

Cron, at

# EN PUNTO

Las utilidades cron y at ayudan a automatizar procesos en un sistema Linux. Pueden configurarse para realizar copias de seguridad automáticas o incluso despertarte por la mañana con un repaso a tu colección de MP3. **POR HEIKE JURZIK**

**E**n Linux no hay necesidad de tomar notas para recordar tareas que necesitas llevar a cabo en el futuro. El programa *at* permite programar las tareas a realizar, mientras que *cron* controla los trabajos recurrentes.

Un demonio corre en el fondo para asegurarse de que las tareas se realizan según la programación establecida. El demonio comprueba los trabajos nuevos una vez por minuto. El que corresponde a *at* se denomina *atd*, mientras que el de cron es conocido como *crond*.

## At Tu Servicio

Puede invocarse a *at* desde la línea de comandos y esperar a que lleve a cabo su trabajo. A continuación se escriben más comandos en la shell y se sale presionando [Ctrl] + [D]:

```
$ at 07:00
warning: commands will be
executed using /bin/sh
at> mpg321 -z /home/huhn/mp3/*
at> <EOT>
job 6 at 2006-01-03 07:00
```

De esta manera se le está diciendo al reproductor de la línea de comando Mpg321 que nos despierte a las 7 en punto reproduciendo una selección de canciones del directorio */home/huhn/mp3/*. Para que esto funcione, lógicamente, el ordenador debe estar encendido a la hora designada.

La Tabla 1 ofrece los formatos más comunes para la hora. Se observa que *at* es persistente, es decir, que continuará corriendo después de reiniciar la máquina y recordará la programación durante los próximos seis meses.

Después de completar la tarea, esta utilidad envía un mensaje de correo a su propietario con el estado del trabajo, no importa si dicho trabajo se llevó a cabo con éxito o no. Para ello es necesario que la configuración obtenida del correo funcione correctamente (al menos para los envíos locales). Para comandos que no producen salidas por defecto, tales como *rm*, *mv* o *cp*, puede forzarse un mensaje de correo. Para hacerlo, se configura el parámetro *-m* en *at* de la siguiente manera:

```
at -m 13:31
```

## Mostrando y Borrando Tareas

Los comandos *at* programados se almacenan en la cola. Ésta puede verse en la pantalla si se invoca a *at -l* o *atq*:

```
$ atq
7      2006-01-03 07:00 a huhnix
11     2006-01-03 12:00 a huhnix
12     2006-06-09 14:24 a huhnix
```

Desafortunadamente, *at* no es muy comunicativo en este sentido, pero ofrece el número de tarea al comienzo de la línea, la fecha y la hora, el nombre de la cola (*a*) y el nombre del usuario. No dice nada de los trabajos que se encuentran programados. Además, sólo permite ver los trabajos propios como usuario normal. Como se habrá podido adivinar, el único que puede obtener una lista completa de los trabajos programados es el administrador del sistema.

Si se desean más detalles sobre lo que nos depara el futuro hay que cambiar a superusuario y modificar todo el

directorio de trabajo de *at* bajo */var/spool/atjobs/* (para Suse Linux), o */var/spool/cron/atjobs/* (para Debian). Los ficheros de texto dicen exactamente qué comandos correrán.

Para borrar una tarea, se escribe el comando *at -d* o *atrm*, especificándose el número del trabajo:

```
$ atrm 7 11
$ atq
12     2006-06-09 14:24 a huhnix
```

## Privilegios de Acceso

Existen dos ficheros que controlan quiénes pueden trabajar con *at*. Uno de ellos es */etc/at.allow*, y el otro */etc/at.deny*. La mayoría de las distribuciones tienden a disponer de un fichero *at.deny* con unas cuantas entradas “pseudo-usuario” para *lp* (el demonio de impresión) o *mail* (para el demonio de correo). Si se crea un fichero *at.allow* como superusuario, se necesitan entradas para todos los usuarios que tienen permitido correr trabajos *at*. En este caso, *at.deny* no se analiza.

## Un Cron para todas las Estaciones

Si se desean manejar tareas recurrentes de forma regular, no es recomendable correr *at* repetidamente. En su lugar, debería probarse con la otra opción que brinda Linux. *cron*. *Cron* también corre en el fondo trabajos a intervalos regulares. De nuevo, el programa necesita de la máquina encendida, y reprograma su “agenda” cuando se inicializa el ordenador. De hecho, los dos programas tienen más cosas en común, y así, *cron*, al igual que *at*, envía un correo a la cuenta propietaria para confirmar que un trabajo se ha completado con éxito.

Las tareas individuales son referidas como “trabajos cron” y se administran en la etiqueta cron. Existe una tabla con seis

### Listado 1: Entradas Crontab de Debian

```
17 * * * * root run-parts -- report /etc/cron.hourly
25 6 * * * root test -x /usr/sbin/anacron || run-parts -- report /etc/cron.daily
47 6 * * * root test -x /usr/sbin/anacron || run-parts -- report /etc/cron.weekly
52 6 * * * root test -x /usr/sbin/anacron || run-parts -- report /etc/cron-monthly
```

columnas que define el momento en el que ha de llevarse a cabo un trabajo específico. Cada comando ocupa en dicha etiqueta una línea única. Los primeros cinco campos definen la hora, mientras que el sexto contiene el programa que va a correr, incluídos cualesquiera parámetros.

Como usuario normal, se puede crear una etiqueta cron en la línea de comandos corriendo el programa crontab. El comando correspondiente para realizarlo es `crontab -e`, donde el parámetro `-e` indica que la tabla se está editando.

Adicionalmente, como administrador del sistema pueden modificarse los crontabs de cualquier usuario si se especifica el parámetro `-u` y se proporciona el nombre de la cuenta:

```
crontab -u huhn -e
```

Por defecto, esto llama al editor vi. Si se prefiere otro editor de texto diferente, se configura la variable de entorno `$EDITOR` para que quede reflejado, tal y como sigue:

```
export EDITOR=/usr/bin/kwrite
```

Para que este cambio permanezca siempre, se añade esta línea al fichero de configuración Bash y se reanaliza introduciendo `source ~/.bashrc`.

### Bien Estructurado

Las líneas de crontab no pueden contener órdenes que abarquen más de una línea. En el fichero, los comentarios se indican mediante una almohadilla (#) al comienzo de la línea. Los seis campos contienen la información siguiente, y en este orden:

- Minuto: Son posibles valores desde 0 a 59, así como el comodín \*
- Hora: Valores entre 0 y 23 ó \*
- Día: Valores desde 1 a 31 ó \*
- Mes: De 1 a 12, de *Jan* a *Dec*, *jan* a *dec* o \*
- Día de la semana: De 0 a 7 (donde tanto 0 como 7 significan Domingo), *Sun* a *Sat*, *sun* a *sat* o \*
- Comando: el comando a correr incluyendo sus opciones. Alternativamente puede ser el nombre de un script con más comandos.

Si se desea que el ordenador nos despierte a las siete cada mañana, se añade una entrada al crontab como la que sigue:

```
0 7 * * * mpg321 -z \
/home/huhn/mp3/*
```

En los campos individuales los valores pueden estar separados por una coma. Para que los sábados y domingos no nos despierte la alarma, se añade al quinto campo correspondiente al día de la semana lo siguiente:

```
0 7 * * 1,2,3,4,5 mpg321 -z \
/home/huhn/mp3/*
```

También es posible especificar un rango usando guiones (1-5). Pero si se usan nombres de los días de la semana es más fácil leer las entradas:

```
0 7 * * mon-fri ...
```

También puede ser útil una combinación de veces. Si el cuarto campo (para el mes) tiene valores de *1-4*, *7*, *10-12*, debe interpretarse “de Enero a Abril”, “Julio”, “Octubre a Diciembre” respectivamente. Una barra seguida de un número también define períodos de tiempo regulares; por ejemplo, *\*/2* en la segunda columna significa “cada dos horas” y *1-6/2* “1,3,5”.

Las tablas de cron para los usuarios se almacenan en el directorio `/var`. Cada distribución tiene una propuesta diferente para la organización de las tablas. Por ejemplo, Debian las almacena en `/var/spool/cron/crontabs/` y las clasifica por el nombre de usuario; en cambio, Suse Linux usa el directorio `/var/spool/cron/tabs/`.

Como no se dispone de permiso de lectura para este directorio como un usuario normal, puede presentarse una tabla de cron propia en la línea de comando corriendo el programa crontab:

```
$ crontab -l
10 8 * * mon-fri mpg321 -z \
/home/huhn/mp3/*
```

Para borrar entradas individuales, se arranca de nuevo el editor introduciendo `crontab -e`; si se desea borrar la tabla completa, puede correrse en su lugar `crontab -r`.

### Tablas de Cron Globales

Cron no maneja solamente listas de usuarios específicos, sino que también ayuda a los superusuarios en las tareas de administración del sistema. Trabajando como superusuario, se aconseja echar un

Tabla 1: Formatos de Hora para at

Formato	Significado
16:16	16:16 horas de hoy.
07:00pm	19:00 horas de hoy (si no se especifica <i>am</i> o <i>pm</i> , se supone que se trata de <i>am</i> ).
now	Ahora
tomorrow	Mañana
today	Hoy
now+10min	En diez minutos, también pueden especificarse <i>horas</i> , <i>días</i> , <i>semanas</i> y <i>meses</i> .
noon tomorrow	A las 12:00 del próximo día; también puede tratarse del <i>teatime</i> (=4:00 pm) o de medianoche.
6/9/06	9 de Junio del 2006, notaciones alternativas para la fecha son, por ejemplo, <i>6.9.06</i> y <i>6906</i> .

vistazo al fichero `/etc/crontab`, éste dirá los trabajos que maneja cron. Dependiendo de la distribución, el cron global puede variar; las entradas de Debian se muestran en el Listado 1, por ejemplo.

A diferencia de lo que ocurre con usuarios crontabs normales, el crontab global tiene un séptimo campo que contiene el nombre del usuario con los privilegios que correrá el comando (esto es algo muy típico de root). El Listado que se ve más abajo nos dice que el demonio cron corre las `run-parts--report /etc/cron.hourly` con privilegios de superusuario, una vez cada hora a los 17 minutos de cada hora, mientras que a las 6:52 am del primer día de cada mes corre `run-parts--report/etc/cron.monthly`.

Cron se ocupa de las tareas rutinarias diariamente a las 6:25 de la mañana (los scripts ejecutables en `/etc/cron.daily`), incluyendo al script `logrotate`, el cual rota, comprime y clasifica los ficheros log.

Si el ordenador que se está usando no corre veinticuatro horas al día, siete días

a la semana, se aconseja modificar esas entradas y especificar las horas en las que se sabe que la máquina estará activa:

```
25 17 * * * root ↵
test -x ↵
/usr/sbin/anacron || ↵
run-parts --report ↵
/etc/cron.daily
```

### Asistentes Cron Basados en GUI

Existen gran cantidad de herramientas basadas en GUI que ayudan a crear una tabla cron. Los usuarios de Gnome tienen Gcrontab, un programa muy bien organizado y de fácil uso que permite confeccionar una programación con unos cuantos clics de ratón. La herramienta KCron de KDE es incluso más fácil de usar, sin embargo, no presenta las entradas que crea una sintaxis cron, lo cual no ayuda a familiarizarse con cron (Figura

1). Pero, como suele ocurrir, la línea de comandos proporciona una mayor flexibilidad, y siempre será más rápido escribiendo entradas que haciendo clic en un montón de botones.

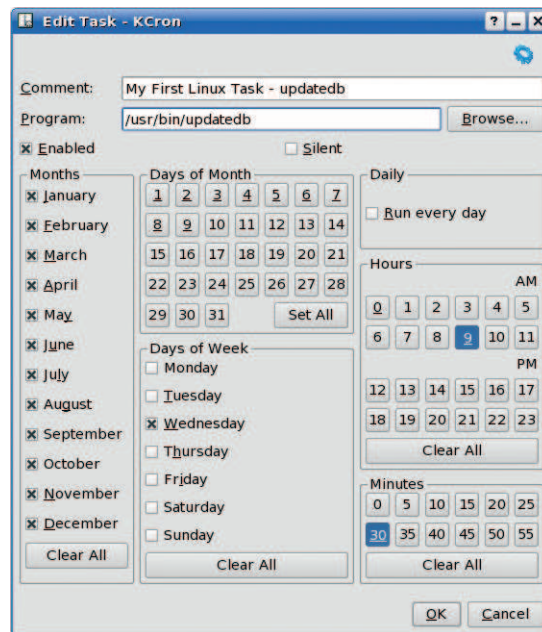


Figura 1: Kcron proporciona una GUI muy cómoda para administrar la configuración de cron.

# From end-user to power-user in 3 easy steps



Step 1: Relax in the sun.



Step 2: Meet people with similar interests.

Step 3: Get cool Linux training.

Find out more at [www.linux-magazine.es/training](http://www.linux-magazine.es/training)