# Planificació de tasques en GNU/Linux

## El dimoni Cron a Ubuntu 20.04

Vicent Benavent

CFGS ASIX
Mòdul: Administració de Sistemes Operatius
UD2: Planificació de tasques en GNU/Linux
Tot el temari el pots trobar a Github: **ASO**

# 1 Planificació de tasques en GNU/Linux

## 1.1 ¿Qué és cron?

El nom cron ve de el grec chronos que significa "temps". En el sistema operatiu Unix, cron és un administrador regular de processos en segon pla (dimoni) que executa processos o guions a intervals regulars (per exemple, cada minut, dia, setmana o mes). Els processos que s'han d'executar i l'hora en què s'ha de fer s'especifiquen en el fitxer crontab.

## 1.2 Com funciona

El dimoni cron s'inicia a /etc/rc.d/ o /etc/init.d depenent de la distribució que utilitzem. Cron s'executa en el background, revisa cada minut la taula de tasques crontab /etc/crontab o en /var/spool/cron buscant tasques que executar. Com a usuari podem afegir ordres o scripts amb tasques a cron per automatitzar alguns processos. Això és útil per exemple per automatitzar l'actualització d'un sistema o un bon sistema de còpies de seguretat.

## 1.3 ¿Qué és Crontab?

Crontab és un simple arxiu de text que guarda una llista d'ordres a executar en un temps especificat per l'usuari. Crontab verificarà la data i hora en què s'ha d'executar el script o l'ordre, els permisos d'execució i el realitzarà en el background. Cada usuari pot tenir el seu propi crontab, de fet a /etc/crontab s'assumeix que és el crontab de l'usuari root, quan la majoria de gent (i fins i tot root) volen generar el seu propi arxiu de crontab, en eixe cas utilitzarem el comandament crontab.

Crontab és la manera més senzilla d'administrar tasques amb cron en sistemes multiusuari, ja sigui com a simple usuari de sistema o usuari root.

## 1.4 Iniciar cron

Cron es un demonio (servicio), lo que significa que solo requiere ser iniciado una vez, generalmente con el mismo arranque del sistema. El servicio de cron antes se llama cron.service (antes crond). En la mayoría de las distribuciones el servicio se instala automáticamente y queda iniciado desde el arranque del sistema, se puede comprobar de varias maneras

```
$ /etc/init.d/cron status
$ systemctl status cron (o en distrubicions antigues: service cron status)
$ ps -ef | grep cron
$ systemctl | grep cron
```

Si per cap motiu cron no està funcionant:

```
# /etc/init.d/cron start o
```

Si el servei no estiguera configurat per arrancar a l'inici, el podem activar amb:

```
# systemctl enable cron
```

D'esta forma arrancarà a l'inici del sistema

O bé, d'una forma més brusca:

```
# chkconfig --level 35 cron on
```

Així estem agregant cron al nivell d'execució 3 i 5, de forma que iniciarà en el moment d'arranc del sistema.

## 1.5 Utilitzar crontab com a superusuari

Hi ha diverses formes d'utilitzar cron, la primera és en el directori /etc, on molt segurament trobaràs els següents directoris:

cron.hourly
cron.daily
cron.weekly
cron.monthly

Si es col·loca un arxiu tipus script en qualsevol d'aquests directoris, el script s'executarà cada hora, cada dia, cada setmana o cada mes, depenent del directori. Perquè l'arxiu puga ser executat ha de tindre una estructura per l'estil al següent:

```
#!/bin/sh
#script que genera una copia
cd /usr/documents
tar czf * copia
cp copia . /altre_directori/
```

Fixeu-se que la primera línia comença amb #!, que indica que es tracta d'un script shell de sh (podem utilitzar bash o la que vullgam), les altres línies són les ordres que desitgem executar. Aquest script podria nomenar-se per exemplecopia.sh i també hem de canviar-li els permisos corresponents perquè puga ser executat, per exemple:

```
#> chmod 700 copia.sh
# ls -l copia.sh
-rwx------ 1 root root 0 Jul 20 09:30 copia.sh
```

La "x" en el grup de permisos de propietari (rwx) indica que pot ser executat. Si aquest script el deixem en cron.hourly, s'executarà cada hora de cada dia.

Com a segon manera d'executar o utilitzar cron és a través de manipular directament l'arxiu /etc/crontab. En la instal·lació per defecte de diverses distribucions Linux, aquest arxiu es veurà a alguna cosa com el següent:

```
#> cat /etc/crontab
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
#MAILTO=root
#HOME=/
```

```
# run-parts
01 * * * * root run-parts /etc/cron.hourly
02 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
```

Les primeres quatre línies són variables que indiquen el següent:

**SHELL** és el interpret amb el que s'executarà el cron. Si no s'especifica, es s'agafarà per defecte l'indicat en la línia /etc/passwd corresponent a l'usuari que està executant cron.

**PATH** conté o correspon al camí al llistat en els quals cron buscarà la comanda a executar. Aquest path és diferent al path global de sistema o de l'usuari.

**MAIL TO** és a qui se li envia l'eixida de la ordre (si és que té alguna eixida). Cron enviarà un correu a qui s'especifique en esta variable, és a dir, ha de ser un usuari vàlid de sistema o d'algun altre sistema. Si no s'especifica, cron enviarà el correu a l'usuari propietari de l'ordre que s'executa.

**HOME** és el directori arrel o principal de la comanda cron, si no s'indica res, l'arrel serà la que s'indique a l'arxiu /etc/passwd corresponent a l'usuari que executa cron.

Els comentaris s'indiquen amb # a l'inici de la línia. Després de l'anterior vénen les línies que executen les tasques programades pròpiament. No hi ha límits de tasques que pot haver, una per línia.

```
vicent@vicent-ms7c84:~$ 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

# 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
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 )
#
```

**Linux és un sistema multiusuari** i cron és de les aplicacions que permet el treball amb diversos usuaris al mateix temps. Cada usuari pot tindre el seu propi crontab, de fet el /etc/crontab és el crontab de l'usuari root, encara que no hi ha problema que apareguen altres usuaris, i per això al sext camp indica precisament qui és l'usuari que executa la tasca i és obligatori a /etc/crontab. Però quan la majoria de gent (i fins i tot root) volen generar el seu propi arxiu de crontab, utilitzen l'ordre crontab.

Per començar amb un exemple simple. Anem a automatitzar l'actualització d'un sistema, per eliminar la història de "sempre dec estar actualitzant el sistema i això no m'agrada".

Primer que res farem un script. Aquest script serà cridat per cron i tindrà totes les instruccions que volem que faça, per tant cal provar-ho en diversos casos i de diverses formes abans d'afegir-lo al cron, un senzill script d'actualització com este:

```
#!/bin/bash
#script exemple d'actualització de repositoris i paquets
apt update & apt -y upgrade
```

Guardem el script com actualizacio.sh (ex. Directori scripts teu home). Canviem els permisos d'execució de l'script amb:

```
chmod u+x ~/scripts/actualizacio.sh
```

Executem el script un parell de vegades per verificar que tot s'execute sense problemes, modifiquem el necessari (no poden existir errors, si no cron només repetirà un error una i altra vegada). Ara a afegir el script al nostre crontab.

### 1.5.1 Agregar tareas a crontab

Editem el fitxer /etc/crontab. L'arxiu crontab tindrà una línia de inforamció pareguda, depenent de la versió poc canviar, fixeu-se a la captura de pantall de dalt.

```
# m h dom mon dow user command
```

On:

**m** correspon a el minut en què es va a executar el script, el valor va de 0a 59
**h** l'hora exacta, es maneja el format de 24 hores, els valors van de 0 A23, sent 0 les 12:00 de la mitjanit.
**dg** fa referència a el dia de el mes, per exemple es pot especificar 15 si es vol executar cada dia 15
**mon** Mes en que la comanda s'executarà, pot ser indicat numèricament (1-12), o pel nom del mes en anglès, nom és les tres primeres lletres.
**dow** significa el dia de la setmana (day of week), pot ser numèric (0 a 7, on 0 i 7 són diumenge) o les 3 primeres lletres del dia en anglès: mon, tue, wed, thu, fri, sat, sun.
**user** defineix l'usuari que va a executar la comanda, pot ser root, o un altre usuari diferent sempre que tingui permisos d'execució de l'script.
**command** refereix a la comanda o a la ruta absoluta de el script a executar, exemple: /home/usuari/scripts/actualizar.sh, si de cas crida a un script aquest ha de ser executable.

O el meu cas:

```
# 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
```

Perquè quede tot clar, uns quants exemples de tasques amb cron explicats:

```
15 10 * * * usuari /home/usuari/scripts/actualitzar.sh
```

Executarà el script actualizar.sh a les 10:15 a.m. cada dia

```
15 22 * * * usuari /home/usuari/scripts/actualitzar.sh
```

Executarà el script actualizar.sh a les 10:15 p.m. cada dia

```
00 10 * * 0 root apt-get -y update usuari root
```

Realitzarà una actualització tots els diumenges a les 10:00 a.m

```
45 10 * * sun root apt-get -y update
```

L'usuari root realitzarà una actualització tots els diumenges (sun) a les 10:45 a.m

```
30 7 20 11 * usuari /home/usuari/scripts/actualitzar.sh
```

El dia 20 de novembre a les 7:30 el usuari executarà el script

```
30 7 11 11 sun usuari /home/usuari/scripts/pastis.sh
```

El dia 11 de novembre a les 7:30 a.m. i que siga diumenge, el usuari felicitarà al seu sysadmin

```
*/1 * * * * usuari /home/usuari/scripts/molestrecordatori.sh
```

Un molest recordatori cada minut de cada hora cada dia (NO recomanable).

Es poden utilitzar rangs especials:

```
30 17 * * 1,2,3,4,5
```

En este cas s'executarà a les 5:30 de la vesprada tots els dies de dilluns a divendres.

```
00 12 1,15,28 * *
```

A les 12 del dia tots els dies primer, quinze i 28 de cada mes (especial per a nòmines)

Si això resulta confús, crontab maneja cadenes especials per a definir estos rangs.

- @reboot Executa una vegada, a l'inici
- @yearly executa només una vegada a l'any: 0 0 1 1 *
- @annually igual que @yearly
- @monthly executa un vegada al mes, el primer dia: 0 0 1 * *
- @weekly Setmanal del primer minut de la primera hora de la setmana. 0 0 * * 0 ''. @ Daily diari, a les 12:00 a.m. 0 0 * * *
- @midnight igual que @daily
- @hourly a el primer minut de cada hora: 0 * * * *

El seu ús és molt senzill:

```
@hourly usuari /home/usuari/scripts/molestrecordatori.sh
```

```
@monthly usuari /home/usuari/scripts/copia.sh
@daily root apt-get update && apt-get -y upgrade
```

## 1.6 Programar tasques amb un usuari estàndard

Tots els usuaris poden programar tasques. De fet, com he comentat abans, cada usuari té el seu propi fitxer crontab (/etc/crontab es considera el crontab de root). Per fer-ho n'hi ha prou amb usar la comanda:

```
$ crontab -e
```

Com que no hi ha cap fitxer creat per al meu usuari, em mostra este missatge, únicament cal seleccionar un editor de text

```
vicent@vicent-ms7c84:~$ crontab -e
no crontab for vicent - using an empty one

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

Una vegada creat, ja ens apareix el fitxer de configuració:

```
# 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
```

Fixeu-vos que a diferència de el fitxer /etc/crontab, ací no apareix el paràmetre 'user', ja que estem modificant el nostre crontab (el de l'usuari actual), és dona per entés que s'executarà com si forem nosaltres.

## 1.7 Administració de treball mitjançant cron

*crontab archivo*

Reemplaça l'existent crontab amb un arxiu definit pel usuari

*crontab -e*

Edita el crontab de l'usuari, cada línia nova serà una nova tasca de crontab.

*crontab -l*

Llista totes les tasques de crontab de l'usuari

*crontab -d*

Borra el crontab de l'usuari

*crontab -c dir*

Defineix el directori de crontab de l'usuari (aquest ha de tenir permisos d'escriptura i execució de l'usuari)

*crontab -u usuari*

Prefix per manejar el crontab d'un altre usuari, exemples:

```
$ sudo crontab -l -u root
$ sudo crontab -e usuari2
#crontab -d -u usuari
```

Com és lògic, cal tindre permisos de superusuari per modificar a un altre usuari.

Més exemples:

| Ejemplo | Descripción |
|---|---|
| 01 * * * * | Se ejecuta al minuto 1 de cada hora de todos los días |
| 15 8 * * * | A las 8:15 a.m. de cada día |
| 15 20 * * * | A las 8:15 p.m. de cada día |
| 00 5 * * 0 | A las 5 a.m. todos los domingos |
| * 5 * * Sun | Cada minuto de 5:00a.m. a 5:59a.m. todos los domingos |
| 45 19 1 * * | A las 7:45 p.m. del primero de cada mes |
| 01 * 20 7 * | Al minuto 1 de cada hora del 20 de julio |
| 10 1 * 12 1 | A la 1:10 a.m. todos los lunes de diciembre |
| 00 12 16 * Wen | Al mediodía de los días 16 de cada mes y que sea Miércoles |
| 30 9 20 7 4 | A las 9:30 a.m. del dia 20 de julio y que sea jueves |
| 30 9 20 7 * | A las 9:30 a.m. del dia 20 de julio sin importar el día de la semana |
| 20 * * * 6 | Al minuto 20 de cada hora de los sábados |
| 20 * * 1 6 | Al minuto 20 de cada hora de los sábados de enero |

| Ejemplo | Descripción |
|---|---|
| 59 11 * 1-3 1,2,3,4,5 | A las 11:59 a.m. de lunes a viernes, de enero a marzo |
| 45 * 10-25 * 6-7 | Al minuto 45 de todas las horas de los días 10 al 25 de todos los meses y que el día sea sábado o domingo |
| 10,30,50 * * * 1,3,5 | En el minuto 10, 30 y 50 de todas las horas de los días lunes, miércoles y viernes |
| */15 10-14 * * * | Cada quince minutos de las 10:00a.m. a las 2:00p.m. |
| * 12 1-10/2 2,8 * | Todos los minutos de las 12 del día, en los días 1,3,5,7 y 9 de febrero y agosto. (El incremento en el tercer campo es de 2 y comienza a partir del 1) |
| 0 */5 1-10,15,20-23 * 3 | Cada 5 horas de los días 1 al 10, el día 15 y del día 20 al 23 de cada mes y que el día sea miércoles |
| 3/3 2/4 2 2 2 | Cada 3 minutos empezando por el minuto 3 (3,6,9, etc.) de las horas 2,6,10, etc (cada 4 horas empezando en la hora 2) del día 2 de febrero y que sea martes |

I una xicoteta ajuda, existeixen diferents calculadores de cron a la xarxa, com per exemple:

- https://crontab.guru/
- https://crontab-generator.org/

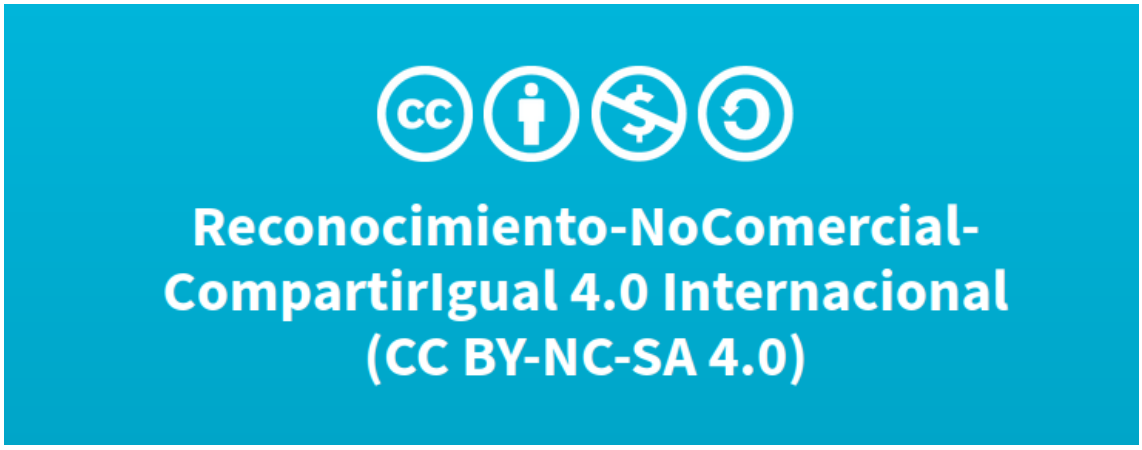
Entre altres…

# 2 Bibliografia

Temari original baix llicència Creative Commons Reconeixement-NoComercial- CompartirIgual 4.0 Internacional:

Javier Martínez (IES María Enríquez)

Vicent Benavent




Actualitzacions:
- Actualitzades les captures a l'última versió LTS d'Ubuntu (20.04) i afegits més exemples per una millor comprensió