PLANIFICANDO TAREAS CON CRON Y AT

Son muchas las posibilidades que nos ofrece un sistema Linux y entre ellas está la posibilidad de planificar y ejecutar tareas automáticamente, dejándonos tiempo libre para otras tareas de administración. Bastará editar un archivo o un simple comando y nuestro Linux se ocupará de todo.
Vamos a tratar aquí el comando at y el servicio crontab, un comando y un planificador de tareas.
Cron se "despierta" una vez por minuto, lee las entradas conocidas y ejecuta las acciones que se le han indicado, de lo cual deducimos que no es necesario reiniciar el servicio cron cada vez que se añade una nueva entrada, ya que al minuto siguiente será el mismo el que compruebe si ha habido cambios.
El funcionamiento de este servicio es sencillo, el demonio cron lee el archivo crontab, cuyo sistema principal está almacenado en /etc/crontab y debe modificarse a mano. Cada usuario debe tener su propia versión de este archivo, es decir, cada usuario tendrá su propio archivo y por tanto sus propias tareas. La sintaxis del programa crontab es la siguiente:

crontab [-u user] archivo
crontab [-u user] -l -e -r

Estos parámetros indican lo siguiente:

***La opción -u indica el usuario cuyo archivo crontab está apunto de utilizar. Si se omite la opción -u se asumirá que se está modificando el archivo crontab.**

***La opción -l le indica a crontab que enumere el archivo a la salida estándar.**

***La opción -e le indica a crontab que modifique el archivo, siendo el editor utilizado el definido en las variables EDITOR o VISUAL, si no existieran, se utilizaría vi.**

***La opcion -r elimina el archivo crontab especificado. Si no se especifica el archivo, se eliminará el archivo crontab de ese usuario.**

Vamos a ver un ejemplo de archivo /etc/crontab:

```
HELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# run-parts
01 * * * * root nice -n 19 run-parts /etc/cron.hourly
02 4 * * * root nice -n 19 run-parts /etc/cron.daily
22 4 * * 0 root nice -n 19 run-parts /etc/cron.weekly
42 4 1 * * root nice -n 19 run-parts /etc/cron.monthly
```

El primer campo es el shell donde se ejecutarán las órdenes. El segundo campo es el valor de la variable PATH, si las ordenes a ejecutar no se encuentran en dicho PATH habrá que modificarlo.
El tercero, manda los outputs del cron al root, o a quien queramos. Si queremos que

no se envie nada sólo debemos dejar la variable así: MAILTO="".
Los /etc/cron.hourly, daily...son ficheros que usa cron para su funcionamiento interno y los lanza cada día.

Para agregar una tarea a nuestros sistema lo primero que hemos de hacer es tener claro el concepto de campos de este archivo:

***El primer campo, los minutos.**
***El segundo campo, las horas.**
***El tercer campo, día.**
***El cuarto campo, día de la semana.**
***El quinto campo es la orden a ejecutar.**

Es decir, para el crontab el fichero es así:
minuto hora dia mes diadelasemana comando

Los minutos se especifican del 0 al 59.
Las horas del 0 al 23.
Los meses del 1 al 12.
Los días de la semana del 1 al 7.

Hay que tener en cuenta tambien el uso de caracteres especiales, como puedan ser el asterisco, el guión y la coma. El asterisco lo usaremos cuando queramos por ejemplo que una tarea se ejecute todos lo dias de la semana, colocando * en el campo correspondiente. El guión se deberá utilizar para indicar un rango de números enteros y la coma para señalar días o meses separados.
El símbolo de la barra (/) nos va a permitir afinar mucho mas el momento exacto de la ejecucion de la tarea, por ejemplo, si deseamos que una determinada tarea se ejecuta durante todos los meses excepto en junio, debemos indicarlo de la siguiente forma:

1-12/6

Pero también podemos indicar que solamente se haga en junio:

***/6**

Bien vamos a ver un ejemplo de como crear una tarea. Abrimos una consola de comandos y tecleamos lo siguiente:

[root@localhost vlad]# crontab -e

Se nos abrirá el editor vi y a continuación metemos nuestra linea:

37 11 4 4 */etc/init.d/cups stop

De esta forma el día 4 de Abril a las 11 y 37 minutos de la mañana, el servicio de impresión cups se parará. Podemos incluir todas las lineas que queramos, cron leerá el archivo y ejecutará la orden cuando le hayamos indicado.

30 17 * * 1,7 tar cfz /tmp/directorio_personal.tar.gz /home/root/

Con esta entrada a las 17:30 de todos los lunes y domingos se nos crearía en /tmp una copia de seguridad de nuestro directorio personal, en este caso root.
Bien, una vez que hemos puesto todas las ordenes, nos ponemos en modo comando del vi pulsando la tecla escape y al final de documento escribimos la orden de grabar y salir:

:wq

Por supuesto hemos de tener los permisos necesarios en el sistema en cuestion para que todas las ordenes que hemos dado a cron se lleven cabo. En ejemplo que he hemos puesto, si no somos root estas no podrán hacerse.

El siguiente comando a tratar ahora es at. Mediante at también vamos a poder establecer trabajos a realizar a una determinada hora, todo ello desde la linea de comandos y con unas sencillas instrucciones. Lo primero que hemos de comprobar es que tenemos instalado at y que el demonio atd está corriendo:

```
linux:/home/vlad # /etc/init.d/atd status
Checking for at daemon: running
```

Para ver el funcionamiento de at lo mejor es ver un ejemplo:

```
vlad@linux:~> at 22:08
warning: commands will be executed using /bin/sh
at>
```

Primeramente introducimos at seguido de la hora en la que dicha acción va a ser realizada y el shell se que da esperando que introduzcamos el comando a realizar:

```
at> rm /home/vlad/file
at>
```

Nuevamente at se queda a la espera de otra entrada, pero si no deseamos meter nada mas pulsaremos la combinación de teclas CONTROL+D para salir de at:

```
at> <EOT>
job 12 at 2004-11-11 22:08
```

At nos informa del número de trabajo asignado y de la hora a la que se llevará a cabo. Si quisiéramos ver las entradas de at que quedan pendientes por hacerse pondríamos esto en la consola:

```
vlad@linux:~> atq
12 2004-11-11 22:08 a vlad
```

Podemos borrar trabajos pendientes con el comando atrm:

```
vlad@linux:~> atq
12 2004-11-11 22:08 a vlad
vlad@linux:~> atrm 12
vlad@linux:~> atq
vlad@linux:~>
```

Por supuesto podemos ser mas específicos a la hora de definir el momento exacto de la ejecución de una tarea:

vlad@linux:~> at 10am Jul 31

vlad@linux:~> at 4pm + 3 days

vlad@linux:~> at 15:25 04/24/08

En este último ejemplo el comando que especificáramos a continuación se ejecutaría a las 15 y 25 minutos del 24 de Abril del 2006.