Gestión de paquetes: una explicación práctica

TRABAJO EN PROGRESO

Anatomía de un paquete de Slackware

Un paquete de Slackware es un archivo comprimido simple TGZ o TXZ que contiene:

- la estructura de árbol de archivos y directorios;
- secuencias de comandos posteriores a la instalación;
- La descripción del paquete.

El nombre de cada paquete proporciona una serie de informaciones:

- el nombre del programa;
- la versión del programa;
- la arquitectura del paquete;
- el número de compilación.

Aquí hay algunos ejemplos:

- `emacs-24.2-i486-1`
- `mozilla-firefox-15.0.1-i486-1`
- `vim-7.3.645-x86_64-1`

Administrar paquetes de Slackware usando las herramientas tradicionales

Desde sus primeros lanzamientos, Slackware proporciona una colección de herramientas simples, las `pkgtools` , que permiten al usuario instalar, actualizar y eliminar paquetes de software, así como también construirlos:

- `installpkg`
- `removepkg`
- `upgradepkg`
- `explodepkg`
- `makepkg`

Instalación de paquetes de software

Instale el editor de Emacs desde el DVD de Slackware [1]:

```
# mount /dev/cdrom /mnt/cdrom
# cd /mnt/cdrom/slackware/e
# installpkg emacs-24.2-i486-1.txz
Verifying package emacs-24.2-i486-1.txz.
Installing package emacs-24.2-i486-1.txz [ADD]:
PACKAGE DESCRIPTION:
# emacs (GNU Emacs)
#
# Emacs is the extensible, customizable, self-documenting real-time
# display editor. If this seems to be a bit of a mouthful, an
# easier explanation is that Emacs is a text editor and more. At
# its core is an interpreter for Emacs Lisp, a dialect of the Lisp
# programming language with extensions to support text editing.
# This version supports X.
#
# http://www.gnu.org/software/emacs/
#
Executing install script for emacs-24.2-i486-1.txz.
Package emacs-24.2-i486-1.txz installed.
```

Si está utilizando el conjunto de CD, Emacs está en el primer CD.

Comprobando si un paquete está instalado

El proceso de instalación del paquete ha creado una nueva entrada en`/var/log/packages` :

```
# ls /var/log/packages/em*
/var/log/packages/emacs-24.2-i486-1
```

Saber si un paquete está instalado se reduce a verificar la existencia de la entrada correspondiente en `/var/log/packages` . Ejemplo:

```
# ls /var/log/packages/*firefox*
/var/log/packages/mozilla-firefox-15.0.1-i486-1
```

Firefox está instalado en el sistema, en la versión 15.0.1. Otro ejemplo:

```
# ls /var/log/packages/kdebase*
ls: cannot access /var/log/packages/kdebase*: No such file or directory
```

No hay un paquete `kdebase- *` instalado en el sistema.

Eliminar un paquete

Use `removepkg` para eliminar un paquete instalado. El comando puede tomar el nombre base simple del paquete como argumento. Ejemplo:

```
# removepkg emacs
```

También es posible proporcionar el nombre completo como argumento. En ese caso, es mejor llamar al comando desde `/var/log/packages` y usar la finalización de tabulación:

```
# cd /var/log/packages
# removepkg emacs-24.2-i486-1
```

Actualizar un paquete

Slackware proporciona actualizaciones de seguridad para sus últimos lanzamientos. Visite el sitio oficial para saber más sobre las últimas actualizaciones:

```
# links http://www.slackware.com
```

- Siga el enlace `ChangeLogs` .

1. Consulte `ChangeLog estable de Slackware` .
2. Lea el archivo `ChangeLog.txt` correspondiente a la arquitectura de su sistema.

También puede usar el navegador Enlaces para buscar actualizaciones manualmente. Antes de iniciar Enlaces, cree un directorio `/ root / updates` [2] para almacenar sus actualizaciones descargadas:

```
# cd
# mkdir updates
# cd updates/
# links mirrors.slackware.com
```

- Siga el enlace `Slackware File Tree` .

1. Consulte el directorio correspondiente a su versión y arquitectura.
2. Cambie al directorio `parches / paquetes` .
3. Descargue las actualizaciones disponibles.

Salga de Links e instale sus actualizaciones así:

```
# upgradepkg bind-9.9.1_P4-i486-1_slack14.0.txz

+==============================================================================
===
| Upgrading bind-9.9.1_P3-i486-1 package using ./bind-9.9.1_P4-
i486-1_slack14.0.txz
+==============================================================================
===
Pre-installing package bind-9.9.1_P4-i486-1_slack14.0...
Removing package /var/log/packages/bind-9.9.1_P3-i486-1-
upgraded-2012-11-21,12:14:32...
```

```
  --> Deleting /usr/doc/bind-9.9.1-P3/CHANGES
  --> Deleting /usr/doc/bind-9.9.1-P3/COPYRIGHT
  --> Deleting /usr/doc/bind-9.9.1-P3/FAQ
  ...
Verifying package bind-9.9.1_P4-i486-1_slack14.0.txz.
Installing package bind-9.9.1_P4-i486-1_slack14.0.txz:
PACKAGE DESCRIPTION:
bind (DNS server and utilities)
#
# The named daemon and support utilities such as dig, host, and
# nslookup.  Sample configuration files for running a simple caching
# nameserver are included.  Documentation for advanced name server
# setup can be found in /usr/doc/bind-9.x.x/.
#
Executing install script for bind-9.9.1_P4-i486-1_slack14.0.txz.
Package bind-9.9.1_P4-i486-1_slack14.0.txz installed.
Package bind-9.9.1_P3-i486-1 upgraded with new package
./bind-9.9.1_P4-i486-1_slack14.0.txz.
```

Otro ejemplo:

```
# upgradepkg iptables-1.4.14-i486-2_slack14.0.txz
```

Conozca más sobre el contenido de un paquete

Cada paquete tiene una entrada correspondiente en `/ var / log / packages` . Todos estos son archivos de texto simples que proporcionan información sobre el contenido de los respectivos paquetes. Ejemplo:

```
# less /var/log/packages/wget-1.14-i486-1
PACKAGE NAME:     wget-1.14-i486-1
COMPRESSED PACKAGE SIZE:     478.5K
UNCOMPRESSED PACKAGE SIZE:     2.0M
PACKAGE LOCATION: /var/log/mount/slackware/n/wget-1.14-i486-1.txz
PACKAGE DESCRIPTION:
wget: wget (a non-interactive network retriever)
wget:
wget: GNU Wget is a free network utility to retrieve files from the
wget: World Wide Web using HTTP and FTP, the two most widely used Internet
wget: protocols.  It works non-interactively, thus enabling work in the
wget: background after having logged off.
wget:
wget: The author of Wget is Hrvoje Niksic <hniksic@srce.hr>.
wget:
wget:
wget:
FILE LIST:
./
```

```
install/
install/slack-desc
install/doinst.sh
usr/
usr/bin/
usr/bin/wget
usr/man/
usr/man/man1/
usr/man/man1/wget.1.gz
usr/info/
usr/info/wget.info.gz
...
```

Administrar paquetes de Slackware con slackpkg

La utilidad `slackpkg` se ha incluido oficialmente en Slackware desde la versión 13.0. Permite al usuario administrar paquetes de Slackware mucho más cómodamente.

Algunas observaciones:

1. Solo los paquetes oficiales de Slackware son manejados por `slackpkg` .
2. Los paquetes de terceros se pueden administrar si usa el complemento `slackpkg +` de Matteo Rossini.
3. Las dependencias aún deben gestionarse manualmente.

Configuración inicial

Edite `/etc/slackpkg/mirrors` y comente *una y solo una* fuente del paquete, por ejemplo:

```
# /etc/slackpkg/mirrors
...
# FRANCE (FR)
ftp://mirror.ovh.net/mirrors/ftp.slackware.com/slackware-14.0/
# http://mirror.ovh.net/mirrors/ftp.slackware.com/slackware-14.0/
```

Si está utilizando una versión estable de Slackware, no se equivoque en la sección y descomente un espejo de `Slackware-current` . ¡Si lo hace, se actualizará a una versión de desarrollo de Slackware!

Si prefiere administrar paquetes localmente sin el beneficio de las actualizaciones, aún puede usar el DVD de instalación de Slackware como fuente del paquete. En ese caso, deberá configurar el punto de montaje predeterminado:

```
# /etc/slackpkg/mirrors
...
#------------------------------------------------------------------
```

```
# Local CD/DVD drive
#-------------------------------------------------------------
cdrom://mnt/cdrom/
...
```

No olvides montar el DVD antes de llamar a `slackpkg` :

```
# mount /dev/cdrom /mnt/cdrom
```

Actualice la información sobre los paquetes disponibles:

```
# slackpkg update
```

Tenga en cuenta que el comando anterior no instala ninguna actualización de paquete. Solo actualiza la lista interna de paquetes que *puede* instalar.

Siempre es una buena idea invocar `slackpkg update` antes de buscar, instalar o actualizar un paquete, para que la información del sistema sobre los paquetes disponibles esté actualizada.

Instalación de paquetes

Ejemplo con un solo paquete:

```
# slackpkg install mplayerplug-in
```

Confirme la instalación en la pantalla siguiente, y el paquete se descarga e instala automáticamente.

También puede proporcionar varios paquetes como argumento:

```
# slackpkg install mplayerplug-in bittorrent
```

También puede administrar grupos completos de paquetes:

```
# slackpkg install kde
```

Otro ejemplo para grupos de paquetes:

```
# slackpkg install xfce
```

Eliminar paquetes

Ejemplo con un solo paquete:

```
# slackpkg remove mplayerplug-in
```

Como se indicó anteriormente, confirme la eliminación del paquete en la pantalla siguiente.

Eliminar varios paquetes a la vez:

```
# slackpkg remove mplayerplug-in bittorrent
```

Del mismo modo, puede eliminar un grupo de paquetes completo:

```
# slackpkg remove kde
```

O:

```
# slackpkg remove xfce
```

Actualización de paquetes

Cuando hay una actualización de paquete disponible, puede instalarla con el siguiente comando:

```
# slackpkg upgrade iptables
```

Actualice varios paquetes a la vez:

```
# slackpkg upgrade mozilla-firefox mozilla-thunderbird
```

Es una práctica común mantener todo su sistema actualizado:

```
# slackpkg upgrade-all
```

Buscar paquetes o archivos específicos

Busque un paquete específico:

```
# slackpkg search k3b
Looking for k3b in package list. Please wait... DONE
The list below shows all packages with name matching "k3b".
[uninstalled] - k3b-2.0.2_20120226.git-i486-1
```

Si el paquete ya está instalado, esto es lo que obtienes:

```
# slackpkg search Terminal
Looking for Terminal in package list. Please wait... DONE
The list below shows all packages with name matching "Terminal".
[ installed ] - Terminal-0.4.8-i486-1
```

También puede buscar archivos individuales. La búsqueda finalmente se mostrará en o varios

paquetes que contengan el archivo en cuestión:

```
# slackpkg file-search libncurses.so
Looking for libncurses.so in package list. Please wait... DONE
The list below shows the packages that contains "libncurses\.so" file.
[ installed ] - aaa_elflibs-14.0-i486-4
[ installed ] - ncurses-5.9-i486-1
```

Si desea saber más sobre el contenido de un paquete:

```
# slackpkg info mesa

PACKAGE NAME:  mesa-8.0.4-i486-1.txz
PACKAGE LOCATION:  ./slackware/x
PACKAGE SIZE (compressed):  19208 K
PACKAGE SIZE (uncompressed):  83930 K
PACKAGE DESCRIPTION:
mesa: mesa (a 3-D graphics library)
mesa:
mesa: Mesa is a 3-D graphics library with an API very similar to that of
mesa: another well-known 3-D graphics library.  :-)  The Mesa libraries are
mesa: used by X to provide both software and hardware accelerated graphics.
mesa:
mesa: Mesa was written by Brian Paul.
mesa:
```

Limpieza del sistema

Eliminar todos los paquetes de terceros:

```
# slackpkg clean-system
```

Si decide conservar algunos de los paquetes, simplemente desactívelos en la pantalla siguiente.

También puede usar `slackpkg` para reparar un paquete dañado. Digamos que borré accidentalmente el archivo `/ usr / bin / glxgears` . Primero, tengo que buscar el paquete que proporciona ese archivo:

```
# slackpkg file-search glxgears
Looking for glxgears in package list. Please wait... DONE
The list below shows the packages that contains "glxgears" file.
[ installed ] - mesa-8.0.4-i486-1
```

Con esta información, simplemente puedo reinstalar el paquete:

```
# slackpkg reinstall mesa
```

Reconstruir paquetes oficiales

Slackware proporciona el código fuente de todo el sistema en el directorio `fuente` . Cada paquete de sistema binario tendrá su directorio fuente correspondiente. Estos directorios fuente generalmente contienen:

- el código fuente de la aplicación o la biblioteca;
- su receta de fabricación en forma de archivo `* .SlackBuild` ;
- la descripción del paquete en un archivo `slack-desc` ;
- eventualmente, un script posterior a la instalación llamado `doinst.sh` ;
- varios otros archivos como parches, entradas de menú personalizadas, etc.

Construir un paquete desde la fuente

En el siguiente ejemplo, crearemos la aplicación `Terminal` a partir del código fuente proporcionado por Slackware. Es posible que desee eliminar el paquete correspondiente si está instalado.

El paquete `Terminal` es el terminal de Xfce. En Slackware 14.1, el paquete ha cambiado de nombre a “xfce4-terminal”.

```
# removepkg Terminal
```

Elija un lugar apropiado en su sistema para almacenar el código fuente y los scripts, por ejemplo:

```
# cd
# mkdir -pv source/Terminal
mkdir: created directory 'source'
mkdir: created directory 'source/Terminal'
# cd source/Terminal/
# links mirrors.slackware.com
```

Obtenga el contenido del directorio `source/xfce/Terminal` en un espejo de Slackware. Esto es lo que obtenemos:

```
# ls -lh
total 1,4M
-rw-r--r-- 1 root root  821 nov.  24 15:09 slack-desc
-rw-r--r-- 1 root root 1,4M nov.  24 15:11 Terminal-0.4.8.tar.xz
-rw-r--r-- 1 root root 3,6K nov.  24 15:10 Terminal.SlackBuild
```

Haga que el archivo `Terminal.SlackBuild` sea ejecutable e inicie el proceso de construcción:

```
# chmod +x Terminal.SlackBuild
# ./Terminal.SlackBuild
```

El script inicia la compilación del paquete. Si todo sale como se esperaba, la operación sale con el

siguiente mensaje:

```
Slackware package /tmp/Terminal-0.4.8-i486-1.txz created.
```

Ahora podemos instalar el paquete resultante:

```
# installpkg /tmp/Terminal-0.4.8-i486-1.txz
```

Modificar un paquete oficial de Slackware

La razón principal para reconstruir un paquete oficial es modificarlo, por ejemplo, para agregar o quitar ciertas funcionalidades. En el siguiente ejemplo, reconstruiremos el paquete `audacious-plugins` para modificar el reproductor de audio Audacious. La aplicación Vanilla tiene dos interfaces gráficas diferentes, y desactivaremos una de ellas.

Comencemos por eliminar el paquete si está instalado:

```
# removepkg audacious-plugins
```

Now create a suitable directory to store the source code:

```
# cd /root/source
# mkdir audacious-plugins
# cd audacious-plugins
# links mirrors.slackware.com
```

Obtenga el contenido del directorio `/source/xap/audacious-plugins` y haga que el script `audacious-plugins.SlackBuild` sea ejecutable:

```
# chmod +x audacious-plugins.SlackBuild
# ls -lh
total 1,4M
-rw-r--r-- 1 root root 1,4M nov.  24 15:28 audacious-plugins-3.3.1.tar.xz
-rwxr-xr-x 1 root root 4,0K nov.  24 15:28 audacious-plugins.SlackBuild*
-rw-r--r-- 1 root root  892 nov.  24 15:28 slack-desc
```

Ahora edite `audacious-plugins.SlackBuild` y agregue una opción:

```
...
# Configure:
CFLAGS="$SLKCFLAGS" \
CXXFLAGS="$SLKCFLAGS" \
./configure \
  --prefix=/usr \
  --libdir=/usr/lib${LIBDIRSUFFIX} \
  --sysconfdir=/etc \
  --mandir=/usr/man \
  --enable-amidiplug \
```

```
  --disable-gtkui \          -> add this option
  --program-prefix= \
  --program-suffix= \
  ${ARCHOPTS} \
  --build=$ARCH-slackware-linux
...
```

Compila e instala el paquete:

```
# ./audacious-plugins.SlackBuild
...
Slackware package /tmp/audacious-plugins-3.3.1-i486-1.txz created.
# installpkg /tmp/audacious-plugins-3.3.1-i486-1.txz
```

Elegir las opciones de configuración para compilar

El script de configuración de la fuente (o más exactamente la línea a veces muy larga en SlackBuild que comienza con `./configure`) a menudo muestra una descripción general de las opciones activadas y/o desactivadas. Para interrumpir el proceso de construcción del paquete y mostrar esta descripción general, puede editar temporalmente el SlackBuild de esta manera:

```
...
# Configure:
CFLAGS="$SLKCFLAGS" \
CXXFLAGS="$SLKCFLAGS" \
./configure \
  --prefix=/usr \
  --libdir=/usr/lib${LIBDIRSUFFIX} \
  --sysconfdir=/etc \
  --mandir=/usr/man \
  --enable-amidiplug \
  --program-prefix= \
  --program-suffix= \
  ${ARCHOPTS} \
  --build=$ARCH-slackware-linux

exit 1          -> add this option to interrupt the script

# Build and install:
make $NUMJOBS || make || exit 1
make install DESTDIR=$PKG || exit 1
...
```

Ahora ejecute el script y espere unos segundos para ver la descripción general de la configuración:

```
# ./audacious-plugins.SlackBuild
...
Configuration:
...
```

```
Interfaces
----------
GTK (gtkui):                            yes
Winamp Classic (skins):                 yes
```

Use la opción `./configure –help` para mostrar una lista de todas las opciones posibles:

```
# tar xvf audacious-plugins-3.3.1.tar.xz
# cd audacious-plugins-3.3.1
# ./configure --help | less
...
--disable-speedpitch    disable Speed and Pitch effect plugin
--disable-gtkui         disable GTK interface (gtkui)
--disable-skins         disable Winamp Classic interface (skins)
--disable-lyricwiki     disable LyricWiki plugin (default=enabled)
...
```

El script SlackBuild ya se encarga de descomprimir automáticamente el archivo tar de origen en el directorio `/tmp` . Entonces simplemente puede ejecutar `./configure –help | menos` desde este directorio, sin descomprimir manualmente el tarball fuente al directorio actual.

En el presente caso, la activación de ciertas funcionalidades como, por ejemplo, la administración de formatos de audio propietarios dependerá de la presencia de las bibliotecas correspondientes en su sistema.

Una vez que haya elegido todas sus opciones de configuración, elimine el comando temporal `salir 1` en su script e inicie el proceso de compilación e instalación:

```
# ./audacious-plugins.SlackBuild
...
Slackware package /tmp/audacious-plugins-3.3.1-i486-1.txz created.
# installpkg /tmp/audacious-plugins-3.3.1-i486-1.txz
```

Crear paquetes de terceros

Slackware ofrece solo una selección limitada de paquetes en comparación con distribuciones gigantes como Ubuntu o Debian. La mayoría de las veces, querrás instalar un paquete que no sea proporcionado por la distribución. En ese caso, ¿qué puede hacer un niño pobre?

El sitio web SlackBuilds.org es probablemente la mejor dirección para encontrar software de terceros. No encontrará ningún paquete allí, porque SlackBuilds.org *no* es un repositorio de paquetes binarios ni lo será nunca. Es una colección extremadamente limpia y bien organizada de scripts de compilación, cada uno revisado y probado. El uso de estos scripts le permitirá construir sobre cada pieza de software de terceros bajo el sol.

Construyendo paquetes usando los scripts de SlackBuilds.org

En el siguiente ejemplo, crearemos e instalaremos el paquete `cowsay` utilizando el script de compilación proporcionado por SlackBuilds.org.

Para empezar, `cd` en el directorio de compilación que hemos definido anteriormente:

```
# cd /root/source
```

Descargue los siguientes componentes en este directorio:

1. el tarball comprimido que contiene los scripts para construir el paquete;
2. el código fuente comprimido tarball.

En nuestro caso:

```
# links http://slackbuilds.org
```

- En el campo Buscar en la esquina superior izquierda de la pantalla, escriba `cowsay` , mueva el cursor a `Buscar` (tecla CursorDown) y confirme presionando Enter.

1. Siga el enlace `cowsay` en la página de resultados de búsqueda.
2. Una vez que esté en la página `cowsay` , descargue el SlackBuild (`cowsay.tar.gz`) y el código fuente (`cowsay-3.03.tar.gz`) y salga de Links.

Alternativamente, use `lynx` en lugar de `enlaces` .

Aquí están nuestros dos tarballs descargados:

```
# ls -l cowsay*
-rw-r--r-- 1 root root 15136 nov.  25 08:14 cowsay-3.03.tar.gz
-rw-r--r-- 1 root root  2855 nov.  25 08:14 cowsay.tar.gz
```

Descomprima el tarball que contiene los scripts:

```
# tar xvzf cowsay.tar.gz
cowsay/
cowsay/cowsay.SlackBuild.patch
cowsay/README
cowsay/slack-desc
cowsay/cowsay.SlackBuild
cowsay/cowsay.info
```

Eventualmente, puede hacer una pequeña limpieza y eliminar el tarball:

```
# rm -f cowsay.tar.gz
```

Ahora mueva el archivo tar de origen al directorio `cowsay/` recién creado:

```
# mv -v cowsay-3.03.tar.gz cowsay/
« cowsay-3.03.tar.gz » -> « cowsay/cowsay-3.03.tar.gz »
```

Esto es lo que tenemos:

```
# tree cowsay
cowsay
|-- cowsay-3.03.tar.gz
|-- cowsay.info
|-- cowsay.SlackBuild
|-- cowsay.SlackBuild.patch
|-- README
`-- slack-desc
```

Ahora `cd` en ese directorio. Compruebe si el SlackBuild `cowsay` es ejecutable y luego inícielo para comenzar la construcción del paquete:

```
# cd cowsay/
# ls -l cowsay.SlackBuild
-rwxr-xr-x 1 kikinovak users 1475 mai   27  2010 cowsay.SlackBuild*
# ./cowsay.SlackBuild
...
```

Si todo va bien, el proceso arroja un paquete en `/ tmp` , o más exactamente en el directorio `$ OUTPUT` definido por el script:

```
...
Slackware package /tmp/cowsay-3.03-noarch-1_SBo.tgz created.
```

Todo lo que queda por hacer es instalar el paquete usando `installpkg` :

```
# installpkg /tmp/cowsay-3.03-noarch-1_SBo.tgz
# cowsay Hi there !
 -------------
< Hi there ! >
 -------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
```

Gestión de dependencias de paquetes

Algunos paquetes requieren la presencia de otros paquetes, ya sea para compilar (*dependencias de compilación*) y / o para ejecutar (*dependencias de tiempo de ejecución*) correctamente. En algunos casos, un paquete requerido puede depender de uno o más paquetes, y así sucesivamente.

Para tomar un ejemplo, echemos un vistazo a la página `libgnomeprint` en SlackBuilds.org. La descripción del paquete es seguida por la siguiente advertencia:

```
This requires: libgnomecups.
```

Además, cada script tarball contiene un archivo `*.info` que establece explícitamente todas las dependencias de paquetes requeridas. Si miramos el archivo `libgnomeprint.info`, encontraremos el campo `REQUERIMIENTOS`:

```
PRGNAM="libgnomeprint"
VERSION="2.18.8"
HOMEPAGE="http://www.gnome.org"
...
REQUIRES="libgnomecups" ----> package dependency
...
```

El campo `REQUISITOS` se ha introducido con Slackware 14.0.

Esto simplemente significa que antes de construir el paquete `libgnomeprint`, tenemos que construir e instalar el paquete `libgnomecups`.

Además de las dependencias estrictamente requeridas, un paquete también puede tener algunas dependencias opcionales para ofrecer alguna funcionalidad adicional. Como ejemplo, el editor de texto de Leafpad se puede construir contra las dependencias opcionales `libgnomeprint` y `libgnomeprintui`.

TRABAJO EN PROGRESO

Funetes

- Originalmente escrito por Niki Kovacs
- Traducido por: Victor 2019/08/13 19:37 (UTC)

slackware, package, management, author kikinovak

1)

si aún no está instalado

2)

Esto es solo un ejemplo, por supuesto. Siéntase libre de usar cualquier lugar conveniente en su sistema

From:
https://docs.slackware.com/ - **SlackDocs**

Permanent link:
https://docs.slackware.com/es:slackware:package_management_hands_on

Last update: **2019/08/13 19:39 (UTC)**