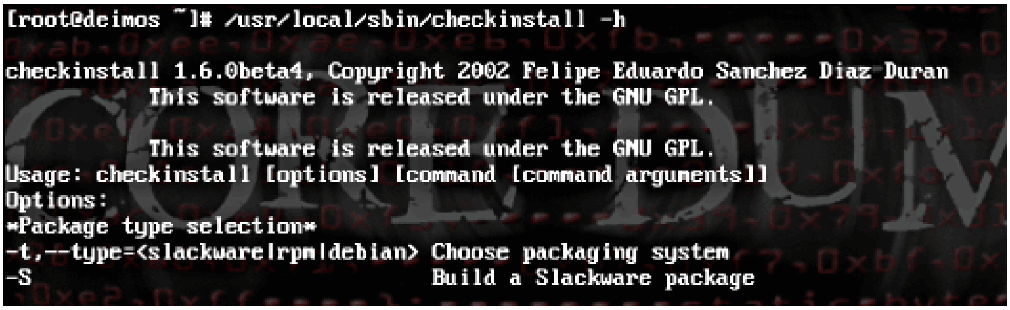
LinuxFocus article number 360
http://linuxfocus.org

by Mario M. Knopf
(homepage)

About the author:

Mario enjoys working with Linux, networks and security related topics.

CheckInstall



```
[root@deimos ~]# /usr/local/sbin/checkinstall -h

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

           This software is released under the GNU GPL.
Usage: checkinstall [options] [command [command arguments]]
Options:
*Package type selection*
-t,--type=<slackware|rpm|debian> Choose packaging system
-S                               Build a Slackware package
```

Abstract:

Checkinstall is utility to automatically build RPM/Debian or Slackware packages from tar.gz source pages. This makes it possible to do a "clean" installation and de–installation of almost any source code tar.gz package.

Translated to English by:
Mario M. Knopf
(homepage)

Introduction

It often happens that the program which one certainly would like to test, is only available as tar.gz source code (no rpm or Debian package available). In such a case, you download the source package, unpack it and compile it manually. So far, so good. However, what happens if you like to delete the program?

The *Makefile* contains only in very few cases an appropriate routine to uninstall the program. Of course there is the possibility of installing the program into a temporary directory and write down all of the files which were created or modified by the installation for a later removing. But this procedure is painful and very extensive, if programs are frequently compiled from the sources. The tool *CheckInstall* [1] written by Felipe Eduardo Sánchez Díaz Durán solves this problem.

As a rule, you compile and install a *GNU Autoconf* compliant program under use of the widely known command sequence
./configure && make && make install.
The *configure* shell script attempts to guess correct values for various system–dependent variables, which are used later during the compilation. It checks whether all requirements for compilation are fulfilled and uses those values to create a *Makefile* in each directory of the package. Furthermore the *configure* script generates additional files. In summary these are:

- one or more *Makefile(s)* in every directory/subdirectory
- a shell script called *config.status*
- a text file *config.log*

- another shell script normally called *config.cache* (optional)
- various C header files (*.h) with system−specific definitions (optional)

After the *configure* shell script has completed successfully, you type *make* to compile the package. This generates the executable binaries. There is the possibility to run immediately after *make* any self−tests with *make check*. But this is an optional step, because the package must support this process also. If *make* has performed its work, you can install the compiled program with the command *make install* − for obvious reasons, you need privileged user rights for this step. After the program was installed successfully, you can remove the program binaries and object files from the source code directory by typing *make clean*. If you prefer to delete also the files which created by *configure*, then type *make distclean*. However, the two last−named steps are, just like *make check*, optional and are normally used by the developer during development and testing. They can also be used by the user to save hard disk capacity or rather to keep the directory structure clean. Furthermore *make distclean* makes it possible to compile the package for a different kind of computer.

Detailed information about *GNU Autoconf* can be found in the Online−Manual on [2]. Besides a basic introduction, you can learn more about the *GNU Build System*, by making your own *configure* scripts, programming in *M4* and creating own macros, portable shell programming et cetera.

CheckInstall

As previously mentioned, the command sequence to build a *GNU Autoconf* compliant program from its sources is:

```
./configure && make && make install
```

At this point it is time to use *CheckInstall*. This is replacing *make install* with its own command *checkinstall*. The two other instructions remain unaffected by this and are normally used as before. So the new command sequence with *CheckInstall* now is:

```
./configure && make && checkinstall
```

However, the instruction *checkinstall* nevertheless starts *make install* by default and monitors all writing events which are done by the installation. For this purpose *CheckInstall* uses the program *Installwatch* [3], which was originally written by Pancrazio de Mauro. After *make install* has successfully completed, *CheckInstall* generates a Slackware−, Debian− or RPM−Package and installs it with distribution's default package manager and leaves a copy of the package in the current source directory or in the standard storage directory. Moreover, it is possible to change the default storage directory with the variable *PAK_DIR* inside the configuration file. The addressed copy can be installed, of course under consideration of possible package dependencies, on other machines in the network without compiling the source package every time again.

CheckInstall does not only make use of *make install*, but cooperates also with other installation instructions. If the installation script is for example *setup.sh* the instruction set is:

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

Furthermore there is the possibility of starting *CheckInstall* with various options. The following command prints a complete overview of all available options which in turn are subdivided into the sections *Install options*, *Scripting options*, *Info display options*, *Package tuning options*, *Cleanup options* and *About CheckInstall*:

```
# checkinstall −h
```

If *CheckInstall* is started with one of these options, it will override the values inside the configuration file *checkinstallrc*.

But also *CheckInstall* has limits. It can't handle statically linked programs, because *Installwatch* is not able to watch the modified files during the installation process. In general there are two types of program libraries: static or dynamic linked libraries. These libraries are integrated into a program by an *include*−directive. Statically linked programs already have all necessary library functions and must not load these into the RAM at run time. Furthermore they are independent of the libraries actually installed on the target system, because a so−called *Linker* has embedded the library into the program executable at compile time.

Installation

CheckInstall is already for a quite long time a part of the software pools of larger distributions and can be installed with the respective packet management system. If this shouldn't be the case, you can download suitable tar−balls or prebuild packages for various distributions from the project−website on [4].

The installation of *CheckInstall* is quite simple and runs in few steps − but for a successful installation of *CheckInstall* you will paradoxically need *CheckInstall*. After the obligatory instruction *make install* you type *checkinstall*, which generates a appropriate binary−package from the compiled program. Now you can install this package with your package manager and it is also possible to uninstall the program cleanly. But before *CheckInstall* creates the package, you must answer a question about your used package manager and check the information fields for correctness. Those appear in the package header later.

The procedure to install the new beta release *checkinstall−1.6.0beta4.tgz* is demonstrated in the following code−box. This will install *CheckInstall*, *Installwatch* and *makepak*, a modified version of *makepkg*. If you are interested in the changes of the new version, take a look in the *Release Notes* [5] and/or in the *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 #
```

User of Debian based distributions install the package with *dpkg -i*. Slackware users can use *installpkg* for this purpose.

By using the query function of your package manager, in this case RPM, you can check whether the package was cleanly integrated in its database and view the additional information field of the package header.

```
$ 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

You can modify the well commented text file */usr/lib/local/checkinstall/checkinstallrc* to change *CheckInstall's*default behavior.

Since *CheckInstall* asks you at every start which package type shall be produced, it is wise to add this value permanently by setting *INSTYPE*. It is also a good idea to look at the variables *INSTALL*, *PAK_DIR* and *RPM_FLAGS* alternatively *DPKG_FLAGS*. With the last two variables you can define some optional install flags, and with modifying *PAK_DIR* you can specify another storage directory for the package copy. *INSTALL* lets you decide between only generating the package and/or installing it.

```
$ 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 is a brilliant tool which can make the administration of a Linux box a lot easier. Particularly if programs must be frequently compiled from its sources, *CheckInstall* gives you the possibility to cleanly remove the programs without the risk of an inconsistent system. Furthermore you can install this packages also on other machines without having to compile the program every time again – of course you must consider possible package dependencies. However, this usually isn't a too big problem for identical machines.

Links

[1] http://asic−linux.com.mx/~izto/checkinstall/ [Home of CheckInstall]
[2] http://www.gnu.org/software/autoconf/manual/autoconf−2.57/autoconf.html [GNU Autoconf Online Manual]
[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 [Release Notes]
[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)

2005−01−15, generated by lfparser_pdf version 2.51