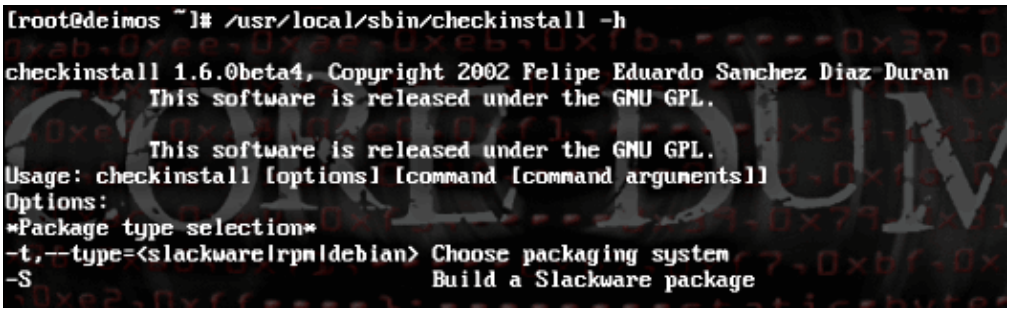
CheckInstall

par Mario M. Knopf
(homepage)

L´auteur:

Mario apprécie de travailler avec Linux, les réseaux et tout sujet relatif à la sécurité.



Résumé:

Checkinstall est un utilitaire qui génère automatiquement des paquets RPM, Debian ou Slackware à partir de sources en tar.gz. Ce qui permet une installation « propre » et la désinstallation de quasi n'importe quel code source en tar.gz.

Introduction

Il arrive fréquemment que des programmes que l'on désire tester n'existent qu'au format tar.gz sans être disponibles au format RPM ou Debian. Dans un tel cas, vous téléchargez le paquet (package) source, vous le décompactez et le compilez manuellement. Jusque là, c'est tout bon. Oui, mais que faire si vous voulez supprimer ce programme ?

Le fichier *Makefile* peut contenir une routine appropriée pour désinstaller le programme mais ce n'est pas systématique. Bien sûr, il reste la possibilité d'installer le programme dans un répertoire temporaire et d'inscrire tous les fichiers qui ont été créés ou modifiés par l'installation et ce en prévision d'une suppression ultérieure. Mais convenons que cette procédure est plutôt pénible et astreignante surtout si des programmes sont fréquemment compilés à partir des sources. *CheckInstall* [1] écrit par Felipe Eduardo Sánchez Díaz Durán résoud ce problème.

Il est de règle de compiler et d'installer un programme répondant au standard *GNU Autoconf* en utilisant la séquence bien connue de commandes
./configure && make && make install.
Le script shell *configure* tente de deviner les valeurs correctes de différentes variables dépendantes du système, qui seront utilisées après, au cours de la compilation. Il vérifie si tous les requis pour une compilation correcte sont rencontrés et utilise ces valeurs pour générer un *Makefile* dans chaque répertoire du paquet. En outre, le script *configure* génère divers autres fichiers. En résumé, il s'agit de :

- un ou plusieurs *Makefile(s)* dans chaque répertoire/sous−répertoire
- un script shell appelé *config.status*
- un fichier texte *config.log*

- un autre script shell normalement intitulé *config.cache* (optionnel)
- divers fichiers d'en−tête C (*.h) avec des définitions spécifiques au système (optionnel)

Après que le script *configure* ait achevé sa tâche avec succès, vous tapez la commande *make* pour compiler le paquet. Ce qui va générer les binaires exécutables. Il y a la possibilité de lancer, immédiatement après un *make*, un self−test avec la commande *make check*. Mais il s'agit d'une étape éventuelle car le paquet doit supporter cette procédure supplémentaire. Si le *make* a accompli son oeuvre, vous pouvez alors installer le programme compilé grâce à la commande *make install* − pour des raisons évidentes, vous devez bénéficier des droits d'utilisateur privilégié pour l'accomplissement de cette étape. Après que le programme ait été installé vous pouvez enlever les binaires et les fichiers objets du répertoire qui contient le code source en tapant la commande *make clean*. Si vous préférez aussi effacer les fichiers créés par le script *configure*, alors tapez *make distclean*. Cependant ces deux dernières étapes sont, tout comme le *make check*, optionnelles et sont habituellement utilisées par le développeur durant la phase de développement et de test. Elles peuvent aussi être utilisées par l'utilisateur pour épargner la capacité du disque dur ou pour conserver une structure de répertoires claire. En outre, *make distclean* permet de compiler le paquet pour un type d'ordinateur différent.

Des informations plus détaillées sur *GNU Autoconf* sont disponibles dans le manuel en ligne à l'adresse [2]. En plus d'une introduction basique, vous en apprendrez plus sur le « *GNU Build System* », en créant vos propres scripts *configure* , en programmant en *M4* et en créant vos propres macros, programmes shell portables et autres.

CheckInstall

Ainsi que nous l'avons déjà mentionné, la séquence de commande pour générer, à partir de ses sources, un programme répondant au standard *GNU Autoconf*, est :

./configure && make && make install

Arrivé à ce stade, c'est le moment d'utiliser *CheckInstall*. La commande *checkinstall* remplace la commande *make install* . Étant entendu que les deux autres commandes sont maintenues et utilisées tout comme auparavant. Ainsi, la nouvelle séquence de commandes, quand on utilise *CheckInstall*, devient :

./configure && make && checkinstall

Pourtant, l'instruction *checkinstall* lance bien *make install* par défaut et supervise tous les événements en écriture engendrés par l'installation. A cet effet *CheckInstall* utilise le programme *Installwatch* [3], écrit à l'origine par Pancrazio de Mauro. Après que *make install* se soit achevé avec succès, *CheckInstall* génère un paquet Slackware, Debian ou RPM et l'installe avec le gestionnaire par défaut de la distribution tout en laissant une copie du paquet dans le répertoire source ou dans le répertoire de stockage standard. Par ailleurs, il est possible de changer le répertoire de stockage par défaut à travers la variable *PAK_DIR* au sein du fichier de configuration. La copie ainsi « adressée » peut être installée, bien sûr en tenant compte de possibles dépendances, sur d'autres machines du réseau sans avoir à recompiler à chaque fois les sources du paquet.

CheckInstall ne se limite pas à l'utilisation de *make install* mais coopère avec d'autres instructions d'installation. Si, par exemple, le script d'installation est *setup.sh*, la séquence de commande sera :

./configure && make && checkinstall setup.sh

Par ailleurs, il est possible de lancer *CheckInstall* avec diverses options. La commande suivante propose une vue d'ensemble de toutes les options disponibles qui sont réparties dans les sections *Install options*, *Scripting*

options, *Info display options*, *Package tuning options*, *Cleanup options* et *About CheckInstall*:

checkinstall –h

Si *CheckInstall* est lancé avec une de ces options, cette dernière supplantera celles qui sont reprises dans le fichier de configuration *checkinstallrc*.

Mais *CheckInstall* a aussi ses limites. Il ne peut manipuler des programmes liés (linked) statiquement, parce que *Installwatch* n'est pas capable de surveiller les fichiers modifiés durant la procédure d'installation. En général, il y a deux types de bibliothèques de programme : des bibliothèques liées statiquement ou dynamiquement. Ces bibliothèques sont intégrées dans un programme par une directive *include*. Les programmes liés statiquement ont déjà toutes les fonctions de bibliothèques nécessaires et ne doivent pas les charger en mémoire lors de leur exécution. En outre, ils sont indépendants des bibliothèques installées sur le système utilisé car le *Linker* a imbriqué la bibliothèque dans le programme exécutable au moment de la compilation.

Installation

CheckInstall fait déjà parti, depuis longtemps, de l'ensemble des logiciels proposés par les distributions majeures et peut être installé avec le gestionnaire de paquet du système. Si ça n'est pas le cas, vous pouvez télécharger un fichier « tar–balls » ou un paquet pré–compilé convenant à diverses distributions, disponible sur le site web du projet [4].

L'installation de *CheckInstall* est très simple et se déroule en peu d'étapes mais pour une installation réussie, vous avez paradoxalement besoin de *CheckInstall*. Après l'instruction incontournable *make install*, vous tapez *checkinstall*, qui génère un paquet binaire approprié venant du programme compilé. Alors vous pouvez installer ce paquet avec votre gestionnaire de paquet et il est aussi possible de désinstaller ce paquet proprement. Mais avant que *CheckInstall* ne crée le paquet, vous devez répondre à une question sur le gestionnaire de paquet utilisé et vérifier l'exactitude des champs d'information. Ces derniers apparaîtront peu après dans l'en–tête du paquet.

La procédure d'installation de la nouvelle version bêta *checkinstall–1.6.0beta4.tgz* est exposée dans l'encart qui suit. Cela installera *CheckInstall*, *Installwatch* et *makepak*, une version modifiée de *makepkg*. Si les changements de la nouvelle version vous intéressent, jetez un oeil sur la *Release Notes* [5] et/ou dans le *Changelog* [6].

```
$ tar xzf checkinstall-1.6.0beta4.tgz
$ cd checkinstall-1.6.0beta4
checkinstall-1.6.0beta4 $ make
[...]
checkinstall-1.6.0beta4 $ su
Password:
checkinstall-1.6.0beta4 # make install
[...]
checkinstall-1.6.0beta4 # checkinstall

checkinstall 1.6.0beta4, Copyright 2002 Felipe Eduardo Sanchez Diaz Duran
This software is released under the GNU GPL.

Please choose the packaging method you want to use.
Slackware [S], RPM [R] or Debian [D]? R

*************************************
```

```
*** RPM package creation selected ***
**************************************

This package will be built according to these values:

1 -  Summary:  [ CheckInstall installations tracker, version 1.6.0beta4 ]
2 -  Name:     [ checkinstall ]
3 -  Version:  [ 1.6.0beta4 ]
4 -  Release:  [ 1 ]
5 -  License:  [ GPL ]
6 -  Group:    [ Applications/System ]
7 -  Architecture: [ i386 ]
8 -  Source location: [ checkinstall-1.6.0beta4 ]
9 -  Alternate source location: [  ]
10 - Provides: [ checkinstall ]
11 - Requires: [  ]

Enter a number to change any of them or press ENTER to continue:

Installing with make install...

========================= Installation results ==========================
[...]



========================= Installation successful =======================

Copying documentation directory...
./
./NLS_SUPPORT
./README
./FAQ
./TODO
./CREDITS
./INSTALL
./Changelog
./BUGS
./installwatch-0.7.0beta4/
./installwatch-0.7.0beta4/README
./installwatch-0.7.0beta4/TODO
./installwatch-0.7.0beta4/VERSION
./installwatch-0.7.0beta4/INSTALL
./installwatch-0.7.0beta4/CHANGELOG
./installwatch-0.7.0beta4/BUGS
./installwatch-0.7.0beta4/COPYING
./RELNOTES
./COPYING

Copying files to the temporary directory...OK

Striping ELF binaries and libraries...OK

Compressing man pages...OK

Building file list...OK

Building RPM package...OK

NOTE: The package will not be installed

Erasing temporary files...OK

Writing backup package...OK
```

```
Deleting temp dir...OK

**************************************************************

Done. The new package has been saved to

/usr/src/redhat/RPMS/i386/checkinstall-1.6.0beta4-1.i386.rpm
You can install it in your system anytime using:

        rpm -i checkinstall-1.6.0beta4-1.i386.rpm

**************************************************************

checkinstall-1.6.0beta4 # cd /usr/src/redhat/RPMS/i386/
i386 # rpm -i checkinstall-1.6.0beta4-1.i386.rpm
i386 #
```

L'utilisateur d'une distribution basée sur Debian installera le paquet avec *dpkg −i*. Les utilisateurs de Slackware utiliseront *installpkg*.

En utilisant la fonction query de votre gestionnaire de paquet, en l'occurence ici RPM, vous pouvez vérifier si le paquet a été proprement installé dans la base de données du gestionnaire et voir les champs d'information complémentaires de l'en−tête du paquet.

```
$ rpm -qi checkinstall
Name        : checkinstall            Relocations: (not relocatable)
Version     : 1.6.0beta4              Vendor     : (none)
Release     : 1                       Build Date : Mo 06 Dez 2004 17:05:45 CET
Install Date: Di 07 Dez 2004 01:41:49 Build Host : deimos.neo5k.lan
Group       : Applications/System     Source RPM : checkinstall-1.6.0beta4-1.src.rpm
Size        : 264621                  License    : GPL
Signature   : (none)
Packager    : checkinstall-1.6.0beta4
Summary     : CheckInstall installations tracker, version 1.6.0beta4
Description :
CheckInstall installations tracker, version 1.6.0beta4

CheckInstall  keeps  track of all the files created  or
modified  by your installation  script  ("make install"
"make install_modules",  "setup",   etc),   builds    a
standard   binary   package and  installs  it  in  your
system giving you the ability to uninstall it with your
distribution's  standard package management  utilities.
```

Configuration

Vous pouvez modifier le fichier texte */usr/lib/local/checkinstall/checkinstallrc*, suffisamment commenté, en vue de changer le comportement par défaut de *CheckInstall*.

Du fait que *CheckInstall* vous demande quel type de paquet doit être engendré, à chaque utilisation, il est plus judicieux de reprendre cette valeur de manière permanente dans la variable *INSTYPE*. Il est également utile de s'attarder sur les variables *INSTALL*, *PAK_DIR* et *RPM_FLAGS* en alternative de *DPKG_FLAGS*. Avec les deux dernières variables, vous pouvez définir quelques « flags (drapeau) » d'installation optionnels et en modifiant *PAK_DIR*, vous pouvez déterminer un autre répertoire qui contiendra la copie du paquet. *INSTALL* vous permet de décider entre simplement générer le paquet ou l'installer.

```
$ cat /usr/lib/local/checkinstall/checkinstallrc

######################################################################
                #    CheckInstall configuration file   #
                ########################################

######################################################################
# These are default settings for CheckInstall, modify them as you  #
# need. Remember that command line switches will override them.    #
######################################################################

# Debug level
#   0: No debug
#   1: Keep all temp files except the package's files
#   2: Keep the package's files too

DEBUG=0

# Location of the "installwatch" program
INSTALLWATCH_PREFIX="/usr/local"
INSTALLWATCH=${INSTALLWATCH_PREFIX}/bin/installwatch

# Location of the makepkg program. "makepak" is the default, and is
# included with checkinstall. If you want to use Slackware's native "makepkg"
# then set this to "makepkg"

MAKEPKG=/sbin/makepkg

# makepkg optional flags. These are recommended if running a newer Slackware
# version: "-l y -c n"

MAKEPKG_FLAGS="-l y -c n"

# Is MAKEPKG running interactively? If so, you might want
# to see what it's doing:

SHOW_MAKEPKG=0

# Where will we keep our temp files?
BASE_TMP_DIR=/var/tmp   ##  Don't set this to /tmp or / !!

# Where to place the installed document files
DOC_DIR=""

# Default architecture type (Leave it empty to allow auto-guessing)
ARCHITECTURE=""

# Default package type. Leave it empty to enable asking everytime
#   S : Slackware
#   R : RPM
#   D : Debian

INSTYPE="R"

# Storage directory for newly created packages
# By default they will be stored at the default
# location defined for the package type

PAK_DIR=""

# RPM optional flags
RPM_FLAGS=" --force --nodeps --replacepkgs "

# dpkg optional flags
```

```
DPKG_FLAGS=""


## These are boolean. Set them to 1 or 0

# Interactively show the results of the install command (i.e. "make install")?
# This is useful for interactive installation commands
SHOW_INSTALL=1

# Show Slackware package installation script while it runs? Again, useful if
# it's an interactive script
SHOW_SLACK_INSTALL=0

# Automatic deletion of "doc-pak" upon termination?
DEL_DOCPAK=1

# Automatic deletion of the spec file?
DEL_SPEC=1

# Automatic deletion of "description-pak"?
DEL_DESC=1

# Automatically strip all ELF binaries?
STRIP_ELF=1

# Automatically strip all ELF shared libraries?
# Note: this setting will automatically be set to "0" if STRIP_ELF=0
STRIP_SO_ELF=1

# Automatically search for shared libraries and add them to /etc/ld.so.conf?
# This is experimental and could mess up your dynamic loader configuration.
# Use it only if you know what you are doing.
ADD_SO=0

# Automatically compress all man pages?
COMPRESS_MAN=1

# Set the umask to this value
CKUMASK=0022

# Backup files overwritten or modified by your install command?
BACKUP=1

# Write a doinst.sh file that installs your description (Slackware)?
AUTODOINST=1

# Are we going to use filesystem translation?
TRANSLATE=1

# Reset the owner/group of all files to root.root?
RESET_UIDS=0

# Use the new (8.1+) Slackware description file format?
NEW_SLACK=1

# Comma delimited list of files/directories to be ignored
EXCLUDE=""

# Accept default values for all questions?
ACCEPT_DEFAULT=0

# Use "-U" flag in rpm by default when installing a rpm package
# This tells rpm to (U)pdate the package instead of (i)nstalling it.
RPM_IU=U
```

```
# Inspect the file list before creating the package
CK_INSPECT=0

# Review the .spec file before creating a .rpm
REVIEW_SPEC=0

# Review the control file before creating a .deb
REVIEW_CONTROL=0

# Install the package or just create it?
INSTALL=0
```

Conclusion

CheckInstall est un outil remarquable qui facilite grandement l'administration d'un système Linux. Particulièrement si des programmes doivent être fréquemment compilés depuis leurs sources, *CheckInstall* vous donne la possibilité de les enlever proprement sans risque de rendre le système incohérent. Par ailleurs, vous pouvez aussi installer ces paquets sur d'autres machines sans avoir à re−compiler le programme à chaque fois, évidemment en prenant en considération de possibles dépendances de paquet. Cependant, ceci n'est pas un gros problème pour les machines identiques.

Links

[1] http://asic−linux.com.mx/~izto/checkinstall/ [Page d'accueil de CheckInstall]
[2] http://www.gnu.org/software/autoconf/manual/autoconf−2.57/autoconf.html [Manuel en ligne de GNU Autoconf]
[3] http://asic−linux.com.mx/~izto/checkinstall/installwatch.html [Installwatch]
[4] http://asic−linux.com.mx/~izto/checkinstall/download.php [Téléchargement de CheckInstall]
[5] http://asic−linux.com.mx/~izto/checkinstall/docs/RELNOTES [Release Notes]
[6] http://asic−linux.com.mx/~izto/checkinstall/docs/Changelog [Changelog]

2005−01−22, generated by lfparser_pdf version 2.51