cron

CRON

El cron es un servicio usado para ejecutar órdenes a intervalos de tiempo. Las tareas se pueden programar por cada usuario o para el sistema.

El demonio cron se inicia o se detiene como cualquier otro servicio del sistema de la distribución correspondiente, por ejemplo:

```
/etc/rc.d/initd/crond start|stop|restart
```

aunque lo habitual es que se lance automáticamente al arrancar el sistema operativo.

Para comprobar si está funcionando

```
# /etc/rc.d/init.d/crond status
```

o si tienes el comando service instalado:

```
# service crond status
```

crond (pid 507) is running...

se puede también revisar a través del comando ps:

```
# ps -ef   |  grep crond
```

Si el servicio no estuviera configurado para arrancar desde un principio, bastaría con agregarlo con el comando chkconfig:

```
# chkconfig --level 35 crond on
```

Hay varias formas de usar cron

Por el Sistema

Para editar el del sistema, metemos las tareas en /etc/cron.d. Si se cambia estos ficheros hay que reiniciar el cron.

Cada tarea se añade indicando 5 campos que indican el período de ejecución, en las del sistema (/etc/cron.d), un campo más para el usuario con el que se ejecutará

Los 5 campos que indican el período son

- minutos
- horas
- día del mes
- mes
- día de la semana

Los posibles valores que puede tomar

- Un asterisco (*) indican todos los valores
- Una lista de números separados por comas
- Un rango, declarado como dos números separados por un guión
- Cada cierto tiempo, con una barra (valor/incremento)

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	l 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 a 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

Como se puede apreciar en el último ejemplo la tarea cron que estuviera asignada a ese renglón con esos datos, solo se ejecutaría si se cumple con los 5 campos (AND). Es decir, para que la tarea se ejecute tiene que ser un martes 2 de febrero a las 02:03. Siempre es un AND booleano que solo resulta verdadero si los 5 campos son ciertos en el minuto específico.

El caso anterior deja claro entonces que:

El programa cron se invoca cada minuto y ejecuta las tareas que sus campos se cumplan en ese preciso minuto.

Incluyendo el campo del usuario y el comando, los renglones de crontab podrían quedar entonces de la siguiente manera:

```
0 22 * * * root /usr/respaldodiario.sh
0 23 * * 5 root /usr/respaldosemanal.sh
0 8,20 * * * sergio mail -s "sistema funcionando" sgd@ejemplo.com
```

Las dos primeras líneas las ejecuta el usuario root y la primera ejecuta a las 10 de la noche de todos los días el script que genera un respaldo diario. La seguna ejecuta a las 11 de la noche de todos los viernes un script que genera un respaldo semana. La tercera línea la ejecuta el usuario sergio y se ejecutaría a las 8 de la mañana y 8 de la noche de todos los día y el comando es enviar un correo a la

cuenta sgd@ejemplo.com con el asunto “sistema funcionando”, una manera de que un administrador este enterado de que un sistema remoto esta activo en las horas indicadas, sino recibe un correo en esas horas, algo anda mal.

Siendo root, es posible entonces, modificar directamente crontab:

```
#> vi /etc/crontab
```

Por usuario

En el directorio /var/spool/cron (puede variar según la distribución), se genera un archivo cron para cada usuario, este archivo aunque es de texto, no debe editarse directamente.

Se tiene entonces, dos situaciones, generar directamente el archivo crontab con el comando:

```
$> crontab -e
```

Con lo cual se abrira el editor por default (generalemente vi) con el archivo llamado crontab vacio y donde el usuario ingresará su tabla de tareas y que se guardará automáticamente como /var/spool/cron/usuario.

El otro caso es que el usuario edite un archivo de texto normal con las entradas de las tareas y como ejemplo lo nombre 'mi_cron', después el comando $> crontab mi_cron se encargará de establecerlo como su archivo cron del usuario en /var/spool/cron/usuario

```
$> vi mi_cron
# borra archivos de carpeta compartida
0 20 * * * rm -f /home/sergio/compartidos/*
# ejecuta un script que realiza un respaldo de la carpeta documentos el
primer día de cada mes
0 22 1 * * /home/sergio/respaldomensual.sh
# cada 5 horas de lun a vie, se asegura que los permisos sean los correctos
en mi home
1 *5 * * * 1-5 chmod -R 640 /home/sergio/*
:wq (se guarda el archivo)
```

```
$> ls
```

mi_cron

```
$> crontab mi_cron
```

(se establece en /var/spool/cron/usuario)

Por usuario, usamos el comando crontab, el cual tiene las siguientes opciones:

- -e edita la tabla del cron
- -l lista las entradas
- -u usuario usa la tabla de otro usuario, en lugar del actual (sólo vale para root)
- -r borra la tabla entera

El formato de este fichero es el siguiente: minutos horas dia mes diadelasemana comando

- minutos→entre 0 y 59
- horas→Entre 0 y 23
- día→Entre 1 y 31
- mes→Entre 1 y 12
- día semana→Entre 0 y 6. 0 es Domingo, 1 Lunes, ... 6 Sábado
- El comando o comandos a ejecutar. Si no está en el PATH hay que especificar toda su ruta

Ejemplos:

Una vez cada hora	0 * * * *
Cada 20 minutos	0,20,40 * * * *
Otra cada 20 minutos	*/3 * * * *
Una vez al día, a las 05:00 am	05 00 * * *
Los domingos a las 12:00	00 12 * * 7
El primero de cada mes	00 20 1 * *
lanzar script todos los días a las 7	0 7 * * * /path/script.sh
lanzar script cada primer día del mes a las 7	0 7 1 * * /path/script.sh
lanzar script cada Viernes a las 17:30	30 17 * * 5 /path/script.sh

Si ponemos un * se ejecutarán una vez por hora en el caso de las horas, y una vez por minuto en el caso de los minutos

Hay que tener cuidado con los mensajes que envía el cron en caso de tener deshabilitado el sendmail. Para evitar que los mensajes se queden en /var/spool/clientmqueue hay que ejecutar

```
crontab -e
```

y añadir al final de la línea de cada trabajo una de las siguientes opciones:

- >/dev/null 2>&1
- &> /dev/null

Despues reinicial el servicio cron

```
/etc/init.d/crond restart
```

Otra opción es editar el fichero crontab

```
nano /etc/crontab
```

y modificar o añadir al principio la línea

```
MAILTO=""
```

Crontab predeterminados

La primera es en el directorio /etc, donde muy seguramente encontrarás los siguientes directorios:

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

Si se coloca un archivo tipo script en cualquiera de estos directorios, entonces el script se ejecutará cada hora, cada día, cada semana o cada mes, dependiendo del directorio.

Si se coloca un archivo tipo script en cualquiera de estos directorios, entonces el script se ejecutará cada hora, cada día, cada semana o cada mes, dependiendo del directorio.

Para que el archivo pueda ser ejecutado tiene que ser algo similar a lo siguiente:

```
    #!/bin/sh
#script que genera un respaldo
cd /usr/documentos
tar czf * respaldo
cp respaldo /otra_directorio/.
```

Nótese que la primera línea empieza con #!, que indica que se trata de un script shell de bash, las demás líneas son los comandos que deseamos ejecute el script. Este script podría nombrarse por ejemplo respaldo.sh y también debemos cambiarle los permisos correspondientes para que pueda ser ejecutado, por ejemplo:

```
#> chmod 700 respaldo.sh
#> ls -l respaldo.sh
```

-rwx—— 1 root root 0 Jul 20 09:30 respaldo.sh

La “x” en el grupo de permisos del propietario (rwx) indica que puede ser ejecutado.

Si este script lo dejamos en cron.hourly, entonces se ejecutará cada hora con un minuto de todos los días.

Otra forma es editar como root el fichero /etc/crontab

```
#> 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
```

SHELL es el 'shell' bajo el cual se ejecuta el cron. Si no se especifica, se tomará por defecto el indicado en la línea passwd dentro de etc, correspondiente al usuario que este ejecutando cron.

PATH contiene o indica la ruta a los directorios en los cuales cron buscará el comando a ejecutar. Este path es distinto al path global del sistema o del usuario.

MAIL TO es a quien se le envía la salida del comando (si es que este tiene alguna salida). Cron enviará un correo a quien se especifique en este variable, es decir, debe ser un usuario válido del sistema o de algún otro sistema. Si no se especifica, entonces cron enviará el correo al usuario propietario del comando que se ejecuta.

HOME es el directorio raíz o principal del comando cron, si no se indica entonces, la raíz será la que se indique en el archivo passwd correspondiente al usuario que ejecuta cron.

Los comentarios se indican con # al inicio de la línea.

Después de lo anterior vienen las líneas que ejecutan las tareas programadas propiamente. No hay límites de cuantas tareas pueda haber, una por renglón. Los campos (son 7) que forman estas líneas están formados de la siguiente manera:

```
Minuto  Hora  DiaDelMes  Mes  DiaDeLaSemana  Usuario  Comando
```

Campo	**Descripción**
Minuto	Controla el minuto de la hora en que el comando será ejecutado, este valor debe de estar entre 0 y 59.
Hora	Controla la hora en que el comando será ejecutado, se especifica en un formato de 24 horas, los valores deben estar entre 0 y 23, 0 es medianoche.
Día del Mes	Día del mes en que se quiere ejecutar el comando. Por ejemplo se indicaría 20, para ejecutar el comando el día 20 del mes.
Mes	Mes en que el comando se ejecutará, puede ser indicado numéricamente (1-12), o por el nombre del mes en inglés, solo las tres primeras letras.
Día de la semana	Día en la semana en que se ejecutará el comando, puede ser numérico (0-7) o por el nombre del día en inglés, solo las tres primeras letras. (0 y 7 = domingo)
Usuario	Usuario que ejecuta el comando.

Campo	Descripción
Comando	Comando, script o programa que se desea ejecutar. Este campo puede contener múltiples palabras y espacios.

El * es lo mismo que decir todo

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

También es posible especificar listas en los campos. Las listas pueden estar en la forma de 1,2,3,4 o en la forma de 1-4 que sería lo mismo. Cron, de igual manera soporta incrementos en las listas, que se indican de la siguiente manera:

```
Valor o lista/incremento
```

Controlando el acceso a cron

Cron permite controlar que usuarios pueden o no pueden usar los servicios de cron. Esto se logra de una manera muy sencilla a través de los siguientes archivos:

- /etc/cron.allow
- /etc/cron.deny

Para impedir que un usuario utilice cron o mejor dicho el comando crontab, basta con agregar su nombre de usuario al archivo /etc/cron.deny, para permitirle su uso entonces sería agregar su nombre de usuario en /etc/cron.allow, si por alguna razón se desea negar el uso de cron a todos los usuarios, entonces se puede escribir la palabra ALL al inicio de cron.deny y con eso bastaría.

#> echo ALL »/etc/cron.deny o para agregar un usuario mas a cron.allow #> echo juan »/etc/cron.allow

Si no existe el archivo cron.allow ni el archivo cron.deny, en teoría el uso de cron esta entonces sin restricciones de usuario. Si se añaden nombres de usuarios en cron.allow, sin crear un archivo cron.deny, tendrá el mismo efecto que haberlo creado con la palabra ALL. Esto quiere decir que una vez creado cron.allow con un solo usuario, siempre se tendrán que especificar los demás usuarios que se quiere usen cron, en este archivo.

Los ficheros de configuración de los distintos usuarios se almacenan en el directorio /var/spool/cron y un fichero de configuación del sistema llamado /etc/crontab. Cuando el demonio cron, llamado crond, está activo comprueba con minuto de frecuencia si hay alguna modificación de algún fichero de configuración o algún proceso que lanzar y en caso afirmativo lo ejecuta. Cron toma la salida del proceso, estándar y de errores, y se la envía al usuario correspondiente, a root en caso del crontab del sistema. Si queremos cambiar el usuario que recibe este mensaje de correo podemos utilizar la variable MAILTO.

Cada usuario puede tener y gestionar su propio fichero de configuración para cron. Bueno, hay dos ficheros, /etc/cron.allow y /etc/cron.deny donde se pueden poner restricciones. En caso de existir el fichero /etc/cron.allow, sólo los usuarios incluidos en este fichero podrán disponer de un crontab propio. Si no existe /etc/cron.allow pero sí existe un fichero /etc/cron.deny, cualquier usuario incluido en este último fichero no podrá disponer de fichero contab propio.

Los ficheros de configuración de crontab no están diseñados para editarse directamente por el usuario; aunque son ficheros de texto estos ficheros se modifican mediante la orden crontab. La orden crontab se puede utilizar como:

crontab [-u usuario] fichero crontab [-u usuario] { -l | -r | -e }

La opción -u se utiliza para indicar el usuario cuyo crontab queremos gestionar. Evidentemente sólo root podrá usar la orden crontab con esta opción. La ausencia de esta opción supone que es el usuario que ejecuta la orden el que gestiona su propio crontab.

La opción -l muestra el crontab activo en la salida estándar.

La opción -r elimina el crontab.

La opción -e se usa para crear y editar el crontab activo mediante el editor especificado en las variables de entorno EDITOR. El crontab modificado se instala automáticamente al salir del editor guardando los cambios.

GUIS para cron

Editores gráficos de cron, como son:

- GNOME Crontab Editor. Proyecto en fase alpha, pero ya genera archivos de cron funcionales. Lo encuentran en http://jodrell.net/projects/gnome-crontab.

Referencias

- http://dns.bdat.net/documentos/cron/x50.html
- http://www.linuca.org/body.phtml?nIdNoticia=256
- http://www.mexicoextremo.com.mx/content/view/19/62/
- http://www.linuxtotal.com.mx/index.php?cont=info_admon_006