Gestionar Software - Paquetes

Paquetes ¿Qué son?

Los paquetes como su nombre indica son paquetes de software que incluyen los ficheros, en algunos casos muchos, que componen una aplicación. Estos paquetes pueden incluir ficheros fuente o bien binarios, numero de versión, así como las referencias a las librerías comunes que necesite, ahora explicaré que significa cada cosa.

Ficheros fuente, son aquellos escritos directamente por los desarrolladores de la aplicación, si utilizamos estos paquetes tenemos algunas ventajas e inconvenientes. Empezaré por los inconvenientes, debemos compilar nosotros mismos el software, con lo que debemos saber hacerlo, y hacerlo bien, ya que si lo hacemos de forma descuidada, o errónea podríamos entre otras cosas generar una aplicación con permisos de root lo que le concedería acceso total a nuestro sistema lo que podría ser un agujero de seguridad muy importante. Como ventaja es que la aplicación estaría adaptada totalmente a nuestro sistema, pudiendo así aprovechar los recursos del mismo plenamente, otra ventaja es que al tener acceso al código fuente, si tenemos ganas y paciencia podremos comprobar que la aplicación no realice acciones indeseadas (p.e. acceso a información confidencial).

Ficheros binarios, son aquellos compilados con unos valores genéricos, lo que les hace aptos u operativos para una arquitectura concreta o varias (tipos de procesares, familia Intel i386 X64, Sparc, powerPC, ...). Los inconvenientes principales son una posible disminución en el aprovechamiento de los recursos del sistema, otro inconveniente seria que si el paquete proviene de una fuente desconocida, no podremos comprobar si realiza alguna acción no deseada. Como ventaja principal la comodidad, prácticamente no tendremos que hacer nada para utilizar la aplicación.

Librerías comunes, en las aplicaciones hay tareas repetitivas que pueden ser utilizadas por varias aplicaciones, minimizar una ventana, barras de desplazamiento, gestión de memoria, operaciones en coma flotante, etc. Pues bien los ficheros que se encargan de realizar todas estas tareas podrían estar incluidas en la propia aplicación, lo que nos llevaría en el caso de tener funcionando varias aplicaciones que utilicen estas tareas, a unas necesidades de memoria desorbitadas, por lo que se utilizan unas librerías compartidas que realizan estas tareas y que se van cargando si no están cargadas ya por otra aplicación en memoria según se van necesitando. aparte de los recursos que liberamos con esta técnica, los desarrolladores evitan tener que reprogramar cada vez que necesiten de estas funcionalidades, dependiendo de la arquitectura, aplicación, etc.

Estas **librerías comunes**, o **funcionalidades compartidas** se conocen como **dependencias**, lo que viene a querer decir que la aplicación del paquete que queremos instalar necesita ciertas librerías para poder funcionar, ya que va a intentar utilizar las funcionalidades ofrecidas por estas.

Gestores de Paquetes

Para instalar las aplicaciones, podemos copiar los archivos que la componen en los directorios adecuados, buscar las dependencias en la versión adecuada, ver si ya están instaladas en nuestro sistema, en las rutas que la nueva aplicación espera, si no lo estan hacerlo ....,

Por el contrario si lo que queremos es desinstalar alguna aplicación, podemos buscar su ubicación, la de los archivos de configuración, las dependencias y comprobar que estas nos son utilizadas por otra aplicación, y eliminarlo todo.

O bien podemos utilizar un gestor de paquetes que realizara todas estas tareas, y otras que no he comentado por abreviar, por nosotros.

Cada distribución utiliza un gestor especifico, que es el que suele venir por defecto, aunque si nos gusta cualquier otro normalmente funcionará, ahora bien, aunque si no es estrictamente necesario, lo recomendable es utilizar el que la distribución aconseja. Estos gestores de paquetes nos facilitan la instalación, actualización y eliminación de software, manteniendo una base de datos en la que se refleja el el software instalado, versiones, dependencias,...

Aunque los gestores de paquetes pueden funcionar en distribuciones en las que no vienen instalados por defecto, hemos de tener en cuenta que cada uno tiene sus peculiaridades, por lo que en un sistema con varios gestores de paquetes, podemos instalar varias veces el mismo software (aplicaciones, bibliotecas ...) una vez por cada gestor de paquetes, ya que no comparten las bases de datos, y ocasionar inestabilidades en el sistema e incluso su colapso y quedar inutilizable. Dicho esto si decidimos utilizar varios gestores de paquetes debemos ser muy cuidadosos a la hora de realizar las acciones que nos permiten.

Cada gestor de paquetes tiene sus propios ficheros de configuración en los cuales entre algunas cosas mas se guarda los lugares donde buscar el software, llamados repositorios, aunque algunos como yum buscara en varias ubicaciones intentando localizar el paquete solicitado

Gestores de Paquetes Comunes

RPM Gestor de paquetes utilizado por las distribuciones derivadas de Red Hat, es uno de los mas extendidos. podéis obtener información de su uso desde su web [1]

YUM Gestor de paquetes de Yellow Dog y que utilizan algunas distribuciones mas. Información sobre este gestor en español en esta wiki de fedora project sobre YUM. [2]

DPKG y APT para Debian y sus distribuciones derivadas, mas información en esta wiki de Debian, para APT [3]

Como yo utilizo Ubuntu que es una distribución derivada de Debian explicaré como realizar alguna de las tareas mas habituales, que podemos realizar con APT, aunque en la actualidad existen interfaces gráficas mucho mas comodas de utilizar y normalmente es lo que se utiliza. En Ubuntu lo llaman Centro de Software Ubuntu, y habitualmente tiene una entrada en el menú principal, o lanzador (en unity), si nos decantamos por esta opción, que será lo mas probable, tendremos una larga lista de aplicaciones para instalar.

Si queremos instalar una aplicación que no aparece en la lista inicial, será suficiente con buscar en google y añadir el repositorio a la lista de lugares donde buscar aplicaciones, se puede hacer desde el entorno gráfico o bien siguiendo las instrucciones que suele poner en la web de los desarrolladores, debemos tener en cuenta que solo debemos incluir repositorios que merezcan nuestra confianza, ¿como...?, efectivamente buscamos en google a ver si ese repositorio tiene entradas en internet que digan algo al respecto

apt

apt es una aplicación importante, por lo que debemos ejecutarlo como un usuario con los permisos necesarios.

Instalar un paquete (normalmente tendrá en cuenta sus dependencias)

```
sudo apt-get install nombre-de-paquete
```

Eliminar un paquete

```
sudo apt-get remove nombre-de-paquete
```

Eliminar ficheros de paquetes eliminados y los de configuración

```
sudo apt-get purge
```

Existen mas opciones interesantes, algunas nos permiten descargar los paquetes fuentes, compilarlos, instalarlos y tener en cuenta/forzar la satisfacción de sus dependencias

Utilizar paquetes de una distribución distinta - alien

Cada paquete esta construido para su distribución, o dicho de otro modo, cada aplicación debe tener un paquete construido para cada gestor con el que pueda ser utilizado, si no existe podemos construirlo a partir del los ficheros fuente.

Si tenemos un paquete para una distribución y queremos utilizarlo en otra para el que no existe paquete, podemos utilizar una aplicación llamada `alien`, con una simple busqueda en google encontrareis varios tutoriales sobre el mismo (ademas de man, info, apropos).

Bibliotecas Compartidas

Bibliotecas Compartidas - ¿Qué son?

Las Bibliotecas compartidas son ficheros que contienen colecciones de fragmentos de código, que nos permite el poder utilizarlos en el nuestro código o/y por los programas que tenemos instalados en nuestra máquina, sin tener que volver a programar ciertas tareas rutinarias (p.e. crear ventanas, botones, acceso a dispositivos, ...).

Además de ahorrarnos tiempo a la hora de desarrollar aplicaciones, y posibles errores, estas bibliotecas al ser compartidas se instalan en nuestra máquina y como su nombre indica son o pueden ser compartidas por varios programas, consiguiendo así:

- Ahorrar espacio de almacenamiento
- Se cargan una sola vez en memoria, independientemente de las aplicaciones que las requieran

Cada aplicación suele utilizar una versión concreta de cada biblioteca compartida, pero esto en Linux no es ningún problema ya que Linux utiliza un sistema de numeración de versiones, que posibilita el tener instaladas varias versiones de cada biblioteca.

Las aplicaciones deben poder encontrar las bibliotecas, por lo que los cambios deben ser reflejados en unos ficheros de configuración, que son los que las aplicaciones consultan para poder localizarlas.

Si una biblioteca deja de estar accesible, las aplicaciones fallaran, o incluso lo hará el sistema completo.

Todo esto nos lleva a que debemos tener un buen control de nuestras bibliotecas, que normalmente no nos causará problemas si utilizamos el gestor de paquetes correspondiente a nuestra distribución (y no utilizamos opciones que ignoren dependencias).

Los nombres de las bibliotecas compartidas en Linux terminan en .so (shared object, objeto compartido).

Rutas de los ficheros de Biblioteca

Algunas de las rutas son tenidas en cuenta por Linux aunque no las especifiquemos como `/lib` y `/usr/lib`

Existen varias formas de definir la ruta a los ficheros de biblioteca.

Editando el fichero `/ etc/ld.so.conf` . En este fichero se encuentran algunas de las rutas a bibliotecas, una por línea.

En `/ etc/ld.so.conf` pueden existir líneas `include` que como saben los programadores de C indica que se debe incluir el contenido de otro/s fichero/s, p.e. en Ubuntu contiene una línea `include / etc/ld.so.conf.d/*.conf` que indica que se deben incluir todos los ficheros que terminen en .conf dentro del directorio `/ etc/ ld.so.conf.d/` y estos a su vez pueden contener otras líneas `include`. En otras distribuciones se adoptan soluciones parecidas aunque con algunas pequeñas diferencias.

Linux en una de sus tareas de arranque lee todos los ficheros que puedan estar involucrados en la definición de rutas a bibliotecas (incluidas aquellas que tiene siempre `/lib` `/usr/lib`). y genera un sistema de datos mas optimizado que los ficheros en texto plano que se utilizan para configurar las rutas, por eso cuando modifiquemos un fichero de en el que se encuentren rutas a bibliotecas debemos ejecutar la herramienta `ldconfig` que se encarga de leer los ficheros de configuración y volver a generar este sistema de datos optimizado con las rutas de las bibliotecas.

Normalmente no tendremos que realizar esta tarea, ya que utilizaremos el gestor de paquetes de nuestra distribución el cual, normalmente, realiza todas las acciones necesarias al instalar un nuevo paquete. Estas tareas solamente nos veremos obligados a hacerlas cuando instalemos alguna biblioteca compilada por nosotros mismos, o en circunstancias poco habituales.

Cambiar la Ruta de Bibliotecas de Forma Temporal

Imaginemos que queremos utilizar temporalmente una biblioteca, p.e. para comprobar su efecto en el sistema, podemos guardarla en una carpeta que tenemos para estos temas quizá ~/pruebas/lib (recordad que ~ hace referencia a la carpeta home de usuario). Ahora que ya tenemos localizada la ruta para probar nuestra librería, podemos utilizar la variable de entorno `LD_LIBRARY_PATH` las rutas añadidas de este modo serán colocadas al principio de la cadena de búsqueda interna para las librerías por lo que siempre se utilizarán antes que las que se estaban utilizando hasta el momento, si es que sustituye a alguna.

El comando a utilizar sería

```
export LD_LIBRARY_PATH=~/pruebas/lib/libreria1:~/pruebas/lib/libreria2
```

Como podemos ver se pueden poner tantas rutas como necesitemos separadas por ":" , no tienen porque estar en la carpeta de usuario, en principio somos libres de utilizar la ruta que necesitemos, siempre y cuando tengamos permiso para acceder a ella.

Para esta tarea no es necesario ejecutar `ldconfig` para actualizar las rutas.

Detectar y Corregir Problemas con las Rutas de las Bibliotecas

Si estamos utilizando un entorno gráfico, si algún programa no se ejecuta por problemas con las librerías, normalmente no veremos la razón del fallo. Lo podremos averiguar de varias formas pero normalmente en nuestra consola, podemos escribir el nombre el programa para ejecutarlo y si se produce un error por la falta de algunas librerías nos devolverá un mensaje indicándonos que librerías nos faltan, o que su ruta no está registrada correctamente.

Supongamos que un programa nos devuelve un error indicándonos que no encuentra la librería "libreriaQueFalta.so", si normalmente utilizamos el gestor de paquetes, esto no debería suceder.

Primero buscaríamos el fichero a ver si esta instalado, p.e. podemos utilizar `find` y si no lo encontramos buscaremos el paquete al que pertenece y lo instalaremos con el gestor de paquetes.

Si el fichero de biblioteca lo tenemos instalado, probablemente el problema esté en que no se encuentra en la ruta que el programa espera, podremos averiguar cual es esta ruta que el programa espera ejecutando ldd /ruta/al/programa/programa , también puede suceder que tengamos instalada una revisión de la versión que no es exactamente la que el programa espera, si esta es una revisión menor p.e. tenemos la versión libreriaQueNecesitamos.so.3 y la que tenemos es la libreriaQueNecesitamos.so.3.4, por norma estas revisiones solamente tienen correcciones menores, por lo que podremos utilizarla.

Tanto si nuestro problema es que la ubicación no es la que espera el programa como si es que la versión de la que podemos disponer es una revisión menor (siempre si la que tenemos es superior a la esperada) la solución es la misma:

Crear un enlace simbólico como root y encontrándonos en el directorio donde tengamos guardada la biblioteca

```
sudo ln -s libreriaQueNecesitamos.so.3.4 libreriaQueNecesitamos.so.3
```

ó en su caso desde la carpeta donde se espera que este la biblioteca utilizando la ruta completa de la biblioteca original

```
sudo ln -s /ruta/a/la/libreria/instalada/libreriaQueNecesitamos.so
libreriaQueNecesitamos.so
```

Tras esto debemos ejecutar `ldconfig` para que se actualicen las rutas en el sistema.

Francisco Javier López Torrijos
Analista Sistemas Informáticos de Gestión
Diseño y Desarrollo Web

URL de origen (modified on 02/12/2015 - 20:57): http://www.lopeztorrijos.com/node/36

Enlaces
[1] http://www.rpm.org

[2] https://fedoraproject.org/wiki/FedoraProject:Translating/es
[3] http://wiki.debian.org/es/Apt