Version : **2021.01**

Updated : 2021/10/27 04:19

LCE505 - Process Scheduling

Contents

cron

The **crond** service, launched at boot time, is responsible for executing certain scripts and commands at specific intervals. The crond service assumes that the system is online permanently. In the case of a downtime coinciding with a cronjob, the cronjob is simply not executed until the following period.

Under Red Hat/CentOS, every 60 seconds crond reads the system crontab file, **/etc/crontab**, any crontabs in **/etc/cron.d/** and all the user crontabs.

User crontabs are files named after the user who created them and can be found in **/var/spool/cron/**.

The crond service writes to a log file - **/var/log/cron**.

If a command or script produces an output, that output is sent to root by mail.

The **root** user can establish lists of users that can or cannot create their own crontabs by editing either the **/etc/cron.allow** or **/etc/cron.deny** files.

The /etc/crontab file

A typical system crontab under Red Hat/CentOS looks as follows:

```
[root@centos8 ~]# cat /etc/crontab
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root

# For details see man 4 crontabs

# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name  command to be executed
```

Time Fields

A crontab file has five fields for specifying day, date and time:

Minutes	**Hours**	**Day of the month**	**Month**	**Day of the week**
(0-59)	(0-23)	(1-31)	(1-12)	(0|7-6)*

The following examples of values in those columns help explain how versatile a crontab can be:

Example	Description
An absolute value such as 10	In the *Minutes* column = 10 minutes after **each** hour
A series of values such as 2,6,8	In the *Month* column = February, June and August
A range such as 1-5	In the *Day of the week* column = From Monday through to Friday
The wildcard *	In the *Day of the month* column = Every day of the month
A regular interval such as 0-23/2	In the *Hour* column = Every two hours

Names of months or days of the week can be specified by name. Environment variables can be set in the crontab.

Special nicknames are supported :

- @reboot : Run once after reboot.
- @yearly : Run once a year, ie. “0 0 1 1 *”.
- @annually : Run once a year, ie. “0 0 1 1 *”.
- @monthly : Run once a month, ie. “0 0 1 * *”.
- @weekly : Run once a week, ie. “0 0 * * 0”.
- @daily : Run once * a day, ie. “0 0 * * *”.
- @hourly : Run once an hour, ie. “0 * * * *”.

For example, the **/etc/cron.d/0hourly** is as follows:

```
[root@centos8 ~]# cat /etc/cron.d/0hourly
# Run the hourly jobs
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
01 * * * * root run-parts /etc/cron.hourly
```

User Crontabs

As already stated, each authorized user can create a crontab. To check if a crontab exists, the user needs to execute the following command:

```
[root@centos8 ~]# exit
```

```
logout
[trainee@centos8 ~]$ crontab -l
no crontab for trainee
```

In order to create a crontab, the user should use the following command:

```
[trainee@centos8 ~]$ crontab -e
```

This command launches VI. Edit your crontab as follows:

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

Save and quit VI.

You will obtain a result similar to the following example:

```
[trainee@centos8 ~]$ crontab -e
no crontab for trainee - using an empty one
crontab: installing new crontab
```

The crontab that you have just created has been saved to disk in **/var/spool/cron/**.

You cannot view the contents of the file as a normal user. To do so you must become root:

```
[trainee@centos8 ~]$ su -
Password: fenestros
[root@centos8 ~]# cat /var/spool/cron/trainee
* * * * * /bin/pwd > pwd.txt
```

In order to allow or deny the right to a user to edit their crontab, root can edit either the **cron.allow** or **cron.deny** files.

- **/etc/cron.allow**,
- **/etc/cron.deny**.

In order to allow or deny the right to a user to edit their crontab, root can edit either the **cron.allow** or **cron.deny** files. However, if root puts a user in the cron.deny file, any existing cronjobs will continue unless they are manually deleted.

anacron

The major drawback with cron is that it assumes the machine is up and running at all times. For this reason, anacron is now responsable for executing the contents of the following directories :

- /etc/cron.daily
- /etc/cron.weekly
- /etc/cron.monthly

Anacron reads a list of jobs from its configuration file **/etc/anacrontab**:

```
[root@centos8 ~]# 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 run-parts /etc/cron.daily
7   25    cron.weekly        nice run-parts /etc/cron.weekly
@monthly 45 cron.monthly        nice run-parts /etc/cron.monthly
```

This file contains two important columns at the beginning of each line:

Interval	**Delay**
Period in days	Delay in minutes

Anacron records the date of execution in a special timestamp file for that job, so it can know when to execute it again. Only the date is used for the time calculations. The hour is not used :

```
[root@centos8 ~]# ls /var/spool/anacron/
cron.daily  cron.monthly  cron.weekly
[root@centos8 ~]# cat /var/spool/anacron/cron.daily
20210602
[root@centos8 ~]# cat /var/spool/anacron/cron.weekly
20210602
[root@centos8 ~]# cat /var/spool/anacron/cron.monthly
20210526
```

Anacron consults the **/var/spool/anacron/cron.daily** file every hour as specified in the **/etc/cron.hourly/0anacron** script. Anacron then decides whether to execute the **/usr/sbin/anacron -s** based upon the current date:

```
[root@centos8 ~]# 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
online=1
for psupply in AC ADP0 ; do
    sysfile="/sys/class/power_supply/$psupply/online"
    if [ -f $sysfile ] ; then
        if [ `cat $sysfile 2>/dev/null`x = 1x ]; then
            online=1
            break
        else
            online=0
        fi
    fi
done
if [ $online = 0 ]; then
    exit 0
fi
/usr/sbin/anacron -s
```

at

As in the case of cron, root has the ability to control who can and cannot use the at command by editing one of the following:

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

If the at.allow file exists, only users in that file can use the at command.

As root, create two jobs for the 31/12/2042 at 13h00 and 14h00 respectively:

```
[root@centos8 ~]# at 13:00 12/31/2042
warning: commands will be executed using /bin/sh
at> pwd > /tmp/test13.atd
[^D]
at> <EOT>
job 1 at Wed Dec 31 13:00:00 2042

[root@centos8 ~]# at 14:00 12/31/2042
```

```
warning: commands will be executed using /bin/sh
at> free > /tmp/test14.atd
[^D]
at> <EOT>
job 2 at Wed Dec 31 14:00:00 2042
```

The at files created can be found in **/var/spool/at/** :

```
[root@centos8 ~]# ls /var/spool/at
a000010249d9f8  a000020249da34  spool
```

Viewing the contents of the second file you will see something similar to the following example:

```
[root@centos8 ~]# at -c 2
#!/bin/sh
# atrun uid=0 gid=0
# mail trainee 0
umask 22
LS_COLORS=rs=0:di=38\;5\;33:ln=38\;5\;51:mh=00:pi=40\;38\;5\;11:so=38\;5\;13:do=38\;5\;5:bd=48\;5\;232\;38\;5\;11
:cd=48\;5\;232\;38\;5\;3:or=48\;5\;232\;38\;5\;9:mi=01\;05\;37\;41:su=48\;5\;196\;38\;5\;15:sg=48\;5\;11\;38\;5\;
16:ca=48\;5\;196\;38\;5\;226:tw=48\;5\;10\;38\;5\;16:ow=48\;5\;10\;38\;5\;21:st=48\;5\;21\;38\;5\;15:ex=38\;5\;40
:\*.tar=38\;5\;9:\*.tgz=38\;5\;9:\*.arc=38\;5\;9:\*.arj=38\;5\;9:\*.taz=38\;5\;9:\*.lha=38\;5\;9:\*.lz4=38\;5\;9:
\*.lzh=38\;5\;9:\*.lzma=38\;5\;9:\*.tlz=38\;5\;9:\*.txz=38\;5\;9:\*.tzo=38\;5\;9:\*.t7z=38\;5\;9:\*.zip=38\;5\;9:
\*.z=38\;5\;9:\*.dz=38\;5\;9:\*.gz=38\;5\;9:\*.lrz=38\;5\;9:\*.lz=38\;5\;9:\*.lzo=38\;5\;9:\*.xz=38\;5\;9:\*.zst=
38\;5\;9:\*.tzst=38\;5\;9:\*.bz2=38\;5\;9:\*.bz=38\;5\;9:\*.tbz=38\;5\;9:\*.tbz2=38\;5\;9:\*.tz=38\;5\;9:\*.deb=3
8\;5\;9:\*.rpm=38\;5\;9:\*.jar=38\;5\;9:\*.war=38\;5\;9:\*.ear=38\;5\;9:\*.sar=38\;5\;9:\*.rar=38\;5\;9:\*.alz=38
\;5\;9:\*.ace=38\;5\;9:\*.zoo=38\;5\;9:\*.cpio=38\;5\;9:\*.7z=38\;5\;9:\*.rz=38\;5\;9:\*.cab=38\;5\;9:\*.wim=38\;
5\;9:\*.swm=38\;5\;9:\*.dwm=38\;5\;9:\*.esd=38\;5\;9:\*.jpg=38\;5\;13:\*.jpeg=38\;5\;13:\*.mjpg=38\;5\;13:\*.mjpe
g=38\;5\;13:\*.gif=38\;5\;13:\*.bmp=38\;5\;13:\*.pbm=38\;5\;13:\*.pgm=38\;5\;13:\*.ppm=38\;5\;13:\*.tga=38\;5\;13
:\*.xbm=38\;5\;13:\*.xpm=38\;5\;13:\*.tif=38\;5\;13:\*.tiff=38\;5\;13:\*.png=38\;5\;13:\*.svg=38\;5\;13:\*.svgz=3
8\;5\;13:\*.mng=38\;5\;13:\*.pcx=38\;5\;13:\*.mov=38\;5\;13:\*.mpg=38\;5\;13:\*.mpeg=38\;5\;13:\*.m2v=38\;5\;13:\
*.mkv=38\;5\;13:\*.webm=38\;5\;13:\*.ogm=38\;5\;13:\*.mp4=38\;5\;13:\*.m4v=38\;5\;13:\*.mp4v=38\;5\;13:\*.vob=38\
;5\;13:\*.qt=38\;5\;13:\*.nuv=38\;5\;13:\*.wmv=38\;5\;13:\*.asf=38\;5\;13:\*.rm=38\;5\;13:\*.rmvb=38\;5\;13:\*.fl
c=38\;5\;13:\*.avi=38\;5\;13:\*.fli=38\;5\;13:\*.flv=38\;5\;13:\*.gl=38\;5\;13:\*.dl=38\;5\;13:\*.xcf=38\;5\;13:\
*.xwd=38\;5\;13:\*.yuv=38\;5\;13:\*.cgm=38\;5\;13:\*.emf=38\;5\;13:\*.ogv=38\;5\;13:\*.ogx=38\;5\;13:\*.aac=38\;5
\;45:\*.au=38\;5\;45:\*.flac=38\;5\;45:\*.m4a=38\;5\;45:\*.mid=38\;5\;45:\*.midi=38\;5\;45:\*.mka=38\;5\;45:\*.mp
3=38\;5\;45:\*.mpc=38\;5\;45:\*.ogg=38\;5\;45:\*.ra=38\;5\;45:\*.wav=38\;5\;45:\*.oga=38\;5\;45:\*.opus=38\;5\;45
:\*.spx=38\;5\;45:\*.xspf=38\;5\;45:; export LS_COLORS
LANG=en_US.UTF-8; export LANG
HISTCONTROL=ignoredups; export HISTCONTROL
GUESTFISH_RESTORE=\\e[0m; export GUESTFISH_RESTORE
HOSTNAME=centos8.ittraining.loc; export HOSTNAME
GUESTFISH_INIT=\\e[1\;34m; export GUESTFISH_INIT
USER=root; export USER
GUESTFISH_PS1=\\[\\e[1\;32m\\]\>\<fs\>\\[\\e[0\;31m\\]\ ; export GUESTFISH_PS1
PWD=/root; export PWD
HOME=/root; export HOME
MAIL=/var/spool/mail/root; export MAIL
SHELL=/bin/bash; export SHELL
SHLVL=1; export SHLVL
LOGNAME=root; export LOGNAME
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/root/bin; export PATH
GUESTFISH_OUTPUT=\\e[0m; export GUESTFISH_OUTPUT
HISTSIZE=1000; export HISTSIZE
LESSOPEN=\|\|/usr/bin/lesspipe.sh\ %s; export LESSOPEN
cd /root || {
     echo 'Execution directory inaccessible' >&2
     exit 1
}
${SHELL:-/bin/sh} << 'marcinDELIMITER4c5fdfe0'
free > /tmp/test14.atd
marcinDELIMITER4c5fdfe0
```

To delete that job you can use the at command:

```
[root@centos8 ~]# at -l
1	Wed Dec 31 13:00:00 2042 a root
2	Wed Dec 31 14:00:00 2042 a root
```

```
[root@centos8 ~]# atq
1	Wed Dec 31 13:00:00 2042 a root
2	Wed Dec 31 14:00:00 2042 a root
[root@centos8 ~]# at -d 2
[root@centos8 ~]# atq
1	Wed Dec 31 13:00:00 2042 a root
```

To execute several commands at the same time, it is simple to create a text file containing the commands and then to redirect the contents of the file to at's standard input:

```
[root@centos8 ~]# touch todo.list
[root@centos8 ~]# echo pwd > todo.list
[root@centos8 ~]# echo free >> todo.list
[root@centos8 ~]# echo who >> todo.list
[root@centos8 ~]# cat todo.list
pwd
free
who

[root@centos8 ~]# at 14:30 12/31/2042 < todo.list
warning: commands will be executed using /bin/sh
job 3 at Wed Dec 31 14:30:00 2042
```

Important - The **batch** command or it's alias **at -b** is used to execute commands when the system load is under a certain level. By default the value is set to 1.5. This value can be modified by the **-l** switch of the **atd** command.