cron / crontab

Conceptos básicos

Cron se ejecuta en el background, revisa cada minuto la tabla de tareas crontab **/etc/crontab** o en **/var/spool/cron**

comprobar si está en ejecución:

```
$ ps -ef | grep crond
```

Añadir el servicio si no lo estuviera:

```
$ chkconfig --level 35 crond on
```

variables de entorno

```
 $ export EDITOR=nano
```

Uso sencillo

añadir el script a ejecutar en alguno de estos directorios (o un link):

- /etc/cron.hourly → se ejecuta cada hora
- /etc/cron.daily → se ejecuta cada día
- /etc/cron.weekly → se ejecuta cada semana
- /etc/cron.monthly → se ejecuta cada mes

/etc/crontab

modificar `/etc/crontab` las lineas que realmente ejecutan los «trabajos» son del estilo:

```
* * * * * <usuario> <comando> [ > /dev/null 2>&1 ]
```

- indicar rangos: 1-3
- indicar listas: 1,3,5,7
- indicar fracciones:
  - */15 → cada 15 minutos (0,15,30,45)
  - 1-59/3 → cada 3 minutos, empezando por el minuto 1… 1,4,7,10… Diferente de */3 (0,3,6,9,12…)
  - 1-10/2 → los 1,3,5,7,9 (empieza en el 1, a saltos de 2)
  - 2/4 → 2,6,10,14 ???
  - indicando el rango de minutos, conseguimos desfasar de las ejecuciones de fracciones más simples
- En algunas distribuciones, se pueden indicar interrogantes → asumirá el minuto/hora/dia en el que fue iniciado el cron y a partir de ese momento siempre será ese.
- *L* : indica «último» (last)
  - en el caso del día de la semana, sería el último X del mes
  - en el caso del día del mes, sería el último día del mes

- *W* : día de la semana (weekday)
  - usado en el día del mes, ejecuta la tarea el día laboral más cercano.
  - ejemplo: 15W, si el 15 es sábado, lo ejecuta el viernes 14, si es domingo lo ejecuta el lunes 16
- *#* : usar en campo dia de la semana
  - permite ejecutar un determinado día de la semana de una semana concreta
  - ejempo: 5#3 ejecutaría el viernes de la tercera semana
- *H* : hashed
  - repite en algún momento indeterminado, pero invariable a partir de entonces
- https://en.wikipedia.org/wiki/Cron#Format

```
 ------------- minutos (0 - 59)
 | ----------- horas (0 - 23)
 | | --------- dia del mes (1 - 31)
 | | | ------- mes (1 - 12)
 | | | | ----- dia de la semana (0 - 6) (domingo=0, lunes=1, ... sabado=6)
 | | | | |
 * * * * * comando a ejecutar


* significa todos los valores validos
/ permite definir una repeticion
- permite definir un rango
, permite definir varios valores
```

otros formatos (special strings):

- @reboot → Run once, at startup.
- @yearly → Run once a year, «0 0 1 1 *».
- @annually → (same as @yearly)
- @monthly → Run once a month, «0 0 1 * *».
- @weekly → Run once a week, «0 0 * * 0».
- @daily → Run once a day, «0 0 * * *».
- @midnight → (same as @daily)
- @hourly → Run once an hour, «0 * * * *».

crontab (por usuario)

con los siguientes comandos podemos gestionar nuestros propios trabajos CRON (individual para cada usuario):

```
$ crontab <fichero> : añade el fichero con formato cron
$ crontab -e : edita el fichero
$ crontab -l : lista los trabajos añadidos a nuestros fichero
$ crontab -r : elimina los crontab
```

- El formato del crontab es ligeramente diferente, no se ha de especificar el usuario.
- permite una linea del tipo:
  - `MAILTO=«direccion@correoelectronico.es»`
  - `CRON_TZ=Europe/Andorra`
  - El fichero se guarda en `/var/spool/cron/crontabs` (en una debian, al menos)

ejemplos

- ejecuta el trabajo a las 00:00, 06:00, 12:00, 18:00

```
0 0,6,12,18 * * * /comando
```

- ejecuta el trabajo a las 00:00, 06:00, 12:00, 18:00 pero solo los días laborables:

```
0 0,6,12,18 * * 1-5 /comando
```

- ejecuta el trabajo a las 00:00, 06:00, de 9:00 a 15:00 cada hora, 18:00 pero solo los días laborables:

```
0 0,6,9-15,18 * * 1-5 /comando
```

- ejecuta el trabajo cada 6 horas (basicamente, cuando la división da 0 de resto) a la hora en punto (el 0 de los minutos):

```
0 */6 * * * /comando
```

- ejecuta el trabajo cada 5 minutos:

```
*/5 * * * * /comando
```

- ejecuta el trabajo todos los días pares a las 12:00:

```
0 12 */2 * * /comando
```

seguridad

se puede controlar quien usa y quien no el servicio CRON en los archivos:

```
/etc/cron.allow
/etc/cron.deny
```

en estos ficheros se añade la lista de usuarios con o sin permisos. Se puede utilizar ALL en cualquiera de los 2

Si no se crea el cron.deny y si el cron.allow, es como si hubiesemos creado un cron.deny con un ALL y tendremos que añadir a todos los usuarios que tengan que utilizar este servicio en el cron.allow

más info

- https://help.ubuntu.com/community/CronHowto
- https://www.freebsd.org/cgi/man.cgi?crontab%285%29
- https://blog.desdelinux.net/cron-crontab-explicados/

From:
https://matebcn.eu/ - **miguel angel torres egea**

Permanent link:
https://matebcn.eu/linux:cron:cron

Last update: **30/05/2026 13:44**