Ingeniería de Software: agregado de archivos cron y ejecución periódica de procesos en los paquetes .deb

Esta serie de papers que comenzó en la edición nro. 11 de la revista "Hackers & Developers Magazine" explicando cómo crear paquetes Debian y continuó en la edición anterior de The Original Hacker hablando acerca de cómo desarrollar archivos pre y post instalación propios, comienza a llegar a su fin en esta entrega, en la que me concentraré en la necesidad de nuestras aplicaciones de incluir la ejecución periódica de procesos que para correr, requieran almacenarse en los directorios /etc/cron.*

Cuando desarrollamos aplicaciones, frecuentemente incluimos instrucciones, sugerencias y hasta incluso directivas de mantenimiento en archivos README o en la documentación de nuestro Software. Sin embargo, incluso aunque el público objetivo de nuestros desarrollos sean profesionales de los sectores IT, toda instrucción, directiva o sugerencia que requiera ser ejecutada post-instalación, suele ser ignorada, sobre todo, si la misma necesita ser llevada a cabo en más de una oportunidad.

Tanto Debian como sus *distros* derivadas, son capaces de ejecutar tareas específicas de forma periódica, que van **más allá de** la programación de tareas mediante **`crontab`**.

Dichas tareas son ejecutadas por el propio Sistema Operativo, cada hora, diaria, semanal o mensualmente y las mimas, se encuentran especificadas en **guiones de *shell* ejecutables**, alojados en los directorios `/etc/cron.hourly`, `/etc/cron.daily`, `/etc/cron.weekly` y `/etc/cron.monthly`.

De allí que, por ejemplo, tareas de limpieza, depuración y mantenimiento de la base de datos de una aplicación *(como la que vimos en el artículo de emulación de tokens temporales[1] en la edición pasada)*, tranquilamente podrían efectuarse mediante un `cron.daily` - agregado al paquete `.deb` -, liberando así al usuario de tener que programar dicha tarea manualmente.

1 Artículo **"Emulación de tokens de seguridad temporales para el registro de usuarios"** (The Original Hacker Nº 1): http://library.originalhacker.org/biblioteca/articulo/ver/115

Los archivos cron.* y su formato

Los archivos `cron.*`, como comenté unos párrafos más arriba, son guiones de *shell* ejecutables. Estos archivos se alojan en las diversas carpetas `/etc/cron.*` bajo el nombre de la aplicación.

Es así, que por ejemplo, en la carpeta `/etc/cron.daily` encontraremos archivos cuyos nombres serán `apache2`, `dpkg`, `sendmail` o `apt`, entre otros.

Cada carpeta `cron.*` está destinada a almacenar guiones que se ejecutarán bajo un mismo período de tiempo. Muy intuitivamente, la relación carpeta-período es la siguiente:

```
/etc/cron.hourly        cada hora
/etc/cron.daily         diariamente
/etc/cron.weekly        semanalmente
/etc/cron.monthly       mensualmente
```

Haciendo un `cat` a cualquiera de los archivos alojados dentro de alguna de estas carpetas `cron.*`, podremos comprobar que efectivamente se trata de guiones de *shell*:

```
eugenia@cococha-gnucita:/etc/cron.daily$ cat passwd
#!/bin/sh

cd /var/backups || exit 0

for FILE in passwd group shadow gshadow; do
        test -f /etc/$FILE              || continue
        cmp -s $FILE.bak /etc/$FILE     && continue
        cp -p /etc/$FILE $FILE.bak && chmod 600 $FILE.bak
done
eugenia@cococha-gnucita:/etc/cron.daily$
```

> *Cualquier guión de shell es válido para ser agregado como archivo cron.**

Excepciones cron.d

La única **excepción** a todo lo anterior, son los archivos cuyo destino será el directorio **`/etc/cron.d.`** Los archivos alojados en este directorio, tendrán un formato diferente y además, solo se deben utilizar cuando el período de ejecución difiera de cualquiera de los anteriores.

Un claro **ejemplo de uso correcto de `cron.d`, es el archivo php5 encargado de eliminar las sesiones inactivas**. ¿A que no te lo esperabas? Sin buscarlo, de pronto encuentras la respuesta a **¿cómo hace PHP para eliminar las sesiones inactivas tras un período de tiempo determinado?**. Y la respuesta

justamente, es que **PHP agrega un archivo php5 a `/etc/cron.d`** indicando como período de ejecución (formato `crontab`) el indicado en la directiva `session.gc_maxlifetime` del archivo `php.ini` y procediendo a ejecutar la instrucción encargada de destruir toda sesión cuyo período de inactividad, supere el mencionado.

```
eugenia@cococha-gnucita:/etc/cron.d$ cat php5
# /etc/cron.d/php5: crontab fragment for php5
#  This purges session files older than X, where X is defined in seconds
#  as the largest value of session.gc_maxlifetime from all your php.ini
#  files, or 24 minutes if not defined.  See /usr/lib/php5/maxlifetime

# Look for and purge old sessions every 30 minutes
09,39 *     * * *     root   [ -x /usr/lib/php5/maxlifetime ] && [ -d /var/lib/php5 ] &&
find /var/lib/php5/ -depth -mindepth 1 -maxdepth 1 -type f -cmin +$
(/usr/lib/php5/maxlifetime) ! -execdir fuser -s {} 2>/dev/null \; -delete
```

> *Los archivos* `cron.d` *deben tener el formato requerido por* `crontab` *para poder ser ejecutados por el sistema de forma satisfactoria*

Especificaciones Debian

Como especificaciones puntuales para los paquetes Debian (`.deb`), se deben tener en cuenta sus normas. Los archivos cuyo destino sean las carpetas `/etc/cron.*` seguirán estas pautas:

- El **formato del archivo** debe ser un guión de *shell* para todos los casos excepto para `cron.d` en el que debe tener el formato sugerido por crontab;
- Los archivos deben tener **permisos de ejecución** (se debe hacer un `chmod +x` a cada archivo `.cron.*`);
- Los archivos **deben guardarse en la carpeta DEBIAN** del paquete;
- El **nombre de archivo** debe tener el formato: *`nombre_del_paquete`*`.cron.*` dónde:
  `nombre_del_paquete` será el de la aplicación. Por ejemplo, `apache2`
  `*` será `hourly`, `daily`, `weekly`, `monthly` o `d` según corresponda.

Por ejemplo, un archivo **`DEBIAN/apache2.cron.daily`** se alojará en `/etc/cron.daily/apache2` y será ejecutado diariamente.