

ADMINISTRACIÓN DE PAQUETES

INTRODUCCIÓN

La instalación de programas que conocemos en sistemas operativos como Microsoft Windows se realiza invariablemente a través de la ejecución de un instalador (por ejemplo **setup.exe**). Este instalador es propio y único de cada programa disponible para instalar. Es decir, por ejemplo, que el setup.exe de Microsoft Word no funciona para instalar Adobe Acrobat Reader DC, aún pudiendo tener el mismo nombre.

Esta situación se debe a que los programas a ser instalados y sus instaladores no siguen un estándar único, lo que conlleva a múltiples formatos de instalación.

La instalación de programas en GNU/Linux es diferente a la de otros sistemas operativos. Siguen un estándar para el desarrollo de su software y asimismo para su instalador. Este estándar implica que cada programa a ser instalado se representa como un **paquete** de software.

PAQUETES DE SOFTWARE

Una **paquete** de software es un conjunto de archivos que contiene lo necesario para instalar, desinstalar, configurar y ejecutar un programa en particular, utilizando herramientas de gestión de software del sistema operativo.

Existen 2 tipos de paquetes de software:

- **Paquetes Binarios:** contienen los programas en código máquina y los archivos necesarios para ejecutar las aplicaciones dentro del sistema. Es software precompilado de acuerdo a la arquitectura (amd64, x86, SPARC). Existen varios tipos de paquetes binarios dependiendo de la

distribución Linux, por ejemplo Debian/GNU Linux y sus derivados utilizan paquetes **.deb** y distribuciones como Fedora, Red Hat paquetes **.rpm**

- **Paquetes de Código Fuente:** contienen los códigos fuente y los archivos necesarios para compilar e instalar en forma manual los programas. Estos paquetes sirven para cualquier distribución. Son archivos comprimidos y se reconocen por terminar en **.tgz**, **.tar.gz** o **.tar.bz2**

RELACIÓN ENTRE PAQUETES DE SOFTWARE

- Un programa puede estar contenido en un solo paquete, pero también es común encontrar un programa que consiste de varios paquetes relacionados entre ellos.
- También es posible que varios programas pequeños y relacionados entre sí se encuentren en el mismo paquete: por ejemplo, el paquete *fileutils* que contiene varias órdenes de Unix, tales como *ls*, *cp*, etc.
- Algunos paquetes requieren de otros para funcionar. En Debian, algunos paquetes pueden depender de otro, recomendar, sugerir, romper, o entrar en conflicto con otros paquetes.

DEPENDENCIA ENTRE PAQUETES

- Si un **paquete A depende de otro paquete B**, entonces B es necesario para que A funcione correctamente. Por ejemplo, el paquete *gimp* depende del paquete *gimp-data* para permitir que el editor gráfico GIMP pueda acceder a sus ficheros críticos de datos.
- Si un **paquete A recomienda otro paquete B**, entonces B ofrece una importante funcionalidad adicional para A que sería deseable en la mayoría de las circunstancias. Por ejemplo, el paquete *mozilla-browser*

recomienda el paquete *mozilla-psm*, que añade la capacidad para la transferencia segura de datos al navegador web de Mozilla.

- Si un **paquete A sugiere otro paquete B**, entonces el paquete B ofrece a A una funcionalidad que puede que mejore A, pero que no es necesaria en la mayoría de los casos. Por ejemplo, el paquete *kmail* sugiere el paquete *gnupg*, el cual contiene software de cifrado que KMail puede emplear.
- Si un **paquete A entra en conflicto con otro paquete B**, los dos paquetes no se pueden instalar a la vez. Por ejemplo, *fb-music-hi* entra en conflicto con *fb-music-low* porque ofrecen conjuntos alternativos de sonidos para el juego Frozen Bubble.

INSTALACIÓN DE UN PAQUETE DE SOFTWARE

Para la instalación de un programa se utiliza un **Gestor de Paquetes** que asiste al usuario en la tarea de administrar el conjunto de paquetes.

- Un **Gestor de Paquetes** es una colección de herramientas (comandos) que sirven para automatizar el proceso de instalación, actualización, configuración y eliminación de paquetes de software. Debian y derivados utilizan las herramientas de alto nivel **apt** y **aptitude**.
- Un **Gestor de Paquetes** mantiene un registro del software que está instalado en el sistema y permite instalar software nuevo, actualizarlo a versiones más recientes, o eliminar software de una manera sencilla y centralizada.
- Un **Gestor de Paquetes** forma parte del sistema operativo y posee un único formato de paquetes.
- Los paquetes incluyen, además del programa mismo, nombre completo, descripción de su funcionalidad, número de versión, y una lista de otros

paquetes requeridos para su funcionamiento. El Gestor de Paquetes utiliza esta información para manipularlo.

PROCESO DE INSTALACIÓN DE UN PROGRAMA CON EL GESTOR DE PAQUETES

1° Paso: el usuario solicita instalación de un programa por medio de un comando del Gestor (por ejemplo: `#apt-get install tree` es el pedido para instalar el comando tree).

2° Paso: el Gestor busca el paquete de software correspondiente al programa solicitado en el repositorio (url, discos) definido en el archivo de texto: **/etc/apt/sources.list**

3° Paso: el Gestor ubica los paquetes necesarios, los descarga, los desempaqueta y copia sus archivos binarios, archivos de configuración y manuales a sus respectivos directorios dejando a los mismos listos para su ejecución. Puede realizar sugerencias sobre la instalación y el programa queda listo para ser utilizado.

REPOSITORIOS

- El software se pone a disposición de los usuarios en los repositorios, con el fin de proporcionar un sencillo control sobre los diferentes tipos de software que van a instalar en su sistema.
- Cada distribución mantiene organizados los paquetes en repositorios, lo que permite actualizar e instalar por red todo el sistema desde una localización confiable (repositorios oficiales).
- Son directorios en servidores especiales que únicamente mantienen paquetes e información de estos, en una estructura similar a una base de datos.

- Podemos utilizar estos repositorios para actualizar los programas instalados en nuestro sistema, o para instalar programas que no se encuentren en nuestra distribución.
- Pueden ser de acceso público, o pueden estar protegidos y necesitar de una autenticación previa. Los depósitos más conocidos son los de carácter académico e institucional.

ARCHIVO `/etc/apt/sources.list`

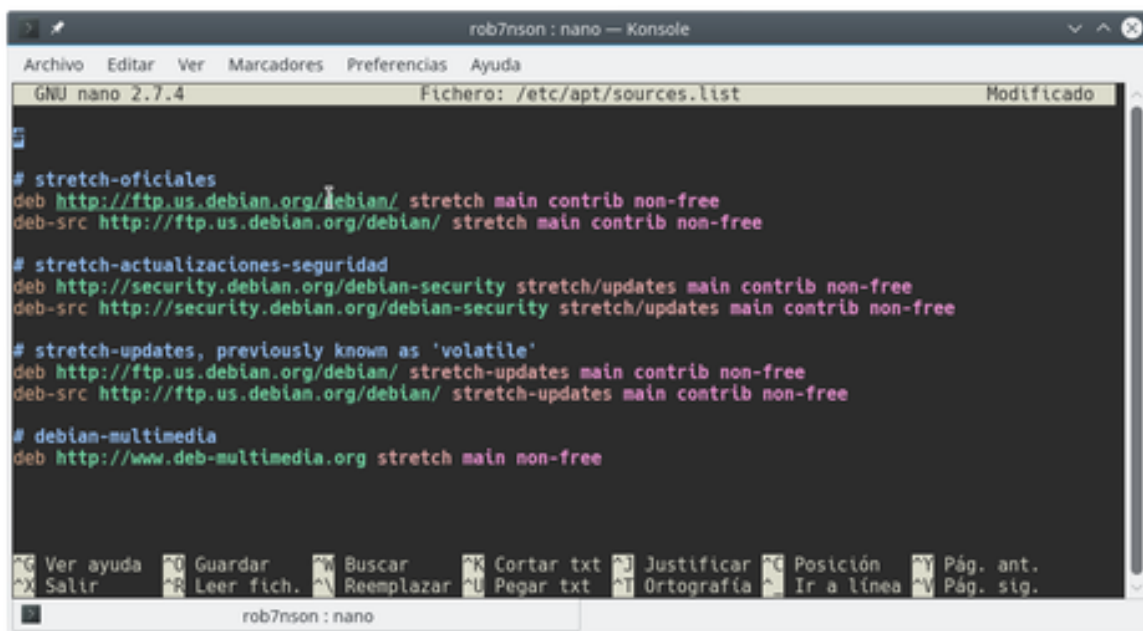
- Es un archivo de texto que contiene una lista de direcciones (fuentes) con las ubicaciones de los repositorios de paquetes de nuestra distribución.
- Antes de instalar programas debemos chequear nuestro archivo `sources.list` y actualizarlo de ser necesario.
- Se puede editar y agregar otras fuentes diferentes a las configuradas durante la instalación con cualquier editor de texto. Por ejemplo:

```
root@mipc:~#nano /etc/apt/sources.list
```

- Luego de editar `sources.list`, siempre actualizar con la línea de comandos:

```
root@mipc:~#apt-get update ó root@mipc:~#aptitude update
```

- Un archivo `sources.list` puede verse como la siguiente imagen al editarlo:



```
rob7nson : nano — Konsole
Archivo  Editar  Ver  Marcadores  Preferencias  Ayuda
GNU nano 2.7.4  Fichero: /etc/apt/sources.list  Modificado

# stretch-oficiales
deb http://ftp.us.debian.org/debian/ stretch main contrib non-free
deb-src http://ftp.us.debian.org/debian/ stretch main contrib non-free

# stretch-actualizaciones-seguridad
deb http://security.debian.org/debian-security stretch/updates main contrib non-free
deb-src http://security.debian.org/debian-security stretch/updates main contrib non-free

# stretch-updates, previously known as 'volatile'
deb http://ftp.us.debian.org/debian/ stretch-updates main contrib non-free
deb-src http://ftp.us.debian.org/debian/ stretch-updates main contrib non-free

# debian-multimedia
deb http://www.deb-multimedia.org stretch main non-free

Ver ayuda  Guardar  Buscar  Cortar txt  Justificar  Posición  Pág. ant.
Salir  Leer fich.  Reemplazar  Pegar txt  Ortografía  Ir a línea  Pág. sig.

rob7nson : nano
```

Dentro de este archivo, las líneas que inician con un símbolo # corresponden a comentarios o fuentes comentadas que no se utilizan por algún motivo.

Las demás líneas corresponden a fuentes (ubicaciones de repositorios). A su vez, éstas tienen varias secciones, a saber:

```
deb http://deb.debian.org/debian stretch main
```

- **tipo:** Pueden ser de dos tipos: **deb** (típico paquete binario de Debian) y **deb-src** (código fuente oficial del paquete)
- **uri:** Identificador Universal de Recursos, tipo de recurso de la cual se obtienen los paquetes.
 - *CD-ROM*: El cdrom permite a APT usar la unidad de CD-ROM local. Se puede usar el programa apt-cdrom para añadir entradas de un cdrom al fichero sources.list de manera automática, en modo consola.
 - *FTP*: Especifica un servidor FTP como archivo.
 - *HTTP*: Especifica un servidor HTTP como archivo.

- *FILE*: Permite considerar como archivo a cualquier fichero en el sistema de ficheros. Esto es útil para particiones montadas mediante *NFS* (sistema de ficheros usado para montar particiones de sistemas remotos) y réplicas locales.
- **distribución**: Distribución instalada. Cada distribución cuenta con sus paquetes.
- **componentes**: Los componentes son los tipos de repositorios clasificados según las licencias de los paquetes que contienen. Dentro de los componentes tenemos **main**, **contrib** y **non-free**.

COMPONENTES O TIPO DE REPOSITORIO

De acuerdo a nuestra filosofía en cuanto a software libre, a la hora de instalar paquetes, podemos elegir seguir una corriente, dos o las tres. En nuestro sistema se van a instalar paquetes según lo especificado en el archivo `sources.list`

- **main**

En este directorio se encuentran los paquetes 100% libres, esto quiere decir que cumplen o están de acuerdo con las directivas de Debian, en donde marcan cuando un paquete se le puede considerar que es 100% software libre.

- **non-free**

Aquí se encuentran paquetes que no pueden considerarse software libre según las directivas de Debian, por dar un ejemplo, hay software que puede ser distribuido e instalado, pero no se tiene acceso a su código fuente (No todos de esta sección son así hay software que si se proporciona su código fuente), simplemente por la licencia que trae el software de este paquete no cuadra con las directivas de Debian.

- **contrib**

En este directorio se pueden encontrar software libre, pero depende de alguna forma de un paquete que no es 100% libre

GESTOR DE PAQUETES DE DEBIAN Y DERIVADOS

- Debian cuenta con la herramienta de alto nivel **apt** como gestor de paquetes.
- **Advanced Packaging Tool** (Herramienta Avanzada de Empaquetado) es un Gestor de Paquetes creado por el proyecto Debian.
- Contiene un grupo de comandos que permite:
 - instalar o eliminar un paquete,
 - actualizar el sistema,
 - listar paquetes disponibles, etc.
- Los comandos de apt descargan los paquetes desde repositorios **resolviendo e instalando automáticamente todas las dependencias** de cada paquete a ser instalado, recomendando la instalación de otros posiblemente relacionados.

OPERACIONES PRINCIPALES CON EL COMANDO APT

```
#apt-get install paquete
```

Descarga el paquete, junto con todas sus dependencias y los instala o actualiza; los paquetes bajados se descargan en /var/cache/apt/archives.

```
#apt-get -f install
```

Repara e instala dependencias pendientes que necesita el sistema.

```
#apt-get remove paquete
```

Desinstala el paquete del sistema.


```
#apt-get remove --purge paquete
```

Desinstala el paquete del sistema y todos los que dependan de él.

```
#apt-get update
```

Actualiza la lista de paquetes disponibles y sus versiones a partir de los repositorios disponibles. Debería usarse si hay que instalar algún paquete nuevo o cada vez que se modifique /etc/apt/sources.list.

```
#apt-get upgrade
```

Actualiza todos los paquetes instalados a las últimas versiones disponibles según el último 'update'. No instalará nuevos paquetes ni borrará de viejos. Si un paquete cambia sus dependencias y requiere la instalación de otro paquete, no será actualizado, sino que será puesto en estado 'hold'. apt-get upgrade no actualizará ningún paquete puesto en 'hold'. También es útil la opción '-u' para saber qué paquetes van a ser actualizados.

```
#apt-cdrom add
```

Lo primero que hay que hacer si se instala un sistema Debian desde CD, añadir los paquetes de los CDs a la base de datos, pues apt sólo trabaja con lo que tiene en su base de datos.

```
#apt-get autoclean
```

Elimina los paquetes que hay en /var/cache/apt/archives cuya versión no está indicada en la base de datos de paquetes (útil para cuando se hace un update y se quieren borrar los paquetes anteriormente bajados cuya versión disponible es ahora superior).

```
#apt-get clean
```

Borra todos los paquetes en /var/cache/apt/archives.

```
#apt-cache search expresion_regular
```

Busca un patrón en los nombres de paquetes y sus descripciones. Donde expresion_regular es una cadena de caracteres que representa al paquete buscado.

```
#apt-cache show paquete
```

Muestra la descripción completa del paquete.

COMANDO [APTITUDE](#) COMO OPCIÓN A APT

Actualizar repositorios:

```
#aptitude update
```

Actualizar todos los paquetes:

```
#aptitude upgrade
```

Instalar paquete:

```
#aptitude install programa
```

Eliminar paquete:

```
#aptitude remove programa
```

Eliminar paquete con todos sus ficheros de configuración:

```
#aptitude remove --purge programa
```

Bloquear paquete para que no se actualice o elimine:

```
#aptitude hold programa
```

Desbloquear paquete:

```
#aptitude unhold programa
```

Descripción de un paquete:

```
#aptitude show programa
```

Limpiar caché de paquetes:

```
#aptitude clean
```

Descargar un paquete al directorio actual:

```
#aptitude download programa
```

Instalar solo las dependencias para un paquete:

```
#aptitude build-deps programa
```

CUANDO NO PODEMOS UTILIZAR APT O APTITUDE

Las herramientas **apt** y **aptitude** sólo pueden instalar paquetes **.deb** desde un repositorio.

No todos los programas se pueden instalar con **apt**. Generalmente se trata de paquetes de programas (por ejemplo el navegador Chrome, Skype, Dropbox y otros) que no se encuentran en los repositorios pero sí están disponibles para descargar de la web, en un DVD-ROM o una unidad de almacenamiento. Para manipular estos paquetes en las distribuciones Debian y derivados se recurre al comando **dpkg**.

COMANDO DPKG

- Es un programa de bajo nivel que gestiona paquetes **.deb**, permite la instalación, desinstalación y consulta de información de los paquetes instalados.
- A diferencia de **apt** **no instala automáticamente las dependencias**. Se limita a indicarnos durante el proceso de instalación.

- Una vez instalado un paquete **.deb** con **dpkg**, se puede ejecutar la línea de comandos para completar las dependencias faltantes o instalarlas de una con dpkg si se dispone de los paquetes de las dependencias.

USO DEL COMANDO DPKG

```
#dpkg -l (letra ele minúscula)
```

Comprueba los paquetes instalados en la máquina y ofrece un listado completo. Si queremos información relacionada con un solo paquete, se puede utilizar grep. Ejemplo: dpkg -l |grep tree

```
#dpkg -L nombrePaquete
```

Informa sobre el contenido (los ficheros) que forman un paquete.

```
#dpkg -i nombrePaquete
```

Para instalar paquetes que tenemos localmente y no necesitamos descargar

```
#dpkg -r nombrePaquete
```

Desinstala el paquete

```
#dpkg -P nombrePaquete
```

Elimina el paquete y archivos de configuración

```
#dpkg -s archivo
```

Informa de los paquetes que contienen ese fichero (a cuáles paquetes pertenece)

EJEMPLO DE INSTALACIÓN CON DPKG

Abrimos una terminal y nos logueamos como superusuario o usuario con privilegios de root (sudo). Nos ubicamos en el directorio en el cual descargamos el paquete **.deb**

```
#cd /home/usuario/Descargas
```

Ejecutamos la instalación:

```
# dpkg -i paquete.deb
```

El paquete se instalará y puede suceder que muestre un error si hay dependencias no satisfechas, en este caso, se resuelven con:

```
# apt-get -f install
```

Este comando descargará las dependencias requeridas, las instalará y concluirá la instalación del paquete **.deb** que quedó interrumpida.

INSTALACIÓN DE PAQUETES DESDE EL CÓDIGO FUENTE

Puede suceder que al querer instalar un programa no haya paquetes hechos para nuestra distribución o no se encuentren actualizados en los repositorios. En estas situaciones podemos descargar la versión de ese programa en CÓDIGO FUENTE (Source Code) desde la página oficial del programa o de desarrollo del mismo .

El Código Fuente es el programa escrito por el programador. Quiere decir que para poder usar dicho programa debemos compilarlo y crear un ejecutable compatible con nuestra distribución. Generalmente estos códigos vienen en lenguaje C y C++, aunque pueden aparecer en Python, Ruby, etc.

Suelen presentarse en formato **.tar.gz** o **tar.bz2** (o sea compactado con tar y comprimido con gzip o bzip).

Lo normal es que cada aplicación tenga la información en el fichero **README** o **INSTALL** de como instalarlo y que podemos leer una vez descomprimido el archivo **.tar.gz** o **tar.bz2**.

EJEMPLO DE COMPILACIÓN E INSTALACIÓN DESDE CÓDIGO FUENTE

En Debian y derivados, contar con el paquete **build-essential** instalado

1° PASO: Descargar el código fuente miprograma.tar.gz en un directorio (/opt) y situarse en el directorio.

```
root@mipc:~#cd /opt
```

2° PASO: Descomprimir el archivo:

```
root@mipc:/opt#tar -xvzf miprograma.tar.gz
```

3° PASO: Al descomprimir, se creará un directorio con el nombre del archivo comprimido. Situar en el directorio producto de la descompresión.

```
root@mipc:/opt#cd miprograma
```

4° PASO: Ejecutar el script **configure** el cual prepara al sistema para compilar el programa.

```
root@mipc:/opt/miprograma#./configure
```

5° PASO: Compilar el código fuente con el comando **make** para generar el código máquina o binarios ejecutables del programa.

```
root@mipc:/opt/miprograma#make
```

6° PASO: Copiar los binarios generados y archivos necesarios a los respectivos directorios con **make install**.

```
root@mipc:/opt/miprograma#make install
```

REFERENCIAS

<https://www.debian.org/doc/manuals/debian-reference/debian-reference.es.pdf>

<https://www.diversidadyunpocodetodo.com/instalar-paquetes-deb-con-dpkg-debian/>

<https://hackinglinux.wordpress.com/2013/08/01/tutorial-instalar-programas-desde-el-codigo-fuente-en-linux/>

<https://blog.carreralinux.com.ar/2018/03/instalar-desde-codigo-fuente-vieja-usanza/>