Administración de redes Linux

Instalación de paquetes software

Iria M. Estévez Ayres

uc3m | Universidad Carlos III de Madrid

Departamento de Ingeniería Telemática

Marzo 2025

Índice

Apartado 1

Paquetes software

Paquetes software

- Todas las distribuciones de Linux tienen un sistema de gestión de software basado en paquetes.
  - Algunos son muy sencillos como es el caso de `openwrt.org`
- Lo que importa es que estandarizan la forma de hacer disponibles nuevos programas en las máquinas de los usuarios llevando un inventario de los paquetes disponibles, instalados, etc.
- Llevan un control de versiones muy estricto que permite expresar **dependencias y conflictos entre paquetes**.

Herramientas de control de paquetes

- `apt`, `apt-get` y `dselect` son las herramientas de control de paquetes más importantes, pero hay más: `tasksel`, `apt-cache`, `dpkg`, `aptitude`.
- La documentación del sistema de paquetes de debian está en el paquete `apt-doc`
  `file:///usr/share/doc/apt-doc/guide.html/index.html`
- Se puede usar `dpkg` directamente. Tiene en cuenta las dependencias, pero en caso de problemas puede forzar las cosas y podemos acabar con el sistema de paquetes "roto". (para más info: `dpkg --help`)
- Habitualmente, es suficiente con utilizar *apt* (o *apt-get*)
  - Comandos más habituales: update, install, remove, upgrade, full-upgrade, autoremove

Ejemplo de paquete

- Vamos a ver un paquete sencillo: `findutils`.
- Creamos un directorio temporal y lo descargamos ahí:

```
$ apt download findutils
$ ls
findutils_4.9.0-4_amd64.deb
```

- Ahora vamos a extraer el paquete en crudo:

```
astt@as2019:~/temp$ PS1='$ '
$ dpkg-deb -R findutils_4.9.0-4_amd64.deb .
 ls -l
total 632
drwxr-xr-x 2 astt astt   4096 ene  8  2023 DEBIAN
-rw-r--r-- 1 astt astt 635500 ene  8  2023 findutils_4.9.0-4_amd64.deb
drwxr-xr-x 4 astt astt   4096 ene  8  2023 usr
```

El control del paquete (I)

- En el directorio DEBIAN se encuentra el control del paquete:

```
$ ls DEBIAN/
control  md5sums
$ cat DEBIAN/control
Package: findutils
Version: 4.9.0-4
Architecture: amd64
Essential: yes
Maintainer: Andreas Metzler <ametzler@debian.org>
Installed-Size: 1746
Pre-Depends: libc6 (>= 2.34), libselinux1 (>= 3.1~)
Breaks: binstats (<< 1.08-8.1), guilt (<< 0.36-0.2), libpython3.4-minimal (<< 3.4.4-2),
libpython3.5-minimal (<< 3.5.1-3), lsat (<< 0.9.7.1-2.1), mc (<< 3:4.8.11-1), switchconf (<< 0.0.9-2.1)
Section: utils
Priority: required
Multi-Arch: foreign
Homepage: https://savannah.gnu.org/projects/findutils/
Description: utilities for finding files--find, xargs
 GNU findutils provides utilities to find files meeting specified
 criteria and perform various actions on the files which are found.
 This package contains 'find' and 'xargs'; however, 'locate' has
 been split off into a separate package.
```

- Esta información (y el MD5sum y el SHA256) también aparece si hacéis

```
$ apt-cache show findutils
```

El control del paquete (II)

Las dependencias se expresan en el fichero de control

- **`Depends`**: lista de dependencias separadas por comas, expresando que se necesita no sólo un determinado paquete, sino también la versión que se necesita
  - `libc6 (>= 2.34)`: necesita el paquete libc con la versión mayor o igual a la 2.34
  - Se usan los operadores `<<`, `<=`, `=`, `>=`, `>>`. El operador `|` indica `or`.
- **`Pre-Depends`**: el paquete en cuestión debe ser instalado y configurado antes de proceder a instalar este.
- **`Recommends`** o **`Suggests`**: recomiendan la instalación (mejor si los instalas) o sugieren la instalación (no es necesario, a menos que sepas qué vas a hacer con el paquete).
- **`Conflicts`**: paquetes que NO deben instalarse en el mismo sistema.
- **`Breaks`**: indican una incompatibilidad transitoria que puede hacer que *rompa* el paquete, un cambio de versión por ejemplo.
  - `dpkg` no va a instalarlo, `apt` va a intentar resolverlo

El control del paquete (y III)

- El fichero `md5sums` nos da garantías de que no se han alterado los ficheros particulares contenidos en el paquete:

```
$ head -n 4 DEBIAN/md5sums
77c69f8faad395a584cc682952786dfc  usr/bin/find
ae9ab3b8498cea8fce2c31ee287dac51  usr/bin/xargs
f78e2d4189be58135a915698efe1cd7a  usr/share/doc-base/findutils.findutils
75ca1d03fcbb9d988ff479d6c9ca6349  usr/share/doc/findutils/NEWS.Debian.gz
```

- Cuando el paquete se instala, las sumas md5 se meten en la base de datos de `dpkg`, de forma que se puede comprobar si se ha manipulado algún fichero:

```
$ sudo rm -Rf /usr/share/locale/zh_CN/LC_MESSAGES/findutils.mo
$ dpkg --verify findutils
??5??????   /usr/share/locale/zh_CN/LC_MESSAGES/findutils.mo
```

Instalaciones más complejas

- La instalación básica de un paquete que realiza `dpkg` (directamente o invocado por `apt-get` o por `apt`) es:
  - comprobar las dependencias
  - guardar los metadatos y los md5sums de los ficheros en la base de datos
  - copiar los ficheros
- Algunos paquetes realizan instalaciones más complejas

```
$ apt-get download unattended-upgrades
$ dpkg-deb -R unattended-upgrades_2.9.1+nmu3_all.deb .
$ ls DEBIAN/
conffiles  config  control  md5sums  postinst  postrm  preinst  templates
```

- Para eso utilizan debconf(7).

Apartado 2

Mantenimientos y actualizaciones

Sintaxis del fichero /etc/apt/sources.list (I)

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

- Tipo de origen
  - deb repositorio de paquetes binarios.
  - deb-src repositorio de paquetes fuente.
- URL de la fuente
  - Puede cambiarse el mirror seleccionando otro de la lista oficial https://www.debian.org/mirror/list
- Distribución/Suite
  - La distribución que usamos es bookworm.
  - Si se usara suite: oldstable, stable, testing, unstable.
  - Recomendación: usar la distribución. Evita "roturas" en el sistema.
- Componentes
  - main, non-free, contrib o non-free-firmware.

Sintaxis del fichero /etc/apt/sources.list (II)

Componentes

- main: paquetes que cumplen con las directrices para el software libre de Debian (DSFG). https://www.debian.org/social_contract#guidelines
- non-free: paquetes que no siguen las DFSG, pero que pueden ser distribuidos sin restricciones. **No forman parte de Debian.**
- contrib: paquetes que siguen las DFSG, pero que no pueden funcionar sin algunos elementos que no son software libre (que estarán en non-free) o incluso con elementos que no pertenecen a Debian en absoluto.
  - Ej: Openoffice necesitaba un entorno Java (propietario) para ejecutarse.
- El proyecto Debian tomó la decisión en 2022-10 de crear un nuevo repositorio, non-free-firmware, que se va a usar a partir de la actual distribución estable (bookworm)

Sintaxis del fichero /etc/apt/sources.list (y III)

Ejemplo de fichero sources.list para distribución bullseye

```
deb http://deb.debian.org/debian/ bookworm main non-free-firmware
deb-src http://deb.debian.org/debian/ bookworm main non-free-firmware

deb http://security.debian.org/debian-security bookworm-security main non-free-firmware
deb-src http://security.debian.org/debian-security bookworm-security main non-free-firmware

deb http://deb.debian.org/debian/ bookworm-updates main non-free-firmware
deb-src http://deb.debian.org/debian/ bookworm-updates main non-free-firmware
```

Comando apt-cache

Muestra gran parte de la información almacenada en la BD interna de APT (que se ha actualizado después de hacer `apt update`).

- `apt-cache search palabra_clave`: realiza una búsqueda en las descripciones de los paquetes, para ello usa el comando `grep`.
- `apt-cache show paquete`: muestra las cabeceras de las versiones disponibles de un paquete.
- `apt-cache policy`: muestra las prioridades para la instalación de todos los paquetes de una fuente o de un paquete concreto.

```
$ apt-cache policy iptables
iptables:
  Instalados: 1.8.7-1
  Candidato:  1.8.7-1
  Tabla de versión:
 *** 1.8.7-1 500
        500 http://deb.debian.org/debian bullseye/main amd64 Packages
        100 /var/lib/dpkg/status
```

Comando `apt-file`

- Puede ser necesario instalarlo en vuestra MV.
- Busca dentro de todos los paquetes existentes un patrón dado.

```
$ sudo apt-file search bin/emacs
elpa-emacsql-sqlite: /usr/bin/emacsql-sqlite
emacs-bin-common: /usr/bin/emacsclient.emacs
emacs-gtk: /usr/bin/emacs-gtk
emacs-lucid: /usr/bin/emacs-lucid
emacs-nox: /usr/bin/emacs-nox
emacspeak: /usr/sbin/emacspeakconfig
```

- Actualiza su BD al hacer `sudo apt-file update`

Posible línea en /etc/crontab

```
@weekly root test -x /usr/bin/apt-file && /usr/bin/apt-file update >> /dev/null 2>&1
```

- Puede usarse también `dpkg -L paquete` para mostrar todos los ficheros instalados a partir de un paquete determinado,
- y `dpkg -S patron` que busca un nombre de fichero en los paquetes instalados.

Autenticidad de un paquete (I)

- En Debian, cada paquete viene firmado para garantizar que, cuando se instale, proviene de su encargado oficial y no de un tercero.
- Esta seguridad funciona a través de una cadena de hashes y una firma (más información en `apt-secure(8)`).
  - Fichero `InRelease` a partir de Debian 10 (antes `Release`) contiene la lista de todos los ficheros incluidos en el paquete y sus hashes SHA256.
- Además, APT necesita un conjunto de claves públicas GnuPG de confianza para verificar las firmas en el fichero `InRelease`.
  - En ficheros en /etc/apt/trusted.gpg.d/

```
ls /etc/apt/trusted.gpg.d/
debian-archive-bullseye-automatic.gpg
debian-archive-bullseye-security-automatic.gpg
debian-archive-bullseye-stable.gpg
debian-archive-buster-automatic.gpg
debian-archive-buster-security-automatic.gpg
debian-archive-buster-stable.gpg
debian-archive-stretch-automatic.gpg
debian-archive-stretch-security-automatic.gpg
debian-archive-stretch-stable.gpg
```

  - En versiones previas: registro de claves /etc/apt/trusted.gpg (gestionado por el comando `apt-key`).

Autenticidad de un paquete (II)

- Las claves oficiales de Debian son proporcionadas y mantenidas al día por el paquete `debian-archive-keyring` que las guarda en `/etc/apt/trusted.gpg.d/`.
- Por problemas de seguridad, Debian 11 fue la última distribución donde esté disponible `apt-key`.

Autenticidad de un paquete (III)

```
$ apt-key fingerprint
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
/etc/apt/trusted.gpg.d/debian-archive-bullseye-automatic.gpg
-----------------------------------------------------------
pub   rsa4096 2021-01-17 [SC] [caduca: 2029-01-15]
      1F89 983E 0081 FDE0 18F3  CC96 73A4 F27B 8DD4 7936
uid        [desconocida] Debian Archive Automatic Signing Key (11/bullseye) <ftpmaster@debian.org>
sub   rsa4096 2021-01-17 [S] [caduca: 2029-01-15]

/etc/apt/trusted.gpg.d/debian-archive-bullseye-security-automatic.gpg
--------------------------------------------------------------------
pub   rsa4096 2021-01-17 [SC] [caduca: 2029-01-15]
      AC53 0D52 0F2F 3269 F5E9  8313 A484 4904 4AAD 5C5D
uid        [desconocida] Debian Security Archive Automatic Signing Key (11/bullseye) <ftpmaster@debian.org>
sub   rsa4096 2021-01-17 [S] [caduca: 2029-01-15]
...
```

- Cuando se agrega una nueva fuente de paquetes de terceros a source.list se necesita informar a APT para que confíe en esta fuente.
  - Necesitamos la clave pública → fichero `key.gpg`

En sistemas antiguos

```
$ apt-key add < key.asc
```

Autenticidad de un paquete (y IV)

Añadir directamente la clave fichero.gpg en `/etc/apt/trusted.gpg.d/` puede ocasionar un problema de seguridad

- Si se añade directamente al directorio, es válido para cualquier paquete que se quiera instalar.
- Un atacante malicioso puede proporcionar su "versión" de un mirror Debian usando una clave ya existente.
- Se debe guardar la clave en el directorio `/usr/share/keyrings/` y usar el campo **`signed-by`** en el fichero `sources.list`

Descarga del certificado

```
$curl https://deriv.example.net/debian/deriv-archive-keyring.pgp |
gpg -o /usr/share/keyrings/deriv-archive-keyring.pgp --dearmor
```

Contenido de `/etc/apt/sources.list.d/deriv.list`

```
deb [signed-by=/usr/share/keyrings/deriv-archive-keyring.pgp]
https://deriv.example.net/debian/ stable main
```

Actualizando de una distribución estable a la siguiente

- No tiene riesgo 0 $\Longrightarrow$ **BACKUP** siempre antes
- Se debería intentar sólo actualizar aquellos paquetes que sean necesarios.

1. Cambiar el fichero `/etc/apt/sources.list`
   - Si `stable` indica la distribución, no hace falta hacer nada.
   - Si no, cambiar el nombre a la última que queremos.
2. `sudo apt update`, para hacer update de las fuentes de los paquetes.
3. `sudo apt upgrade`, para correcciones menores y mejoras.
4. `sudo apt full-upgrade`
   - Lee detenidamente cuando pregunta la instalación.
5. Puedes borrar los que ya no son necesarios con
   `sudo apt autoremove --purge`

Manteniendo el sistema actualizado

- Se puede optar por recibir un correo automático cuando existan actualizaciones usando el paquete `apticron`.
  - Se debe modificar el fichero `/etc/apticron/apticron.conf` para recibir un correo en una determinada dirección y saber si se debe actualizar.
  - Necesita tener instalado MTA (mail transfer agent) previamente (SMTP, por ejemplo).
- Uso de actualizaciones automáticas con el paquete `unattended-upgrades`.

Instalación de unattended-upgrades

```
$ sudo apt update
$ sudo apt install unattended-upgrades
$ sudo nano /etc/apt/apt.conf.d/50unattended-upgrades
```

- Descomenta las 2 primeras líneas en el fichero

```
"origin=Debian,codename=${distro_codename}-updates";
"origin=Debian,codename=${distro_codename}-proposed-updates";
"origin=Debian,codename=${distro_codename},label=Debian";
"origin=Debian,codename=${distro_codename},label=Debian-Security";
```

- Para permitir unattended-upgrades

```
$ sudo dpkg-reconfigure --priority=low unattended-upgrades
```

- Sale una ventana emergente, debes responder Sí.

```
$ cat  /etc/apt/apt.conf.d/20auto-upgrades
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Unattended-Upgrade "1";
$ sudo systemctl status unattended-upgrades.service
```

Fichero de log y Deshabilitando unattended-upgrades

- El log está en /var/log/unattended-upgrades/
  - Puedes usar cat si son ficheros de texto o zcat si están ya comprimidos.
- Para deshabilitar vuelve a ejecutar el comando

```
$ sudo dpkg-reconfigure --priority=low unattended-upgrades
```

- Sale una ventana emergente, debes responder No.

```
$ cat  /etc/apt/apt.conf.d/20auto-upgrades
APT::Periodic::Update-Package-Lists "0";
APT::Periodic::Unattended-Upgrade "0";
```

- Si quisieses comprobar los updates pero no instalarlos, podrías modificar el fichero /etc/apt/apt.conf.d/20auto-upgrades

```
$ cat  /etc/apt/apt.conf.d/20auto-upgrades
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Unattended-Upgrade "0";
```

Apartado 3

Compilando un paquete desde las fuentes

¿Para qué?

- Inspeccionar el código fuente para encontrar un error.
- Añadir nuevas características en los paquetes que ya no se están desarrollando activamente.
- Instalar la versión más reciente de un paquete desde el código fuente, porque los paquetes en los repositorios oficiales pueden ser un poco antiguos.

¿Cómo? (I)

1. Instala `dpkg-dev`: `sudo apt install dpkg-dev`
2. Comprueba en `/etc/apt/sources.list` que la línea de fuentes (empieza por dev-src) está descomentada.
3. Haz update: `sudo apt update`
4. Crea un directorio temporal para bajarte las fuentes y accede a él (mkdir y cd).
5. Bájate las fuentes con `apt source hello`

```
$ ls

hello-2.10 hello_2.10-2.debian.tar.xz hello_2.10-2.dsc hello_2.10.orig.tar.gz
$ ls hello-2.10/
ABOUT-NLS    config.in     doc          maint.mk     README          TODO
aclocal.m4   configure     GNUmakefile  Makefile.am  README-dev
AUTHORS      configure.ac  hello.1      Makefile.in  README-release
build-aux    contrib       INSTALL      man          src
ChangeLog    COPYING       lib          NEWS         tests
ChangeLog.0  debian        m4           po           THANKS
```

6. Instala las dependencias del paquete para después poder compilarlo

```
$ sudo apt build-dep hello
```

¿Cómo? (II)

7. Accede al directorio de las fuentes `cd hello-2.10`
8. Modifica en `src/hello.c` y que aparezca tu nombre y 1er apellido. Puede ser en la ayuda, por ejemplo.
9. Desde el directorio de `hello-2.10`, haz

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

   - `dpkg-buildpackage`: comando para construir paquetes binarios o de fuentes dsde las fuentes.
   - `-rfakeroot`: Crea el entorno fakeroot que simula privilegios de root
   - `-b`: sólo el paquete binario
   - `-uc`: no firmes criptográficamente el changelog.
   - `-us`: no firmes el paquete.
10. Debes encontrar un paquete llamado `hello_2.10-2_amd64.deb`
11. Haz `sudo dpkg -i hello_2.10-2_amd64.deb`

- Si no funciona puede ser que sea por no estar en el directorio adecuado o por fallar alguna prueba unitaria (si cambias el mensaje, por ejemplo, va a dar error).

Apartado 4

Gestión de drivers y módulos

Drivers

- El kernel de Linux gestiona los dispositivos hardware utilizando drivers.
  - Muchos de los drivers están directamente compilados en el sistema.
- El driver exporta sus funciones al sistema a través de una abstracción en el sistema de ficheros: `/dev/blah`.
  - Si escribimos o leemos de `/dev/blah` le estamos diciendo al driver que haga dichas operaciones sobre el hardware que controla.
  - De esta forma, el sistema de control de acceso que impone el sistema de ficheros se aplica también al driver del dispositivo físico al dispositivo
    - rwx del propietario (owner), grupo (group), otros (other) + permisos especiales
- El sistema de capacidades de Linux, SELinux y Apparmor caen fuera del alcance de este curso.

Módulos (I)

¿Tiene sentido incrementar mucho el tamaño del kernel para dar soporte a un hardware que no sabemos siquiera si vamos a conectar (o siquiera comprar)?

- Los módulos encapsulan drivers que se cargan dinámicamente en el kernel.
- Es necesario que el sistema esté preparado para poder recibir dichos módulos.

Módulos (y II)

- modprobe(8) nos permite probar de manera segura a insertar un nuevo módulo en el kernel.
- insmod(8) y rmmod(8) permite forzar la inserción o extracción, pero pueden dar errores si no se resuelven las dependencias en el orden adecuado.
- lsmod(8) permite ver qué módulos están actualmente insertados.
- modinfo(8) nos da información acerca del módulo: fichero, versión, autor, firma y parámetros.
- depmod(8) permite crear la lista de dependencias de versiones de los módulos que declaran (`/lib/modules/<version>/modules.dep`). Se regenera con depmod -a.

Bibliografía

- Raphaël Hertzog, Roland Mas. *The Debian Administrator's Handbook. Debian 11. Debian Bullseye from Discovery to Mastery*. Capítulos 5, 6 y 15. `https://debian-handbook.info/browse/stable/index.html`

Administración de redes Linux

Instalación de paquetes software

Iria M. Estévez Ayres

uc3m | Universidad Carlos III de Madrid

Departamento de Ingeniería Telemática

Marzo 2025