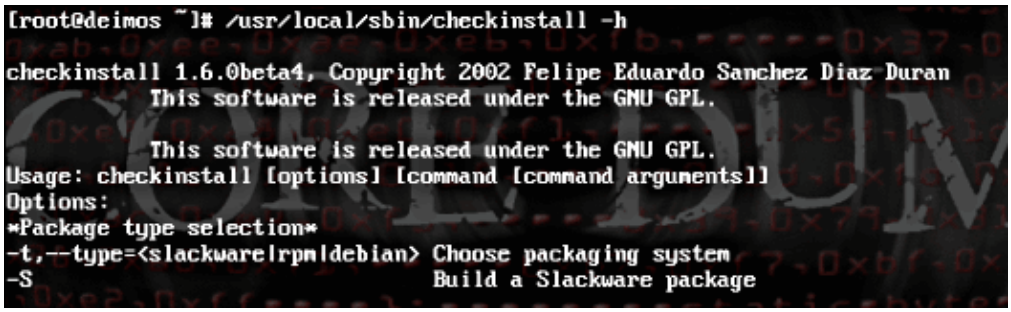
Mario M. Knopf
(homepage)

L'autore:

A Mario piace lavorare con Linux, le reti e gli argomenti relativi alla sicurezza. .

Tradotto in Italiano da:
Roberto Pauletto
<neverquit/at/cwazy.co.uk>

CheckInstall



Premessa:

Checkinstall e' una utility per costruire automaticamente pacchetti RPM/Debian o Slackware da sorgenti in tar.gz. Questo consente di eseguire una installazione e disinstallazione "pulita" di quasi tutti i pacchetti di codice sorgente in tar.gz.

Introduzione

Spesso accade che un programma che qualcuno vorrebbe provare e' disponibile solo attraverso codice sorgente in tar.gz (non e' disponibile nessun pacchetto rpm o Debian). In questo caso, occorre scaricare il pacchetto di codice sorgente, decomprimerlo e compilarlo manualmente. Fino qui tutto bene. Comunque, cosa succede quando volete cancellare il programma?.

Makefile contiene in pochissimi casi una routine appropriata per disinstallare il programma. Naturalmente esiste la possibilita' di installare il programma in una directory temporanea e tenere nota dell'elenco di tutti i file che sono stati creati o modificati dall'installazione per una futura rimozione. Ma questa procedura e' disagevole e dispendiosa, se i programmi sono compilati dalla sorgente con frequenza. Lo strumento *CheckInstall* [1] scritto da Felipe Eduardo Sánchez Díaz Durán risolve questo problema.

Di norma si compila e si installa un programma compatibile con *GNU Autoconf* tramite l'esecuzione della ben nota sequenza di comandi
./configure && make && make install.
Lo script di shell *configure* tenta di identificare i corretti valori per le diverse variabili che dipendono dal sistema, che successivamente saranno utilizzati durante la compilazione. Verifica che tutti i requisiti per la compilazione siano rispettati, quindi usa questi valori per creare un *Makefile* in ciascuna directory del pacchetto. Successivamente lo script *configure* genera dei file addizionati. Riepilogando essi sono:

- Uno o piu' *Makefile* in ogni directory/sottodirectory
- Uno script di shell chiamato *config.status*
- un file di testo *config.log*

- Un altro script di shell comunemente chiamato *config.cache* (opzionale)
- Diversi file header di C (*.h) con definizioni specifiche al sistema (opzionale)

Dopo che lo script di shell *configure* e' stato completato con successo, si digita *make* per compilare il pacchetto. Questo genera il binario eseguibile. Esiste la possibilita' di lanciare immediatamente dopo *make* un qualche auto test con *make check*. Ma questo e' un passo opzionale, visto che il pacchetto stesso deve supportare questo processo. Se *make* ha eseguito il suo compito, si puo' installare il programma compilato con il comando *make install* – per ovvie ragioni occorre detenere diritti privilegiati per eseguire questo passo. Dopo che il programma e' stato installato con successo, si puo' eliminare il programma binario ed i file oggetto dalla directory sorgente del codice digitando *make clean*. Se si preferisce cancellare anche i file creati da *configure*, si digita *make distclean*. Comunque gli ultimi due passi sono, proprio con *make check*, opzionali e sono usati in genere dallo sviluppatore durante lo sviluppo ed il test. Possono essere anche usati dall'utente per recuperare spazio su disco oppure per mantenere in ordine la struttura della directory. Inoltre *make distclean* consente la compilazione del pacchetto per un diverso tipo di computer.

Informazioni dettagliate a proposito di *GNU Autoconf* si possono trovare nel Manuale in linea in [2]. Oltre ad una introduzione base, potete imparare molto di piu' circa il *GNU Build System* creando i vostri script di *configure*, programmando in *M4* e creando le vostre macros, programmando il codice di shell etc.

CheckInstall

Come precedentemente detto, la sequenza di comandi per compilare un programma dai propri sorgenti compatibile con *GNU Autoconf* e':

./configure && make && make install

E' giunto il momento di usare *CheckInstall*. Che rimpiazzera' *make install* con il suo proprio comando *checkinstall*. Le altre due istruzioni rimangono invariate e sono generalmente usate come prima. Quindi la nuova sequenza di comandi con *CheckInstall* ora e':

./configure && make && checkinstall

Comunque l'istruzione *checkinstall* non fa mai partire *make install* per default e tiene traccia di tutti gli eventi di scrittura che sono eseguiti durante l'installazione A questo scopo *CheckInstall* usa il programma *Installwatch* [3], che originariamente fu scritto da Pancrazio de Mauro. Dopo che *make install* e' terminato con successo, *CheckInstall* genera un pacchetto Slackware–, Debian– o RPM e lo installa con il gestore di pacchetti di default della distribuzione, lasciando una copia del pacchetto nella corrente directory dei sorgenti, oppure in una directory standard di salvataggio. Inoltre e' possibile cambiare la directory di salvataggio di default tramite la variabile di salvataggio di default *PAK_DIR* all'interno del file di configurazione. La copia cosi' riferita potra essere installata, naturalmente tenendo conto delle possibili dipendenze dei pacchetti, su altre macchine nella rete senza compilare il pacchetto sorgente ancora una volta.

CheckInstall non solo si serve di *make install*, ma interagisce anche con altre istruzioni di installazione. Se lo script di installazione e' ad esempio:

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

Inoltre c'e' la possibiilita' di lanciare *CheckInstall* con diverse opzioni. Il seguente comando stampa un completo riepilogo di tutte le opzioni disponibili che, a sua volta, viene diviso nelle sezioni *Install Options (Opzioni di Installazione)*, *Scripting options (Opzioni di scripting)* , *Info display options (Opzioni di*

visualizzazione informazioni), *Package tuning options (Opzioni di messa a punto del pacchetto)*, *Cleanup options (Opzioni di pulizia)* e *About CheckInstall (Info su CheckInstall)*:

checkinstall –h

Se *CheckInstall* viene lanciato con una di queste opzioni, verranno ignorati i valori delle opzioni medesime all'interno del file di configurazione *checkinstallrc*.

Anche *CheckInstall* ha dei limiti. Non puo' gestire programmi linkati in modo statico, visto che *Installwatch* non e' in grado di monitorare i file modificati durante il processo di installazione. In generale, ci sono due tipi di librerie di programmi: statiche o dinamicamente linkate. Queste librerie sono integrate in un programma tramite una direttiva *include*–. I programmi linkati in modo statico hanno gia' tutte le funzioni di libreria necessarie e non le devono caricare in RAM in fase di esecuzione. Inoltre esse sono indipendenti dalle librerie effettivamente installate nel sistema di riferimento, perche' un cosiddetto *Linker* ha incorporato la libreria all'interno del programma eseguibile in fase di compilazione.

Installazione

CheckInstall e' da diverso tempo parte del pacchetto di software delle distribuzioni piu' vaste e puo' essere installato con il rispettivo sistema di gestione dei pacchetti. Se questo non fosse il caso per la vostra distribuzione, potete scaricare delle tarball o dei pacchetti precompilati che facciano al caso vostro dal sito web del progetto su [4].

L'installazione di *CheckInstall* e' piuttosto semplice e si sviluppa in pochi passi – ma per una riuscita installazione di *CheckInstall* dovrete paradossalmente avere installato *CheckInstall*. Dopo l'istruzione obbligatoria *make install* digitate *checkinstall*, che provvede a generare un appropriato pacchetto binario dal programma compilato. Ora potete installare questo pacchetto con il vostro gestore di pacchetti ed e' anche possibile disinstallarlo in modo pulito. Ma prima che *CheckInstall* crei il pacchetto, dovete dirgli quale gestore di pacchetti usate e verificare che le informazioni date siano corrette. Queste appariranno successivamente nell'intestazione del pacchetto.

La procedura per installare la nuova beta release *checkinstall–1.6.0beta4.tgz* viene spiegata nel successivo listato. Verra' installato *CheckInstall*, *Installwatch* e *makepak*, una versione modificata di *makepkg*. Se siete interessati alle modifiche della nuova versione, date un'occhiata alle *Note di rilascio* [5] e/o nel *Changelog* [6].

```
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 installaziones 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 #
```

Gli utenti delle distribuzioni basate su Debian installeranno il pacchetto con *dpkg −i*. Gli utenti di Slackware possono usare *installpkg* a questo scopo.

Usando la funzione di interrogazione del vostro gestore di pacchetti, in questo caso RPM, potete verificare se il pacchetto e' stato integrato correttamente nel suo database e visualizzare le informazioni addizionali dell'intestazione del pacchetto.

```
$ 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 installaziones tracker, version 1.6.0beta4
Description :
CheckInstall installaziones tracker, version 1.6.0beta4

CheckInstall  keeps  track of all the files created  or
modified  by your installazione  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.
```

Configurazione

Potete modificare il file di testo ben commentato */usr/lib/local/checkinstall/checkinstallrc* per modificare il comportamento di default di *CheckInstall*.

Poiche' *CheckInstall* vi chiede ogni volta quale tipo di pacchetto dovra' essere prodotto, e' consigliabile aggiungere questo valore in modo permanente impostando *INSTYPE*. E' anche una buona idea cercare le variabili *INSTALL*, *PAK_DIR* e *RPM_FLAGS*oppure *DPKG_FLAGS*. Con le ultime due variabili potete definire alcune impostazioni di installazione opzionali, e modificando *PAK_DIR* potete specificare un'altra directory di salvataggio per la copia del pacchetto. *INSTALL* vi consente di scegliere se generare e/o installare il pacchetto.

```
$ 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 installazione commands
SHOW_INSTALL=1

# Show Slackware package installazione 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
```

Conclusione

CheckInstall e' uno strumento brillante che puo' rendere l'amministrazione di una Linux box molto piu' semplice. In particolare se i programmi devono essere frequentemente compilati dalle sorgenti, *CheckInstall* fornisce la possibilita' di rimuovere in modo pulito i programmi senza il rischio di avere un sistema inconsistente. Inoltre potete installare questi pacchetti anche su altre macchine senza dovere compilare ogni volta il programma – naturalmente occorre considerare eventuali dipendenze dei pacchetti. Comunque non si tratta di un problema particolarmente grave se le macchine sono identiche.

links

[1] http://asic-linux.com.mx/~izto/checkinstall/ [Home di CheckInstall]
[2] http://www.gnu.org/software/autoconf/manual/autoconf-2.57/autoconf.html [GNU Autoconf Online Manual (Manuale in linea)]
[3] http://asic-linux.com.mx/~izto/checkinstall/installwatch.html [Installwatch]
[4] http://asic-linux.com.mx/~izto/checkinstall/download.php [CheckInstall Downloads]
[5] http://asic-linux.com.mx/~izto/checkinstall/docs/RELNOTES [Note di rilascio]
[6] http://asic-linux.com.mx/~izto/checkinstall/docs/Changelog [Changelog]

Webpages maintained by the LinuxFocus Editor team © Mario M. Knopf "some rights reserved" see linuxfocus.org/license/ http://www.LinuxFocus.org	Translation information: de --> -- : Mario M. Knopf (homepage) de --> en: Mario M. Knopf (homepage) en --> it: Roberto Pauletto <neverquit/at/cwazy.co.uk>

2005-01-12, generated by lfparser_pdf version 2.51