Raúl Juncos-k idatzia
Ostirala, 2008(e)ko otsaila(r)en 08-(e)an 21:29etan

There are no translations available.
Descubre en este artículo las herramientas que permiten realizar, con cierta periodicidad, determinadas acciones o tareas...

# Automatización de tareas en sistemas GNU/Linux

## Introducción

En cualquier sistema informático se encuentran herramientas que permiten realizar, con cierta periodicidad, determinadas acciones o tareas (algunas de ellas internas del Sistema Operativo, otras definidas por el administrador, e incluso algunas definidas por un usuario -con los privilegios adecuados-). La necesidad de este tipo de herramientas viene dada, en principio, tanto por el funcionamiento interno del sistema operativo como por la necesidad del administrador de garantizar un aceptable funcionamiento del sistema.

Posiblemente, el caso más conocido de tarea programada sea la encomendada al sistema antivirus de los equipos. Por defecto, desde la instalación de la aplicación antivirus se suele proponer una revisión del sistema cada semana (por lo tanto, este es un ejemplo de una tarea programada de forma semanal).

A continuación se exponen algunos escenarios donde la automatización de tareas  simplifica y garantiza la ejecución de las mismas:

- Es típico que, para un óptimo rendimiento, el responsable del sistema informático deba realizar actividades de gestión periódicas, así como algunas de control del mismo (por ejemplo, revisar diariamente el grado de ocupación de los discos, testear cada 30 minutos la ocupación del procesador y poder generar un aviso en caso de ocupación de más de un 70% ...).
- Desde el ese mismo punto de responsable del sistema, puede resultar interesante definir una tarea que borre de forma periódica los ficheros temporales (por ejemplo, el directorio /tmp), así como que se realicen copias de seguridad cada día de diversos directorios de datos.
- También resulta útil (y en algunos casos obligado como parte de la propia política de empresa) realizar algún tipo de informe mensual sobre el funcionamiento del sistema (sobre todo en caso de servidores).
- Por supuesto, el apagado del equipo de forma automática, adecuándose al horario de trabajo, también puede formar parte de las tareas programadas.
- Además se pueden proponer otra serie de tareas periódicas desde las necesidades del

usuario como un sistema de recordatorios, o la descarga programada de información en horas de poco tráfico (para que sea más rápida).

En los sistema Debian GNU/Linux (al igual que el resto de distribuciones GNU/Linux) ya se encuentran "preconfiguradas" ciertas tareas de forma periódica; es decir, el servicio de automatización de tareas ya está funcionando cuando se realiza la instalación del sistema operativo.

En los entornos GNU/Linux, los programas típicos destinados a la gestión de las tareas automatizadas son dos, "at" y "cron".

# CRON

Se trata de unos de los servicios básicos de los sistemas GNU/Linux. De hecho, el demonio *cron*
siempre está arrancado; además, dicho servicio asume, asimismo, que el sistema siempre está en funcionamiento.

La función básica de *cron* es la de ejecutar tareas programadas para un determinado momento, y por un usuario con los privilegios necesarios para poder programarlas.

## Instalación de *cron* en el sistema

Aunque ya se ha mencionado que los sistema Debian GNU/Linux tienen instalado este servicio "de serie", es importante conocer cuál es el paquete que provee dicha funcionalida. El nombre del paquete es "cron", y se puede obtener una pequeña descripción desde la herramienta APT desde un terminal, ejecutando la orden:

$ apt-cache show cron

O bien, desde el programa gráfico Synaptic, buscando por nombre de paquete.

Por lo tanto, la instalación (o su reinstalación, si fuese necesario y con los correspondientes parámetros) se realizaría ejecutando la orden (con privilegios de *root*):

# apt-get install cron

## Ficheros implicados y configuración básica

Los ficheros más importantes implicados en el funcionamiento de servicio "cron" son:

- el propio demonio de funcionamiento: crond
- el fichero de configuración (disponible para *root*): /etc/crontab
- el fichero de inicio y parada del demonio: /etc/init.d/cron
- la orden para la programación de tareas (disponible para los usuarios con suficientes privilegios): crontab
- el sistema de informes (*logs*) típico de los sistemas GNU/Linux: /var/log/cron

Como se observa, la configuración del funcionamiento de *cron*, como ya es típico, se encuentra dentro del directorio /etc. Pero, poder arrancar o parar el demonio cron se deberían ejecutar las órdenes correspondientes:

- Parada del demonio cron: # /etc/init.d/cron stop
- Arranque del demonio cron: # /etc/init.d/cron start

Para entender cómo se deben programar las tareas para que sean ejecutadas, es necesario entender el formato del fichero de configuración (/etc/crontab). Este fichero está estructurado

por líneas, cada una de las cuales contiene una tarea programada, según el siguiente formato:

# minuto hora dia mes dia_semana usuario orden_a_ejecutar

Evidentemente, cada uno de estos campos tiene un rango de utilización, dependiendo de su naturaleza:

- El campo minuto puede ser definido entre [0-59],
- El campo hora puede ser definido entre [00-23],
- El día del mes, entre [1-31],
- El mes del año, entre los valores [1-12],
- El día de la semana, entre [0-7], asumiendo el 0 como inicio de la semana en domingo, y el valor 7 también como domingo (el 1 será el lunes, 2 martes...),
- El campo usuario es aquel usuario del sistema con permisos que ha definido la tarea programada,
- Por último, se indica la orden (o el script) que se ejecutará.

Además, existen símbolos especiales para los cinco primeros campos, que indicarían aspectos genéricos (no un número concreto):

* : indica cualquier valor

, : actúa como separador de una lista de valores

# : indica que lo que acompaña es un comentario (no se ejecutará)

- : sirve para indicar un rango de valores

/ : sirve para indicar un paso de valor (por ejemplo, en el campo mes si se indica */3 se está

detallando que la tarea se realizará cada tres meses).

Una vez vista su sintaxis, si se deseara ejecutar a las 10 y a las 17 horas, todos los días laborables, la orden `echo ¡Viva GNU y Linux!¡ `, se escribiría en el fichero de configuración la línea:

0 10,17 * * 1-5 echo "Viva GNU y Linux!" | wall

Como aclaración, se puede observar que el día de la semana se puede indicar en dos campos distintos. En caso que los dos forzasen un valor (es decir, que alguno de ellos o los dos no fuesen *), el sistema ejecutará el comando en cualquier de los dos casos (intentará que se cumplan los dos campos). Por ejemplo, en el siguiente ejemplo:

0,45 * 13 * 2 echo ¡ Hola martes o 13!¡ | wall

Esta orden se ejecutará cada 45 minutos, todos los martes, y además todos los 13 de cada mes.

Además, en los cinco primeros campos se puede optar por los siguientes cadenas:

@reboot: Se ejecuta al iniciarse la máquina.

@yearly: Se ejecuta una vez al año.

@monthly: Se ejecuta una vez al mes.

@weekly: Se ejecuta una vez por semana.

@daily: Se ejecuta una vez al día.

@hourly: Se ejecuta una vez por hora.

Y, por último, también es interesante conocer que cada una de las tareas programadas se ejecutan mediante un shell (/bin/sh), y que están disponibles algunas variables de entorno, como pueden ser LOGNAME, SHELL o NAME.

## Crontab

La orden crontab es la responsable de la planificación del servicio, y lo que hace es gestionar los ficheros crontabs asignados a cada usuario (en /var/spool/cron/crontabs/). Es decir, cada usuario (con los permisos adecuados) puede gestionar sus propias planificaciones de tareas.

Su sintaxis consiste en la orden crontab seguida del fichero, y de forma opcional, algunos parámetros:

crontab [-l e r u] fichero

Respecto a los parámetros, su significado es el siguiente:

-l : muestra el fichero de configuración del usuario

-e: edita el fichero de configuración del usuario

-r: borra el fichero de configuración del usuario

-u usuario: especifica el usuario propietario de la tarea (normalmente, esta opción la usa el usuario *root* para cambiar propietarios de tareas).

De este modo, un usuario con permisos (normalmente pasan por pertenecer al grupo de usuarios "crontab") podría generar un fichero con el formato adecuado para que se programasen sus tareas. Un ejemplo sencillo de cómo programar las tareas sería:

1.- Generar un fichero de nombre "ejemplo.cron" de una línea (recordar que se ejecutan las tareas por líneas mediante un shell) con el contenido:

0 9 1 09 * echo "Vuelta al cole :-D"|wall

2.- Ejecutar la carga del fichero de planificación:

$ crontab ejemplo.cron

3.- Para estar seguro que se ha añadido a la lista de tareas, mirar la lista de las mismas:

$ crontab -l

Donde debería devolver el contenido del fichero. En el ejemplo, se mostraría el mensaje (echo) todos los 1 de Septiembre, a las 9 de la mañana.

Una vez acabadas las pruebas, se puede eliminar la carga de tareas mediante crontab con la opción -r.

Para tener permiso y poder utilizar la orden, se deben tener presentes los siguientes ficheros:

/etc/cron.allow: si existe, sólo los usuarios listados en este fichero tienen permiso para ejecutar la utilidad (uno por línea).

/etc/cron.deny: si existe este fichero, y no el anterior, los usuarios detallados en él no pueden ejecutar *crontab*.

## Anacron

Hasta ahora, se ha mostrado cómo trabaja cron, pero se ha asumido que el sistema siempre está funcionando. Evidentemente, esto puede ser cierto para algunas máquinas (por ejemplo, ciertos servidores), pero no para todas (no todos los sistemas están en funcionamiento las 24 horas). Por ello, existe un programador de tareas (acompañando a cron) que no requiere del funcionamiento constante del sistema, anacron.

Este planificador se inicia junto con el sistema, y revisa cuáles son las tareas programadas que no se han llevado a cabo y las realiza. Habitualmente, suelen ser las tareas que se instalan en los directorios del tipo /etc/cron.* (daily, hourly...).

Cuando arranca anacron, revisa qué tareas programadas no se ha realizado y las lleva a cabo.

Como es predecible, la instalación de dicho servicio (aunque es estándar en los sistemas Debian GNU/Linux) se realizaría mediante la orden:

# apt-get install anacron

# AT

Como se ha comentado, el servicio ofertado por "cron" se ejecuta de forma periódica, de modo que las tareas programadas siempre serán realizadas siguiendo los criterios de temporalización programados.

A diferencia de "cron", las tareas que son encomendadas a "at" ( y su demonio, "atd") sólo se realizarán una vez. Es decir, la utilidad "at" se utiliza para programar una tarea que se llevará a cabo en un momento determinado, y no se volverá a ejecutar.

Las utilidades que se pueden encontrar directamente relacionadas con "at" son:

- at: orden que se utiliza para añadir nuevas tareas,
- atd: es el demonio responsable de ejecutar las tareas programadas desde "at"
- atq: muestra la lista de tareas pendientes a ejecutar (por el usuario que llama al comando)
- atrm: elimina una tarea de la lista de pendientes.

## Instalación de *at* en el sistema

Como ya podría predecirse, y siguiendo la pauta anterior, la obtención de "at" en el sistema se realiza mediante la instalación del paquete "at", con la orden:

# apt-get install at

## Ficheros implicados y utilización

Al igual que en el caso de "cron", los ficheros relacionados con "at" son:

- /etc/at.allow: si existe, sólo los usuarios listados en este fichero tienen permiso para ejecutar la utilidad.
- /etc/at.deny: si existe este fichero, y no el anterior, los usuarios detallados en él no pueden ejecutar at (ni atrm, ni atq).
- Si no existiesen ninguno de los anteriores ficheros, sólo el usuario *root* puede disponer de la utilidad "at".

Normalmente, después de la instalación se dispone del fichero /etc/at.deny, aunque vacío, de modo que todos los usuarios tienen acceso a la orden "at".

Para conocer el funcionamiento de "at", como ya es típico en los sistema GNU/Linux, se puede acudir a la ayuda (man) de la orden ejecutando:

$man at

No obstante, su utilización es bastante simple. La sintaxis que se utiliza es la orden (at) seguido de los parámetros:

- HH[:]MM[am|pm] [Mes día]
- Se puede añadir a la fecha/hora un número (seguido de "minutes","hours","days","weeks")
- También se pueden añadir valores relativos como "now", "midnight", "noon", "teatime", "today" o "tomorrow".

A continuación se proponen algunos ejemplos de utilización:

$ at 10am tomorrow

$ at 10am Jun 30

$ at 1730 Feb 28 + 3 days

Para que ejecute en el momento programado -las 12 AM del día siguiente- un script preparado por el usuario, se debe indicar del siguiente modo:

$at 12am tomorrow < copia.sh

El cometido del script puede ser, por ejemplo, una copia de la configuración del sistema gráfico (Xorg) de la máquina, con el siguiente contenido:

#!/bin/sh

#Copia del fichero /etc/X11/xorg.conf al escritorio del usuario

cp -f /etc/X11/xorg.conf $HOME/Desktop

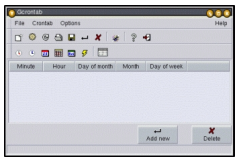
exit 0

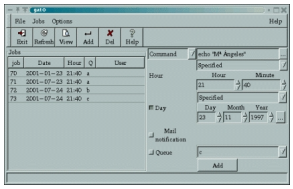
## Utilidades gráficas

Lo descrito anteriormente debe ser conocido por el responsable de la automatización de las tareas. No obstante, existen algunas utilidades en entorno gráfico que facilitan la utilización y gestión de las tareas programadas en el sistema.

Posiblemente la más utilizada (por ser independiente del escritorio -KDE,GNOME- o gestor de ventanas instalado) de forma histórica en sistemas servidores sea ***Webmin*** [ http://www.webmin.com/ ], mediante el módulo de sistema dedicado a la automatización. Una vez instalado webmin, así como los comandos "at" y "cron", se puede acceder a la utilidad gráfica de Webmin desde un navegador web, dirigiéndose a la dirección " http://localhost:10000 " (o https, dependiendo del tipo de instalación), y dirigiéndose a la entrada Webmin->Sistema->Comandos planificados (Scheduled Commands, Scheduled Cron Jobs).

Desde el entorno GNOME, de forma histórica, se encuentran las utilidades "**Gcrontab**" y "**Gato**". Su instalación se realiza de forma análoga a la descrita, pero con los paquetes "gcrontab" y "gato", respectivamente. A continuación se muestran algunas capturas de pantalla:





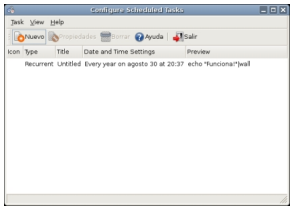
Como se observa, estas utilidades gráficas dependen de la versión GTK 1.2; por ello, actualmente desde el proyecto GNOME se oferta una única utilidad de planificación, **GNOME Schedule** , para poder gestionar estas tareas.

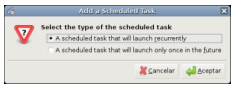
La instalación se realiza mediante la orden (o desde el programa Synaptic):

```
# apt-get install gnome-schedule
```

A partir de ese momento se puede lanzar el programa desde la entrada de menú  Aplicaciones-Herramientas del sistema-Planificación .

A continuación se muestran algunas capturas. El funcionamiento, después de la explicación dada, resulta trivial.




**Algunos sitios de interés:**

- Ayudas de los comandos (man) referidos en el artículo.
- Proyecto gnome-chedule [ http://gnome-schedule.sourceforge.net/ ]
- Proyecto Gato [ http://www.arquired.es/users/aldelgado/proy/gato/ ]
- Proyecto Webmin [ http://www.webmin.com ]
- Proyecto Gcrontab [ http://www.arquired.es/users/aldelgado/proy/gcrontab/ ]
- Proyecto LuCas [ http://es.tldp.org/Manuales-LuCAS/doc-curso-salamanca-admin-avanzada/html/ch11s02.html ]
- Wikipedia, entrada at (Unix) [ http://en.wikipedia.org/wiki/At_%28Unix%29 ]