FROM_HEADER=; export FROM_HEADER  
PAGER=less; export PAGER  
CSHEDIT=emacs; export CSHEDIT  
XDG_CONFIG_DIRS=/etc/xdg; export XDG_CONFIG_DIRS  
MINICOM=-c\ on; export MINICOM  
MAIL=/var/spool/mail/root; export MAIL  
PATH=/sbin:/usr/sbin:/usr/local/sbin:/root/bin:/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games; export PATH  
CPU=x86_64; export CPU  
INPUTRC=/etc/inputrc; export INPUTRC  
PWD=/root; export PWD  
LANG=POSIX; export LANG  
PYTHONSTARTUP=/etc/pythonstart; export PYTHONSTARTUP  
GPG_TTY=/dev/pts/0; export GPG_TTY  
AUDIODRIVER=pulseaudio; export AUDIODRIVER  
QT_SYSTEM_DIR=/usr/share/desktop-data; export QT_SYSTEM_DIR  
SHLVL=1; export SHLVL  
HOME=/root; export HOME  
ALSA_CONFIG_PATH=/etc/alsa-pulse.conf; export ALSA_CONFIG_PATH  
SDL_AUDIODRIVER=pulse; export SDL_AUDIODRIVER  
LESS_ADVANCED_PREPROCESSOR=no; export LESS_ADVANCED_PREPROCESSOR  
OSTYPE=linux; export OSTYPE  
LS_OPTIONS=-A\ -N\ --color=tty\ -T\ 0; export LS_OPTIONS  
XCURSOR_THEME=DMZ; export XCURSOR_THEME  
WINDOWMANAGER=env\ GNOME_SHELL_SESSION_MODE=sle-classic\ gnome-session\ --session\ sle-classic; export  
WINDOWMANAGER  
LESS=-M\ -I\ -R; export LESS  
MACHTYPE=x86_64-suse-linux; export MACHTYPE  
LOGNAME=root; export LOGNAME
```

```
XDG_DATA_DIRS=/usr/share; export XDG_DATA_DIRS
LC_CTYPE=en_GB.UTF-8; export LC_CTYPE
LESSOPEN=lessopen.sh\ %s; export LESSOPEN
XDG_RUNTIME_DIR=/run/user/0; export XDG_RUNTIME_DIR
LESSCLOSE=lessclose.sh\ %s\ %s; export LESSCLOSE
G_BROKEN_FILENAMES=1; export G_BROKEN_FILENAMES
COLORTERM=1; export COLORTERM
cd /root || {
    echo 'Execution directory inaccessible' >&2
    exit 1
}
free > /tmp/test14.atd
```

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

```
# at -d N° de tâche [Entrée]
```

Par exemple :

```
SLES12SP1:~ # at -l
1 Thu Dec 31 13:00:00 2020 a root
2 Thu Dec 31 14:00:00 2020 a root
SLES12SP1:~ # at -d 2
SLES12SP1:~ # at -l
1 Thu Dec 31 13:00:00 2020 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 :

```
SLES12SP1:~ # touch todo.list
SLES12SP1:~ # echo pwd > todo.list
```

```
SLES12SP1:~ # echo free >> todo.list
SLES12SP1:~ # echo who >> todo.list
SLES12SP1:~ # cat todo.list
pwd
free
who
SLES12SP1:~ # at 14:30 12/31/2020 < todo.list
warning: commands will be executed using /bin/sh
job 3 at Thu Dec 31 14:30:00 2020
```

<html>

Copyright © 2004-2017 I2TCH LIMITED

</html>

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

Permanent link:
<https://ittraining.team/doku.php?id=elearning:workbooks:opensuse:11:junior:l110>

Last update: **2020/01/30 03:28**

