APT HOWTO (Obsolete Documentation)

Gustavo Noronha Silva <kov@debian.org>
Hugo Mora <h.mora@melix.com.mx>

1.8.4 - April 2003

Resumen

Este documento pretende proveer al usuario de conocimientos sobre el programa para manejar la paquetería de Debian, el APT. El propósito de este documento, es hacer la vida más fácil a los nuevos usuarios de Debian y ayudar a aquellos que desean tener un conocimiento más profundo de la administración de este sistema. Fue creado para el proyecto Debian para mejorar y ayudar al soporte existente y también a los usuarios de este sistema.

Nota de Copyright

Índice general

Capítulo 1

Introducción

En el principio existían los .tar.gz. Los usuarios tenían que compilar cada programa que quisieran usar en su sistema GNU/Linux. Cuando Debian fue creado, fue imperante que el sistema incluyera un programa que se encargara de manejar la paquetería instalada en la computadora. Este programa se llamó `dpkg`. Así fue como nació el primer "paquete" en el mundo GNU/Linux, aún antes de que Red Hat decidiera crear su propio programa "rpm".

Rápidamente llegó un nuevo dilema a las mentes de los creadores de GNU/Linux. Ellos necesitaban un modo fácil, rápido y eficiente de instalar programas, que manejara automáticamente las dependencias y se hiciera cargo de la configuración mientras se actualizan. Nuevamente Debian fue pionero y creó el APT, Herramienta Avanzada de Empaquetamiento (Advanced Packaging Tool), este programa ha sido adaptado por Conectiva para usarse con rpm y ha sido adoptado por otras distribuciones.

Este manual no cubre el apt-rpm, tal como se denomina a la adaptación de Conectiva del APT, y tal vez algunos "parches" para este documento aparecerán en algún tiempo.

Este capítulo está basado en la siguiente distribución de Debian `Sarge`.

Capítulo 2

Configuración Básica

2.1. El archivo /etc/apt/sources.list

Como parte de su funcionamiento, APT utiliza un archivo que enlista las "fuentes" en donde se encuentran los paquetes. Este archivo es: `/etc/apt/sources.list`.

El contenido de este archivo, normalmente sigue este formato:

```
deb http://host/debian distribución sección1 sección2 sección3
deb-src http://host/debian distribución sección1 sección2 sección3
```

Por supuesto que los renglones arriba mencionados son ficticios y no deberán ser usados. La primera palabra en cada línea, `deb` o `deb-src`, indican el tipo del archivo: ya sea que contenga paquetes binarios (`deb`), esto es, los paquetes pre-compilados que normalmente se usan, o los paquetes fuente (`deb-src`), que son los códigos originales, más el archivo de control de Debian (`.dsc`) y el `diff.gz` que contienen los cambios necesarios para "debianizar" el programa.

Generalmente se encuentra lo siguiente por defecto en el archivo sources.list:

```
# See sources.list(5) for more information, especialy
# Remember that you can only use http, ftp or file URIs
# CDROMs are managed through the apt-cdrom tool.
deb http://http.us.debian.org/debian stable main contrib non-free
deb http://non-us.debian.org/debian-non-US stable/non-US main contrib non-fre
deb http://security.debian.org stable/updates main contrib non-free

# Uncomment if you want the apt-get source function to work
#deb-src http://http.us.debian.org/debian stable main contrib non-free
#deb-src http://non-us.debian.org/debian-non-US stable/non-US main contrib no
```

Éstas son las líneas necesarias para una instalación básica de Debian. La primer línea `deb` apunta al archivo en el servidor oficial, la segunda línea apunta hacia el archivo del servidor non-US y la tercera línea apunta hacia el archivo del servidor de actualizaciones de seguridad.

Las dos últimas líneas están deshabilitadas (con un "#" al inicio), así que apt-get las ignora. Éstas son las líneas de `deb-src`, esto es, apuntan hacia los paquetes fuente de Debian. Si frecuentemente descarga paquetes fuente para probar o recompilar, habilítelas (borrando el "#" al inicio de la línea).

El archivo `/etc/apt/sources.list` puede contener varios tipos de líneas. APT sabe como interpretar líneas del tipo `http`, `ftp`, `file` (archivos locales, p.e., un directorio que contiene un CD-ROM) y `ssh`, según mis conocimientos.

No olvide ejecutar el comando `apt-get update` después de modificar el archivo `/etc/apt/sources.list`. Debe hacer esto para permitir a APT obtener la lista de paquetes de las fuentes que especificó.

2.2. Cómo utilizar APT localmente

Algunas veces se pueden llegar a tener muchos paquetes .deb, los cuales le gustaría instalar utilizando APT para que las dependencias fueran resueltas automáticamente.

Para hacer esto, cree un directorio y coloque los .debs que quiera señalar en el. Por ejemplo:

```
# mkdir /root/debs
```

Usted puede modificar la lista de definiciones en el archivo de control de paquetes para su depósito con un archivo `override`. Dentro de este archivo se pueden definir opciones para descartar las que vienen por defecto en el paquete. Por ejemplo:

```
touch archivo
```

Dentro de este archivo se pueden definir opciones para descartar las que vienen por defecto en el paquete. Por ejemplo:

```
paquete prioridad sección
```

Paquete es el nombre del paquete o programa, la prioridad puede ser baja, media o alta y sección es la sección a donde pertenece. El nombre de archivo no importa, deberá pasarlo como argumento después para `dbpkg-scanpackages`. Si no desea escribir un archivo `override` entonces utilice `/dev/null` cuando ejecute `dbpkg-scanpackages`.

Continuando en el directorio /root se hace lo siguiente:

```
# dpkg-scanpackages debs archivo | gzip > debs/Packages.gz
```

En la línea anterior, *archivo* es el archivo de `override`, el comando genera un archivo `debs /Packages.gz` el cual contiene información acerca de los paquetes, la cual es utilizada por APT. Para utilizar los paquetes, finalmente agregue:

```
deb file:/root debs/
```

Después de eso, utilice los comandos de APT como siempre. También podría generar un deposito de fuentes de paquetes. Para hacer eso, haga exactamente lo mismo, pero recuerde que necesita tener los archivos `.orig.tar.gz`, `.dsc` y `.diff.gz` en el directorio y cambie `Sources.gz` por `Packages.gz`. El programa utilizado también es diferente. Es el `dpkg-scansources`. El comando completo se vería así:

```
# dpkg-scansources debs | gzip > debs/Sources.gz
```

Observe que `dpkg-scansources` no necesita un archivo `override`. La línea de sources.list es:

```
deb-src file:/root debs/
```

2.3. Decidiendo cual archivo sources.list es el mejor: netselect, netselect-apt.

Una duda frecuente, principalmente entre los usuarios novatos es: "¿Cuál servidor Debian debo incluir en el archivo `sources.list`?". Hay muchas formas de decidir esto. Los expertos podrían tener un script que determine el tiempo de un ping entre los distintos servidores. Pero hay un programa que hace esto por nosotros: **netselect**.

Para instalar netselect, hágalo así:

```
# apt-get install netselect
```

Al ejecutarlo sin algún parámetro muestra la ayuda. Ejecutándolo con una lista de servidores separada por un espacio, regresará una lista con la puntuación del host. Esta lista considera el tiempo de ping y el número de "saltos" (servidores por los cuales una petición de red debe pasar para alcanzar su destino) y es inversamente proporcional a la velocidad de descarga (la menor es la mejor). El servidor que regresa es aquel que tiene una puntuación menor (se puede ver la lista completa agregando la opción -vv). Ejemplo:

```
# netselect ftp.debian.org http.us.debian.org ftp.at.debian.org download.unes
 365 ftp.debian.org.br
#
```

Esto significa que, de los servidores incluidos como parámetros en netselect, `ftp.debian.org.br` fue el mejor, con una puntuación de 365. (Atención, la puntuación depende de la localización geográfica, la topología de red, la distancia entre la computadora y

los servidores, por lo tanto, dependiendo de la computadora en la que se ejecute el comando podría ser otro el mejor servidor).

Ahora agregue el servidor más rápido encontrado por netselect en el archivo `/etc/apt/sources.list` (regresa a 'El archivo /etc/apt/sources.list' en la página 3) sigue los pasos en 'Trabajando con paquetes' en la página 9.

Nota: la lista de los servidores se puede encontrar en el archivo:`http://www.debian.org/mirror/mirrors_full`.

Iniciando con la versión 0.3, el paquete netselect incluye el script **netselect-apt**, el cual realiza automáticamente el proceso mencionado arriba. Sólo introduzca el árbol de la distribución como parámetro (el cual, por defecto es "stable") y el archivo `sources.list` será generado con los mejores servidores para main y non-US y será guardado en el directorio actual. El siguiente ejemplo genera un archivo sources.list de una distribución estable

```
# ls sources.list
ls: sources.list: File or directory not found
# netselect-apt stable
(...)
# ls -l sources.list
sources.list
#
```

Recuerde: el archivo `sources.list` se crea en el directorio actual, y debe moverse al directorio `/etc/apt` .

Después, siga los pasos de 'Trabajando con paquetes' en la página 9.

2.4. Agregando un CD-ROM al archivo sources.list

Si prefiere utilizar el CD-ROM para instalar los paquetes o para actualizar su sistema con APT, lo puede agregar a su archivo `sources.list`. Para hacerlo, puede utilizar el programa `apt-cdrom` así:

```
# apt-cdrom add
```

con el CD-ROM de Debian en la unidad. Esta instrucción montará el CD-ROM, y si es un CD válido de Debian buscará la información de los paquetes en el CD. Si la configuración de su unidad de CD-ROM es inusual, tiene las siguientes opciones:

```
-h           - program help
-d directory - CD-ROM mount point
-r           - Rename a recognized CD-ROM
-m           - No mounting
-f           - Fast mode, don't check package files
-a           - Thorough scan mode
```

Por ejemplo:

```
# apt-cdrom -d /home/kov/mycdrom add
```

También puede identificar el CD-ROM sin agregarlo a su lista:

```
# apt-cdrom ident
```

Note que este programa sólo funcionará si el CD-ROM está configurado adecuadamente en el archivo `/etc/fstab` de su sistema.

Capítulo 3

Trabajando con paquetes

3.1. Actualizando la lista de paquetes disponibles

El sistema de paquetes utiliza una base de datos para llevar un monitoreo de los paquetes instalados, los no instalados y cuales están disponibles para su futura instalación. El programa `apt-get` utiliza esta base de datos para averiguar como instalar los paquetes que son requeridos por el usuario y para indagar sobre que paquetes adicionales serán requeridos para que el paquete seleccionado funcione correctamente.

Para actualizar la lista, se utiliza el comando `apt-get update`. Este comando busca el paquete en los archivos listados en `/etc/apt/sources.list`; para más información acerca de este archivo, revise 'El archivo /etc/apt/sources.list' en la página 3

Es una buena costumbre ejecutar este archivo regularmente para mantenerse informado acerca de las posibilidades de actualización para el sistema, particularmente las actualizaciones de seguridad.

3.2. Instalando paquetes

Finalmente, ¡el proceso que estaba esperando!. Con su archivo sources.list listo y su lista de paquetes disponibles al día, todo lo que necesita es ejecutar `apt-get` para tener el paquete que quiera instalado. Por ejemplo, puede ejecutar:

```
# apt-get install xchat
```

APT buscará en su base de datos para encontrar la versión más reciente del paquete y lo descargará del servidor correspondiente especificado en `sources.list`. Si este paquete necesitara otro para funcionar – como en este caso – APT resolverá las dependencias e instalará los paquetes necesarios. Observe este ejemplo:

```
# apt-get install nautilus
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  bonobo libmedusa0 libnautilus0
The following NEW packages will be installed:
  bonobo libmedusa0 libnautilus0 nautilus
0 packages upgraded, 4 newly installed, 0 to remove and 1 not upgraded.
Need to get 8329kB of archives. After unpacking 17.2MB will be used.
Do you want to continue? [Y/n]
```

El paquete `nautilus` necesita las librerías compartidas mencionadas, así pues APT las descargará del servidor. Si se especifican antes los nombres de esas librerías con el comando `apt-get` APT no pregunta, si desea continuar o no; supone automáticamente que se desean instalar esos paquetes.

Esto significa que APT sólo pregunta por confirmación cuando se van a instalar paquetes que no fueron especificados en la línea de comando.

Las siguientes opciones de apt-get podrían ser útiles

```
-h  This help text.
-d  Download only - do NOT install or unpack archives
-f  Attempt to continue if the integrity check fails
-s  No-act. Perform ordering simulation
-y  Assume Yes to all queries and do not prompt
-u  Show a list of upgraded packages as well
```

Pueden seleccionarse varios paquetes para instalar en una sola línea. Los archivos descargados son almacenados en el directorio `/var/cache/apt/archives` para su instalación posterior.

Puede especificar también que paquetes serán eliminados en la misma línea. Sólo agregue un "-" inmediatamente después del nombre del paquete que quiere eliminar, por ejemplo:

```
# apt-get install nautilus gnome-panel-
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  bonobo libmedusa0 libnautilus0
The following packages will be REMOVED:
  gnome-applets gnome-panel gnome-panel-data gnome-session
The following NEW packages will be installed:
  bonobo libmedusa0 libnautilus0 nautilus
0 packages upgraded, 4 newly installed, 4 to remove and 1 not upgraded.
Need to get 8329kB of archives. After unpacking 2594kB will be used.
Do you want to continue? [Y/n]
```

Consulte la sección 'Eliminando paquetes' en esta página para más detalles acerca de la eliminación de paquetes.

Si de alguna forma daña un paquete instalado, o simplemente desea reinstalar la versión más nueva disponible del paquete, puede utilizar la opción `--reinstall` como se muestra:

```
# apt-get --reinstall install gdm
Reading Package Lists... Done
Building Dependency Tree... Done
0 packages upgraded, 0 newly installed, 1 reinstalled, 0 to remove and 1 not
 upgraded.
Need to get 0B/182kB of archives. After unpacking 0B will be used.
Do you want to continue? [Y/n]
```

La versión de APT al crear este documento es la 0.5.3, la cual forma parte de la versión inestable de Debian (`sid`) al momento de escribir. Si tiene esta versión instalada, tiene algunas opciones más a su alcance: Puede utilizar un comando como `apt-get install paquete/distribución` para instalar paquetes de una distribución en específico, o `apt-get install package=versión`. por ejemplo:

```
# apt-get install nautilus/unstable
```

Esta instrucción instalará nautilus de la distribución inestable aun si está utilizando la estable. Los valores aceptados para distribución son `stable`, `testing` y `unstable`.

Podría preferir utilizar el modificador `-t` para especificar una distribución destino, dejando a `apt-get` la oportunidad de resolver a favor de esa distribución las dependencias.

IMPORTANTE: La versión "unstable" de Debian es la versión a la cual se le agregan las nuevas versiones de los paquetes de Debian. Esta distribución tiene todos los cambios por los cuales atraviesan muchos de los paquetes, ya sean cambios pequeños o grandes que afecten a unos paquetes o a todo el sistema. Por esta razón, esta versión de la distribución *no* deberá ser usada por usuarios novatos o por aquellos que necesitan de estabilidad.

La distribución "testing" (de prueba) es un poco mejor que la inestable, con algo de estabilidad, pero para sistemas en producción la que se debe usar es la estable.

3.3. Eliminando paquetes

Si ya no necesita utilizar cierto paquete, puede eliminarlo de su sistema utilizando APT. Para realizar esta tarea sólo escriba: `apt-get remove paquete`. por ejemplo:

```
# apt-get remove gnome-panel
Reading Package Lists... Done
Building Dependency Tree... Done
```

```
The following packages will be REMOVED:
  gnome-applets gnome-panel gnome-panel-data gnome-session
0 packages upgraded, 0 newly installed, 4 to remove and 1 not upgraded.
Need to get 0B of archives. After unpacking 14.6MB will be freed.
Do you want to continue? [Y/n]
```

Como se puede apreciar en el ejemplo anterior, APT se hace cargo de eliminar los paquetes dependientes del paquete eliminado. No hay manera de eliminar un paquete utilizando APT sin eliminar los paquetes que éste necesitaba.

Ejecutando `apt-get` como en el ejemplo causará que los paquetes sean eliminados, pero sus archivos de configuración, si existían, permanecerán intactos en el sistema. Para una eliminación completa del paquete, ejecute:

```
# apt-get --purge remove gnome-panel
Reading Package Lists... Done
Building Dependency Tree... Done
The following packages will be REMOVED:
  gnome-applets* gnome-panel* gnome-panel-data* gnome-session*
0 packages upgraded, 0 newly installed, 4 to remove and 1 not upgraded.
Need to get 0B of archives. After unpacking 14.6MB will be freed.
Do you want to continue? [Y/n]
```

Observe el "*" después de los nombres. Esto indica que los archivos de configuración de cada paquete serán eliminados también.

Al igual que en el caso de la `instalación`, puede utilizar un símbolo con la opción de "`remove`" para invertir el significado de un paquete en especial. En el caso de la eliminación, si agrega un "+" después del nombre del paquete, el paquete será instalado en vez de eliminarlo.

```
# apt-get --purge remove gnome-panel nautilus+
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  bonobo libmedusa0 libnautilus0 nautilus
The following packages will be REMOVED:
  gnome-applets* gnome-panel* gnome-panel-data* gnome-session*
The following NEW packages will be installed:
  bonobo libmedusa0 libnautilus0 nautilus
0 packages upgraded, 4 newly installed, 4 to remove and 1 not upgraded.
Need to get 8329kB of archives. After unpacking 2594kB will be used.
Do you want to continue? [Y/n]
```

Observe que `apt-get` enlista los paquetes que serán instalados aparte de los seleccionados (esto es, aquellos que su instalación es necesaria para el funcionamiento de aquellos que son seleccionados), aquellos que serán eliminados, y aquellos que serán instalados (incluyendo los paquetes adicionales otra vez).

3.4. Actualizando paquetes

Las actualizaciones de los paquetes son un gran éxito de APT. Pueden realizarse con tan sólo un comando: `apt-get upgrade`. Puede utilizar esa opción para actualizar los paquetes de la distribución actual, o bien para actualizar a una nueva distribución, aunque el comando `apt-get dist-upgrade` es una mejor opción; para mayor información, consulte la sección 'Actualizando a una nueva versión' en esta página.

Es muy útil utilizar este comando con la opción `-u`. Esta opción muestra la lista completa de paquetes que APT actualizará. Sin ella, se estaría actualizando a ciegas. APT descargará las versiones más recientes de cada paquete y las instalará de la manera más apropiada. Es muy importante ejecutar siempre `apt-get update` antes de probar esto. Consulte la sección 'Actualizando la lista de paquetes disponibles' en la página 9. Observe este ejemplo:

```
# apt-get -u upgrade
Reading Package Lists... Done
Building Dependency Tree... Done
The following packages have been kept back
  cpp gcc lilo
The following packages will be upgraded
  adduser ae apt autoconf debhelper dpkg-dev esound esound-common ftp indent
  ipchains isapnptools libaudiofile-dev libaudiofile0 libesd0 libesd0-dev
  libgtk1.2 libgtk1.2-dev liblockfile1 libnewt0 liborbit-dev liborbit0
  libstdc++2.10-glibc2.2 libtiff3g libtiff3g-dev modconf orbit procps psmisc
29 packages upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
Need to get 5055B/5055kB of archives. After unpacking 1161kB will be used.
Do you want to continue? [Y/n]
```

El proceso es muy fácil. Note que en las primeras líneas `apt-get` menciona que algunos paquetes fueron `conservados`. Esto significa que hay versiones nuevas de estos paquetes mas no fueron actualizados por alguna razón. Algunas razones pueden ser dependencias fallidas (el paquete del cual depende no tiene una versión nueva para actualizar) o nuevas dependencias (el paquete ahora depende de nuevos paquetes que la versión anterior).

No hay una solución clara para el primer caso. Para el segundo, es suficiente con ejecutar `apt-get install` para el paquete en cuestión, ya que con esto se descargarán las nuevas dependencias. Una solución aún mejor es utilizar `dist-upgrade`. Consulte la sección 'Actualizando a una nueva versión' en esta página.

3.5. Actualizando a una nueva versión

Esta opción de APT permite actualizar un sistema debian en un solo paso, ya sea desde Internet o por CDs (comprado o descargado como una imagen ISO).

También es utilizado cuando son realizados cambios entre las relaciones de los paquetes instalados. Con `apt-get upgrade`, estos paquetes permanecerían sin modificación (`conservados`).

Por ejemplo, supongamos que está utilizando la revisión 0 de la versión estable de Debian, y compra el CD con la revisión 3. Puede utilizar APT para actualizar el sistema al de el CD. Para lograr esto, utilice `apt-cdrom` (consulte la sección 'Agregando un CD-ROM al archivo sources.list' en la página 6) para agregar el CD al archivo `/etc/apt/sources.list` y ejecute `apt-get dist-upgrade`

Es muy importante mencionar que APT siempre busca la versión más reciente de los paquetes. Así pues, si en su archivo `/etc/apt/sources.list` se encontrara alguna otra fuente que tuviera una versión más reciente que la del CD, APT descargaría esta versión.

En el ejemplo mostrado en la sección 'Actualizando paquetes' en la página anterior, observamos que algunos paquetes eran `conservados`. Solucionaremos este problema ahora con el programa `dist-upgrade`:

```
# apt-get -u dist-upgrade
Reading Package Lists... Done
Building Dependency Tree... Done
Calculating Upgrade... Done
The following NEW packages will be installed:
  cpp-2.95 cron exim gcc-2.95 libident libopenldap-runtime libopenldap1
  libpcre2 logrotate mailx
The following packages have been kept back
  lilo
The following packages will be upgraded
  adduser ae apt autoconf cpp debhelper dpkg-dev esound esound-common ftp gcc
  indent ipchains isapnptools libaudiofile-dev libaudiofile0 libesd0
  libesd0-dev libgtk1.2 libgtk1.2-dev liblockfile1 libnewt0 liborbit-dev
  liborbit0 libstdc++2.10-glibc2.2 libtiff3g libtiff3g-dev modconf orbit
  procps psmisc
31 packages upgraded, 10 newly installed, 0 to remove and 1 not upgraded.
Need to get 0B/7098kB of archives. After unpacking 3118kB will be used.
Do you want to continue? [Y/n]
```

Observe que los paquetes ahora serán actualizados, y que nuevos paquetes serán instalados (las nuevas dependencias de los paquetes). Observe también que lilo sigue siendo `conservado`. Probablemente tiene un problema aún más serio que una dependencia. Esto lo podemos saber si ejecutamos:

```
# apt-get -u install lilo
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
```

```
  cron debconf exim libident libopenldap-runtime libopenldap1 libpcre2
  logrotate mailx
The following packages will be REMOVED:
  debconf-tiny
The following NEW packages will be installed:
  cron debconf exim libident libopenldap-runtime libopenldap1 libpcre2
  logrotate mailx
The following packages will be upgraded
  lilo
1 packages upgraded, 9 newly installed, 1 to remove and 31 not upgraded.
Need to get 225kB/1179kB of archives. After unpacking 2659kB will be used.
Do you want to continue? [Y/n]
```

Como se observa arriba, lilo tiene un nuevo conflicto con el paquete `debconf-tiny`, lo cual significa que no podrá ser instalado (o actualizado) sin antes eliminar debconf-tiny

Para saber que guarda o elimina un paquete puede utilizar:

```
# apt-get -o Debug::pkgProblemResolver=yes dist-upgrade
Reading Package Lists... Done
Building Dependency Tree... Done
Calculating Upgrade... Starting
Starting 2
Investigating python1.5
Package python1.5 has broken dep on python1.5-base
  Considering python1.5-base 0 as a solution to python1.5 0
  Holding Back python1.5 rather than change python1.5-base
Investigating python1.5-dev
Package python1.5-dev has broken dep on python1.5
  Considering python1.5 0 as a solution to python1.5-dev 0
  Holding Back python1.5-dev rather than change python1.5
 Try to Re-Instate python1.5-dev
Done
Done
The following packages have been kept back
  gs python1.5-dev
0 packages upgraded, 0 newly installed, 0 to remove and 2  not upgraded.
```

De este modo, es fácil notar que el paquete python1.5-dev no puede ser instalado debido a una dependencia: python1.5.

3.6. Eliminando archivos de paquete no utilizados: apt-get clean y autoclean.

Cuando APT instala un paquete, descarga los archivos necesarios de los servidores enlistados en /etc/apt/sources, estos a su vez son guardados en un depósito local (`/var/cache/apt/archives/`), y de ahí se procede con la instalación, consulte 'Instalando paquetes' en la página 9.

Con el tiempo el depósito puede crecer y ocupar mucho espacio en disco. Afortunadamente, APT provee de herramientas para manejar su depósito local: `apt-get`, `clean` y `autoclean`.

`apt-get clean` elimina todo excepto los archivos "lock" de `/var/cache/apt/archives/` y `/var/cache/apt/archives/partial/`. Así, si necesita reinstalar un paquete APT, lo descargará de nueva cuenta.

`apt-get autoclean` elimina sólo los archivos que no pueden ser descargados de nuevo.

EL siguiente ejemplo muestra como funciona la autolimpieza de apt-get:

```
# ls /var/cache/apt/archives/logrotate* /var/cache/apt/archives/gpm*
logrotate_3.5.9-7_i386.deb
logrotate_3.5.9-8_i386.deb
gpm_1.19.6-11_i386.deb
```

En /var/cache/apt/archives hay dos archivos para el paquete `logrotate`y uno para `gpm`.

```
# apt-show-versions -p logrotate
logrotate/stable uptodate 3.5.9-8
# apt-show-versions -p gpm
gpm/stable upgradeable from 1.19.6-11 to 1.19.6-12
```

`apt-show-versions` muestra que `logrotate_3.5.9-8_i386.deb` provee la versión actualizada de of `logrotate`, así pues `logrotate_3.5.9-7_i386.deb` es innecesario. También `gpm_1.19.6-11_i386.deb` no es necesario debido a que una versión mas reciente puede ser descargada.

```
# apt-get autoclean
Reading Package Lists... Done
Building Dependency Tree... Done
Del gpm 1.19.6-11 [145kB]
Del logrotate 3.5.9-7 [26.5kB]
```

Finalmente, `apt-get autoclean` elimina los archivos viejos. Consulte 'Cómo actualizar paquetes de versiónes específicas de Debian.' en la página 18 para más información sobre apt-show-versions.

3.7. Utilizando APT con dselect

`dselect` es un programa que ayuda a los usuarios a seleccionar paquetes de Debian para su instalación. Es considerado como algo complicado y un poco aburrido, pero con practica podrás utilizar su interfaz basada en consola-ncurses.

dselect se caracteriza por ser un programa que sabe como explotar la capacidad de los paquetes Debian para "recomendar" y "sugerir" otros paquetes para su instalación. Para utilizar el programa, ejecuta `'dselect'` como root. Selecciona 'apt' como el método de acceso. Esto no es verdaderamente necesario, pero si no está utilizando un CD ROM y quiere descargar los paquetes de Internet, la mejor opción es utilizar dselect.

Para entender mejor el uso de dselect, lea la documentación que se encuentra en la página de Debian http://www.debian.org/doc/ddp

Después de seleccionar los paquetes con dselect, utilice:

```
# apt-get -u dselect-upgrade
```

Como en el siguiente ejemplo:

```
# apt-get -u dselect-upgrade
Reading Package Lists... Done
Building Dependency Tree... Done
The following packages will be REMOVED:
  lbxproxy
The following NEW packages will be installed:
  bonobo console-tools-libs cpp-3.0 enscript expat fingerd gcc-3.0
  gcc-3.0-base icepref klogd libdigest-md5-perl libfnlib0 libft-perl
  libgc5-dev libgcc300 libhtml-clean-perl libltdl0-dev libsasl-modules
  libstdc++3.0 metamail nethack proftpd-doc psfontmgr python-newt talk tidy
  util-linux-locales vacation xbill xplanet-images
The following packages will be upgraded
  debian-policy
1 packages upgraded, 30 newly installed, 1 to remove and 0 not upgraded.
Need to get 7140kB of archives. After unpacking 16.3MB will be used.
Do you want to continue? [Y/n]
```

Compárelo con lo que pasa al ejecutar apt-get dist-upgrade en el mismo sistema:

```
# apt-get -u dist-upgrade
Reading Package Lists... Done
Building Dependency Tree... Done
Calculating Upgrade... Done
The following packages will be upgraded
```

```
  debian-policy
1 packages upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 421kB of archives. After unpacking 25.6kB will be freed.
Do you want to continue? [Y/n]
```

Observe que muchos paquetes de arriba están siendo instalados porque otros paquetes los sugirieron o recomendaron. Otros están siendo instalados o eliminados (en el caso de lbxproxy, por ejemplo) por las selecciones que realizamos por medio de la lista de dselect. Dselect puede ser una herramienta poderosa al usarse en conjunción con APT.

3.8. Cómo mantener un sistema Híbrido.

La gente frecuentemente utiliza la distribución de prueba, ya que es más estable que la inestable y más al día que la estable. Sin embargo, los usuarios que quieren la última versión de algunos paquetes mas sin embargo no confian en cambiar su sistema completo, tienen la opción de utilizar una mezcla de un sistema prueba/inestable. Por otro lado, los mas conservadores querrán utilizar una mezcla de estable/prueba.

Para hacer esto, agrega la siguiente línea en `/etc/apt/apt.conf`:

```
APT::Default-Release "testing";
```

Después, cuando esté listo a instalar paquetes de inestable, sólo utilice el modificador `-t`:

```
# apt-get -t unstable install packagename
```

No olvide que para utilizar paquetes de una versión de Debian, necesita agregar una línea de apt source a la lista en `/etc/apt/sources`. En el ejemplo, necesitamos líneas de `unstable`en vez de `testing`.

3.9. Cómo actualizar paquetes de versiónes específicas de Debian.

`apt-show-versions` provee una amnera segura de actualizar los sistemas híbridos sin tener que instalar más de la distribución menos estable de la que tenían en mente. Por ejemplo, es posible actualizar solo los paquetes inestables ejecutando:

```
# apt-get install `apt-show-versions -u -b | grep unstable`
```

3.10. Cómo mantener versiones específicas de paquetes instalados

Usted puede modificarle algo a un programa, y no tener tiempo o no querer portar los cambios hacia una nueva versión del mismo. O tal vez haya actualizado su sistema a la versión 3.0 de Debian, pero quiere continuar con cierta versión de un programa de Debian 2.2. Puede marcar la versión que tiene instalada así ésta no será actualizada.

Utilizar este recurso es simple. Sólo se necesita editar el archivo `/etc/apt/preferences`.

El formato es simple:

```
Package: <package>
Pin: <pin definition>
Pin-Priority: <pin's priority>
```

Por ejemplo, para mantener el paquete `sylpheed` el cual he modificado para utilizar "reply-to-kist" en la versión 0.4.99, agrego:

```
Package: sylpheed
Pin: version 0.4.99*
```

Observe que utilizo un `*` (asterisco). Este es un "comodín"; este específica que yo quiero que este "pin" sea válido para todas las versiones que empiecen con 0.4.99. Esto es así porque Debian numera sus paquetes con una "revisión Debian" y yo no quiero entorpecer la instalación de esas revisiones. Así pues, las versiones 0.4.99-1 y 0.4.99-10 serán instaladas tan pronto como estén disponibles. Observe que si modificó un programa no querrá hacer esto.

El campo `Pin-Priority` es opcional; si no se especifica algo, el valor por omisión es 989

Veamos como trabajan las prioridades de "pin". Una prioridad menor que 0 indica que el paquete jamás deberá ser instalado. Prioridades de 0 a 100 denotan paquetes que no están instalados y que no hay versiones disponibles. Esto no vendrá en el proceso de escoger-versión. La prioridad de 100 es asignada a los paquetes instalados - para la versión instalada de un paquete que será reemplazado por una versión diferente, el reemplazo deberá tener una prioridad mayor a 100.

Prioridades mayores a 100 indican que el paquete debe ser instalado. Típicamente, la versión instalada del paquete cambia sólo con las actualizaciones a una versión más nueva. Cualquier prioridad entre 100 y 1000 (incluyéndolos) indica este comportamiento típico. Un paquete con tal prioridad no se reemplazará por una versión anterior. Por lo tanto, si yo tuviera sylpheed 0.5.3 instalado y definiera un pin en sylpheed 0.4.99 con prioridad 999, el paquete 0.4.99 este `no` se instalaría para satisfacer el pin. Para hacer un programa que pueda reemplazarse por una versión anterior, necesita poseer una prioridad mayor a 1000.

Un "pin" puede ser especificado en una `versión`, `sub-versión` o al `origen` de un paquete.

Estableciendo un "pin" en una `versión`, como hemos visto, pueden ser especificadas utilizando números y comodines, varias versiones al mismo tiempo.

La opción "`release`" depende de el archivo "Release" de un servidor o CD. Esta opción podría no ser de mucha utilidad si está utilizando servidores que no ofrezcan este archivo. Puede ver los contenidos de este archivo que tiene en `/var/lib/apt/lists/`. Los parámetros para un "release" son: `a` (archive), `c` (components), `v` (version), `o` (origin) and `l` (label).

Un ejemplo:

```
Package: *
Pin: release v=2.2*,a=stable,c=main,o=Debian,l=Debian
Pin-Priority: 1001
```

En este ejemplo, escogimos la versión 2.2* de Debian (la cual puede ser 2.2r2, 2.2r3 – esto es un "point release" que típicamente incluyen parches de seguridad y otras actualizaciones importantes), el servidor `estable`, sección `principal` etiqueta y origen Debian. Origen(o=) define quien creó el archivo "Release", la etiqueta (l=) define el nombre d la distribución: Debian para Debian como si mismo y "Progeny" para "Progeny", por ejemplo:

```
$ cat
 /var/lib/apt/lists/ftp.debian.org.br_debian_dists_potato_main_binary-i386_Re
se
Archive: stable
version: 2.2r3
Component: main
Origin: Debian
Label: Debian
Architecture: i386
```

Capítulo 4

Ayudantes muy útiles.

4.1. Cómo instalar paquetes localmente compilados: equivs

Algunas veces, la gente quiere utlilizar una versión específica de un programa disponible sólo en código fuente, sin un paquete Debian. Para el sistema de empaquetamiento puede ser problemático el hacer esto. Suponga que quiere compilar una nueva versión de un servidor de correo. Todo esta bien, pero varios paquetes de Debian dependen de un MTA (Mail Transport Agent). Desde que instala algo que compila, el sistema de adminstración de paquetes no sabe nada al respecto.

Aquí es donde `equivs` entra a la escena. Para utilzarlo, instale el paquete con ese nombre. Lo que hace es crear un paquete vacio que posibilita el satisfacer las dependencias, haciendo que el sistema de administración de paquetes crea que las dependencias han sido satisfechas.

Antes de iniciar, es bueno recordarle que hay maneras más seguras de compilar un programa que esta ya empaquetedo para Debian con diferentes opciones, y que no se debería utilizar equivs para reemplazar dependencias si no sabe lo que hace. Consulte 'Trabajando con los paquetes fuente' en la página 31 para mas informacion.

Continuemos con el ejemplo del MTA, usted acaba de instalar su `postfix` recién compilado y ahora instalará `mutt`. De repente descubre que `mutt` quiere instalar otro MTA. Pero usted ya tiene el suyo.

Vaya a algún directorio (`/tmp`, por ejemplo) y ejecute:

```
# equivs-control name
```

Sustituya *name* por el nombre del archivo de control que quiera crear. El archivo será creado como se muestra:

```
Section: misc
Priority: optional
Standards-Version: 3.0.1
```

```
Package: <enter package name; defaults to equivs-dummy>
Version: <enter version here; defaults to 1.0>
Maintainer: <your name and email address; defaults to username>
Pre-Depends: <packages>
Depends: <packages>
Recommends: <packages>
Suggests: <package>
Provides: <(virtual)package>
Architecture: all
Copyright: <copyright file; defaults to GPL2>
Changelog: <changelog file; defaults to a generic changelog>
Readme: <README.Debian file; defaults to a generic one>
Extra-Files: <additional files for the doc directory, comma separated>
Description: <short description; defaults to some wise words>
 long description and info
 .
 segundo párrafo
```

Nosotros sólo necesitamos modificar esto para hacer lo que queremos. Eche un vistazo al formato del campo y a sus descripciones, no hay necesidad de explicar cada una, vamos a hacer lo que se necesita:

```
Section: misc
Priority: optional
Standards-Version: 3.0.1

Package: mta-local
Provides: mail-transport-agent
```

Si, eso es todo. `mutt` depende de `mail-transport-agent`, este es un paquete virtual provisto por todos los MTA, yo podría simplemente nombrar el paquete `mail-transport-agent`, pero prefiero usar el esquema de paquetes virtuales, utilizando “Provides”.

Ahora sólo necesita crear el paquete:

```
# equivs-build name

dh_testdir
touch build-stamp
dh_testdir
dh_testroot
dh_clean -k
# Add here commands to install the package into debian/tmp.
touch install-stamp
```

```
dh_testdir
dh_testroot
dh_installdocs
dh_installchangelogs
dh_compress
dh_fixperms
dh_installdeb
dh_gencontrol
dh_md5sums
dh_builddeb
dpkg-deb: building package 'name' in '../name_1.0_all.deb'.

The package has been created.
Attention, the package has been created in the current directory,
```

E instalar el `.deb` creado.

Como se puede ver, hay muchos usos para `equivs`. Se puede crear un paquete `my-favorites` el cual dependiera en los paquetes que usted usualmente instala, por ejemplo. Sólo deje volar su imaginación, pero sea cuidadoso.

Es importante recalcar que hay archivos de control ejemplo en `/usr/share/doc/equivs/examples`. Revíselos.

4.2. Eliminando archivos de localización no utilizados: localepurge

Muchos Debianeros utilizan sólo un "locale". Un usuario Mexicano de Debian, por ejemplo, usualmente utulizará el `es_MX` todo el tiempo y no le importará el `de`.

`localepurge` es una herramienta muy útil para estos usuarios. Puede liberar mucho espacio dejando sólo instalados los que realmente necesita. Sólo ejecuta `apt-get install localepurge`.

Es muy fácil de configurar, las preguntas de debconf guían al usuario en una configuración paso a paso. Sea cuidadoso en contestar la primer pregunta, una respuesta errónea podría eliminar todos los archivos de localización, aún aquellos que necesitas. La única manera de recuperarlos es reinstalando todos lo paquetes que los proveen.

4.3. Cómo saber que paquetes pueden ser actualizados.

`apt-show-versions` es un programa que muestra que paquetes en el sistema pueden ser actualizados y mucha información mas. La opción `-u` muestra una lista de paquetes actualizables:

```
$ apt-show-versions -u
libeel0/unstable upgradeable from 1.0.2-5 to 1.0.2-7
libeel-data/unstable upgradeable from 1.0.2-5 to 1.0.2-7
```

Capítulo 5

Obteniendo información de los paquetes.

Existen algunas interfaces para el APT que lo hacen más fácil de utilizar.

Pero nuestro objetivo aquí es aprender a manejar APT puro. Así que, ¿cómo podría saber el nombre de un paquete que quiere instalar?

Tenemos numerosos recursos para realizar esa tarea. Empezaremos con `apt-cache`. Este programa es utilizado por APT para mantener su base de datos. Nosotros sólo veremos un poco de sus aplicaciones.

5.1. Descubriendo nombres de paquetes

Por ejemplo, supongamos que usted quiere revivir la gloria de la época dorada del Atari 2600. Quiere utilizar APT para instalar un emulador de Atari, y después bajar algunos juegos, puede hacer lo siguiente:

```
# apt-cache search atari
atari-fdisk-cross - Partition editor for Atari (running on non-Atari)
circuslinux - The clowns are trying to pop balloons to score points!
madbomber - A Kaboom! clone
tcs - Character set translator.
atari800 - Atari emulator for svgalib/X/curses
stella - Atari 2600 Emulator for X windows
xmess-x - X binaries for Multi-Emulator Super System
```

Hemos encontrado muchos paquetes relacionados con lo que estamos buscando. Para obtener mayor información de un paquete específico, hacemos lo siguiente:

```
# apt-cache show stella
```

```
Package: stella
Priority: extra
Section: non-free/otherosfs
Installed-Size: 830
Maintainer: Tom Lear <tom@trap.mtview.ca.us>
Architecture: i386
version: 1.1-2
Depends: libc6 (>= 2.1), libstdc++2.10, xlib6g (>= 3.3.5-1)
Filename: dists/potato/non-free/binary-i386/otherosfs/stella_1.1-2.deb
Size: 483430
MD5sum: 11b3e86a41a60fa1c4b334dd96c1d4b5
Description: Atari 2600 Emulator for X windows
 Stella is a portable emulator of the old Atari 2600 video-game console
 written in C++. You can play most Atari 2600 games with it. The latest
 news, code and binaries for Stella can be found at:
 http://www4.ncsu.edu/~bwmott/2600
```

En este desplegado tiene muchos detalles sobre paquete seleccionado así como su descripcion. Si el paquete estuviera instalado y hubiera una versión más reciente, vería la información de las dos versiones, por ejemplo:

```
# apt-cache show lilo
Package: lilo
Priority: important
Section: base
Installed-Size: 271
Maintainer: Russell Coker <russell@coker.com.au>
Architecture: i386
version: 1:21.7-3
Depends: libc6 (>= 2.2.1-2), debconf (>=0.2.26), logrotate
Suggests: lilo-doc
Conflicts: manpages (<<1.29-3)
Filename: pool/main/l/lilo/lilo_21.7-3_i386.deb
Size: 143052
MD5sum: 63fe29b5317fe34ed8ec3ae955f8270e
Description: LInux LOader - The Classic OS loader can load Linux and others
 This Package contains lilo (the installer) and boot-record-images to
 install Linux, OS/2, DOS and generic Boot Sectors of other OSes.
 .
 You can use Lilo to manage your Master Boot Record (with a simple text scree
 or call Lilo from other Boot-Loaders to jump-start the Linux kernel.

Package: lilo
Status: install ok installed
Priority: important
```

```
Section: base
Installed-Size: 190
Maintainer: Vincent Renardias <vincent@debian.org>
version: 1:21.4.3-2
Depends: libc6 (>= 2.1.2)
Recommends: mbr
Suggests: lilo-doc
Description: LInux LOader - The Classic OS loader can load Linux and others
 This Package contains lilo (the installer) and boot-record-images to
 install Linux, OS/2, DOS and generic Boot Sectors of other OSes.
 .
 You can use Lilo to manage your Master Boot Record (with a simple text scree
 or call Lilo from other Boot-Loaders to jump-start the Linux kernel.
```

Observe que el primero en la lista es el paquete disponible y el segundo es el instalado. Para mayor información sobre un paquete puede utilizar:

```
# apt-cache showpkg penguin-command
Package: penguin-command
versions:
1.4.5-1(/var/lib/apt/lists/download.sourceforge.net_debian_dists_unstable_mai
inary-i386_Packages)(/var/lib/dpkg/status)

Reverse Depends:
Dependencies:
1.4.5-1 - libc6 (2 2.2.1-2) libpng2 (0 (null)) libsdl-mixer1.1 (2 1.1.0)
 libsdl1.1 (0 (null)) zlib1g (2 1:1.1.3)
Provides:
1.4.5-1 -
Reverse Provides:
```

Y sólo para saber de que paquetes depende:

```
[root]@[/] # apt-cache depends penguin-command
penguin-command
  Depends: libc6
  Depends: libpng2
  Depends: libsdl-mixer1.1
  Depends: libsdl1.1
  Depends: zlib1g
```

En resumen, tenemos una gran variedad de armas que podemos utilizar para averiguar el nombre del paquete que queramos.

5.2. Utilizando dpkg para conocer los nombres de paquetes

Uno de los caminos para conocer el nombre de un paquete es saber el nombre de un archivo importante que sea contenido por ese paquete. Por ejemplo, para conocer el paquete que ofrece ciertos archivos "`.h`" puede ejecutar:

```
# dpkg -S stdio.h
libc6-dev: /usr/include/stdio.h
libc6-dev: /usr/include/bits/stdio.h
perl: /usr/lib/perl/5.6.0/CORE/nostdio.h
```

o:

```
# dpkg -S /usr/include/stdio.h
libc6-dev: /usr/include/stdio.h
```

Para averiguar los nombres de paquetes instalados en su sistema, lo cual es útil por ejemplo, si planea limpiar su disco duro, ejecuta:

```
# dpkg -l | grep mozilla
ii  mozilla-browse 0.9.6-7        Mozilla Web Browser
```

El problema con este comando radica en que puede "truncar" el nombre del programa. En el ejemplo anterior, el nombre completo del paquete es `mozilla-browser`. Para arreglar esto, puede utilizar las variables de ambiente `COLUMNS` de este modo:

```
[kov]@[couve] $ COLUMNS=132 dpkg -l | grep mozilla
ii  mozilla-browser            0.9.6-7                    Mozilla Web Brows
```

O la descripción del paquete, o parte de ella para encontrar el nombre completo. Como se muestra a continuación:

```
# apt-cache search "Mozilla Web Browser"
mozilla-browser - Mozilla Web Browser
```

5.3. Cómo instalar programas "en demanda"

Está compilando un programa, y de repente, boom!, hay un error porque necesita cierto archivo "`.h`" que no tiene. El programa `auto-apt` puede salvarlo de esas situaciones. Le pregunta si instala ciertos programas que son requeridos, deteniendo los procesos relevantes y continuando una vez que el paquete es instalado.

Lo que hace, es básicamente ejecutar:

```
# auto-apt run command
```

Dónde "command" es el comando que al ser ejecutado, necesita ciertos archivos, por ejemplo:

```
# auto-apt run ./configure
```

Le preguntará entonces si desea instalar los paquetes necesarios y llamará automáticamente a apt-get. Si está en modo gráfico, una interfáz gráfica remplazará la interfáz de texto que aparece normalmente.

Auto-apt mantiene las bases de datos que necesitan estar al día para lograr que esto funcione. Esto se realiza con los comandos `auto-apt update`, `auto-apt updatedb` y `auto-apt update-local`.

5.4. Cómo descubrir a que paquete peretenece un archivo.

Si usted quiere instalar un paquete, y no puede saber como se llama al buscarlo con `apt-cache`, pero sabe el nombre de un archivo que pertenece al paquete, entonces puede utilizar `apt-file` para encontrar el nombre del paquete que busca. Esto se hace de la siguiente manera:

```
$ apt-file search filename
```

Trabaja de manera similar a `dpkg -S`, pero tambien le muestra paquetes sin instalar que contengan ese archivo. Puede también ser utilizado para saber que paquetes contiene archivos de inclusión para la compilación de ciertos programas, aunque `auto-apt` es una mejor solución para estos casos, consulte 'Cómo instalar programas "en demanda"' en la página anterior.

También puede enlistar el contenido de un paquete ejecutando:

```
$ apt-file list packagename
```

`apt-file` mantiene una base de datos sobre que archivos tiene cada paquete, tal como auto-apt lo hace y necesita estar al día. Esto se hace asi:

```
# apt-file update
```

Por omisión, `apt-file` utiliza la misma base de datos que `auto-apt` utiliza, consulte 'Cómo instalar programas "en demanda"' en la página anterior.

5.5. Cómo mantenerse informado de los cambios en los paquetes.

Cada paquete instala en su directorio de documentación (`/usr/share/doc/packagename`) un archivo llamado `changelog.Debian.gz` el cual contiene una lista de cambios realizados al paquete desde la última versión. Puede leer estos archivos con `zless`, por ejemplo, pero no es muy fácil estar buscando estos archivos después de una instalación de un sistema completo.

Existe una forma de automatizar esta tarea por medio de una herramienta llamada `apt-listchanges`. Para iniciar, primero necesita instalar el paquete `apt-listchanges`. Durante la instalación del paquete, Debconf lo configurará. Conteste a las preguntas sobre la configuración como desee.

La opción: "Apt-listchanges deberá ser automáticamente ejecutada por APT" esto es muy útil porque muestra los cambios que han sido hechos al paquete que está siendo instalado por apt durante una actualización y le deja analizarlos antes de continuar. La opción: "Debe apt-listchanges preguntar por autorización después de mostrar los cambios" es útil porque le pregunta si desea continuar la instalación después de leer los cambios. Si decide no continuar con la instalación apt-listchanges regresará con un error y apt abortará la instalación.

Después que apt-listchanges sea instalado, tan pronto como los paquetes sean descargados por apt, mostrará las listas de los cambios realizados a estos paquetes antes de instalarlos.

Capítulo 6

Trabajando con los paquetes fuente

6.1. Descargando los paquetes fuente

Es muy común en el mundo del software libre estudiar el código fuente o corregir código erróneo. Para lograr esto, necesita descargar el código fuente del programa. El sistema APT provee de una manera fácil de obtener código fuente de muchos programas contenidos en la distribución, incluyendo todos los archivos necesarios para crear un .deb para el programa.

Otro uso común de las fuentes de Debian es el de adaptar la versión más reciente de un programa, de la versión inestable. Compilar un programa en la versión estable generará .debs con las dependencias ajustadas para acoplarse a los paquetes en la distribución.

Para lograr esto una entrada `deb-src` en el archivo `/etc/apt/sources.list` debe apuntar hacia "unstable". Esta línea debe estar habilitada (sin comentar). Vea a la sección 'El archivo /etc/apt/sources.list' en la página 3.

Para descargar un paquete fuente, haga lo siguiente:

```
$ apt-get source packagename
```

Esto descargará tres archivos: un `.orig.tar.gz`, un `.dsc` y un `.diff.gz`. En el caso de paquetes especiales para Debian, el último de estos archivos no es descargado y el primero generalmente no tendrá el "`orig`" en el nombre.

El archivo `.dsc` es utilizado por dbpkg-source para descomprimir el paquete en el directorio *packagename-version*. Con cada paquete descargado existe un directorio `debian/` que contiene los archivos necesarios para crear un paquete .deb.

Para automáticamente compilar el paquete cuando está siendo descargado, sólo agrega `-b` a la línea de comando así.

```
$ apt-get -b source packagename
```

Si decide no crear el archivo .deb mientras descarga el paquete, puede crearlo después ejecutando:

```
$ dpkg-buildpackage -rfakeroot -uc -b
```

en el directorio que se ha creado después de descargar el paquete. Para instalar el paquete que ha sido creado por los comandos anteriores, hay que usar el sistema de administración de paquetes directamente como aquí:

```
# dpkg -i archivo.deb
```

Hay una diferencia entre el `apt-get source` y las otras opciones. La opción `source` puede ser utilizada por usuarios normales, sin necesitar de poderes especiales de root. Los archivos son descargados a un directorio desde el cual el comando `apt-get source package` es ejecutado.

6.2. Paquetes necesarios para la compilación de un paquete fuente

Normalmente, librerías compartidas y ciertos encabezados son necesarios para lograr una compilación exitosa. Todos los paquetes fuente tienen un campo en sus archivos de control llamado "Build-Depends:" el cual indica que paquetes son necesarios adicionalmente para que el paquete sea compilado.

APT tiene un método sencillo para descargar estos paquetes. Sólo ejecuta `apt-get build-dep package`, donde "package" es el nombre del paquete que vas a compilar, por ejemplo:

```
# apt-get build-dep gmc
Reading Package Lists... Done
Building Dependency Tree... Done
The following NEW packages will be installed:
  comerr-dev e2fslibs-dev gdk-imlib-dev imlib-progs libgnome-dev libgnorba-de
  libgpmg1-dev
0 packages upgraded, 7 newly installed, 0 to remove and 1 not upgraded.
Need to get 1069kB of archives. After unpacking 3514kB will be used.
Do you want to continue? [Y/n]
```

Los paquetes que serán instalados son los paquetes necesarios para que `gmc` sea compilado correctamente. Es importante observar que este comando no busca el paquete fuente del programa a ser compilado. Necesitará entonces ejecutar `apt-get source` para obtenerlo.

Si quiere solamente saber cuales paquetes se requieren para crear un cierto paquete, hay una variante del comando `apt-cache show` (vease 'Obteniendo información de los paquetes.' en la página 25, que enseña, entre otra información, la línea `Build-Depends` que enlista estos paquetes.

```
# apt-cache showsrc
package
```

Capítulo 7

Cómo lidiar con los errores

7.1. Errores comunes

Los errores siempre existirán, muchos de ellos son provocados por usuarios que no ponen atención. Lo siguiente es una lista de algunos de los errores más comunes y cómo corregirlos.

Si recibe un mensaje que se parece al mostrado a continuación cuando trate de ejecutar `apt-get install package`

```
Reading Package Lists... Done
Building Dependency Tree... Done
W: Couldn't stat source package list 'http://people.debian.org unstable/
 Packages'
 (/var/state/apt/lists/people.debian.org_%7ekov_debian_unstable_Packages) - s
 (2 No such file or directory)
W: You may want to run apt-get update to correct these missing files
E: Couldn't find package penguineyes
```

Olvidó ejecutar `apt-get update` después del último cambio al archivo `/etc/apt/sources.list`

Si el error se ve como este:

```
E: Could not open lock file /var/lib/dpkg/lock - open (13 Permission denied)
E: Unable to lock the administration directory (/var/lib/dpkg/), are you root
```

Cuando ejecuta `apt-get` con cualquier opción diferente de `source` , es porque no tiene permisos de root, esto es, lo está ejecutando como usuario normal.

Hay un error similar al de arriba el cual pasa cuando usted ejecuta dos copias de `apt-get` al mismo tiempo, o aun cuando trata de ejecutar `apt-get` mientras un proceso `dpkg` está activo. La única opción que puede ser utilizada simultáneamente es `source`.

Si una instalación se termina abruptamente en la mitad del proceso, y averigua que ya no es posible instalar o eliminar el paquete, intente con estos dos comandos:

```
# apt-get -f install
# dpkg --configure -a
```

Después inténtalo nuevamente. Tal vez sea necesario ejecutar el segundo comando más de una vez. Ésta es una lección importante para aquellos aventureros que utilizan la versión "unstable".

Si recibe el error "E: Dynamic MMap ran out of room" al ejecutar `apt-get update`, agregue la siguiente línea a `/etc/apt/apt.conf`:

```
APT::Cache-Limit 10000000;
```

7.2. ¿Dónde puedo encontrar ayuda?

Si está lleno de dudas, consulte la extensa documentación disponible para el sistema de paquetes de Debian. `--help`, las páginas de manual (man-pages) que pueden ser una enorme ayuda , así como la documentación incluida en los directorios de `/usr/share/doc` tal como `/usr/share/doc/apt`.

Si esta documentación no quita sus miedos, pregunte en las listas de correo de Debian. Puede encontrar más información sobre listas específicas en el sitio de Debian: `http://www.debian.org`.

Recuerde que estas listas y recursos sólo deberán ser utilizados por usuarios de Debian, usuarios de otros sistemas encontrarán un mejor soporte de las comunidades de sus distribuciones.

Capítulo 8

Que distribuciónes soporten APT.

Estos son los nombres de algunas distribuciones que utilizan APT:

Debian GNU/Linux (http://www.debian.org) - Fue por esta distribución que APT fue desarrollado.

Conectiva (http://www.conectiva.com.br) - Esta fue la primer distribución en portar APT con rpm.

Mandrake (http://www.mandrake.com)

PLD (http://www.pld.org.pl)

Vine (http://www.vinelinux.org)

APT4RPM (http://apt4rpm.sf.net)

Alt Linux (http://www.altlinux.ru/)

Red Hat (http://www.redhat.com/)

Sun Solaris (http://www.sun.com/)

SuSE (http://www.suse.de/)

Yellow Dog Linux (http://www.yellowdoglinux.com/)

Capítulo 9

Créditos

Gracias a mis amigos del proyecto Debian-BR, y a Debian mismo, quienes están ayudándome constantemente y siempre me dan fuerza para continuar trabajando para el beneficio de la humanidad, así como el ayudarme en mi meta de salvar al mundo. :)

También agradezco a la enorme ayuda brindada por CIPSGA a nuestro proyecto y a todos los proyectos libres que dan nuevas ideas.

Y un agradecimiento especial a:

Yooseong Yang <yooseong@debian.org>

Michael Bramer <grisu@debian.org>

Bryan Stillwell <bryan@bokeoa.com>

Pawel Tecza <pawel.tecza@poczta.fm>

Hugo Mora <h.mora@melix.com.mx>

Luca Monducci <luca.mo@tiscali.it>

Tomohiro KUBOTA <kubota@debian.org>

Pablo Lorenzzoni <spectra@debian.org>

Steve Langasek <vorlon@netexpress.net>

Arnaldo Carvalho de Melo <acme@conectiva.com.br>

Erik Rossen <rossen@freesurf.ch>

Ross Boylan <RossBoylan@stanfordalumni.org>

Matt Kraai <kraai@debian.org>

Aaron M. Ucko <ucko@debian.org>

Jon Åslund <d98-jas@nada.kth.se>

Capítulo 10

Nuevas versiónes de este tutorial:

Este manual fue creado para el proyecto Debian-BR (http://www.debian-br.org),con el propósito de ayudar al uso diario de Debian.

Nuevas versiónes de este documento estarán disponibles en la página de proyectos de documentos de Debian, en: http://www.debian.org/doc/ddp.

Comentarios y críticas pueden ser enviadas directamente a mi por email en: <kov@debian.org>.