Out of the box

SAY HELLO WAVE GOODBYE

The latest software under Linux is often only available in the form of a tar archive. As Christian Perle explains, installing and removing tarballs can be made much simpler with checkinstall

Source text The form of any software which is legible and alterable by humans. By compiling it with a compiler this is turned into an executable program.

Library A library contains a collection of useful C functions for specific purposes. So, for example, there is the libm, which provides mathematical functions, or the libXt, containing functions for programming the Xwindow system. Often libraries are used jointly ("shared") by a number of programs.

The three-step process: *./configure; make; make install*, will be familiar to anyone who has ever installed a program from its **source text** form. But very few packages support neat uninstallation of the files copied into the file system with *make install*. This is where *checkinstall* by Felipe Eduardo Sanchez Diaz Duran comes in.

When the installwatch **library** is used, the program monitors all write-actions performed when using *make install* or a corresponding installation command. Doing so it records a list of the new files and directories.

Chicken or egg?

If you have installed the GNU-C compiler by hand and the development package necessary for compilation *glibc-dev* (depending on the distribution it might be called something slightly different), you now only need to take a virtual journey to Mexico in order to get hold of the sources for *checkinstall* from *http://proyectos.glo.org.mx/checkinstall/*.

It sounds a bit mad, but to install checkinstall (the program) you do need *checkinstall* (the command). Naturally there's a trick to this: the program is first installed with the usual command *make install*. After that, the new command *checkinstall* is available, with which you repeat the installation:

```
tar xzf checkinstall-1.5.1.tgz
cd checkinstall-1.5.1
make
su (enter root password)
make install
checkinstall
exit
```

Before the new tool is installed as a package, *checkinstall* wants to know a few things about which **package manager** is normally used on the system. On the system in Figure 1, Debian GNU/Linux is used, so the option *d* would be your choice. In the following menu *checkinstall* shows various data fields on the package, most of which are filled with meaningful figures. You can change the content of a field via the corresponding numbers.

Finally *checkinstall* informs you how you can get rid of the package just installed. In the example, the command reads *dpkg –r checkinstall*.

Out of the box

There are thousands of tools and utilities for Linux. "Out of the box" takes the pick of the bunch and each month suggests a little program, which we feel is either absolutely indispensable or unduly ignored.

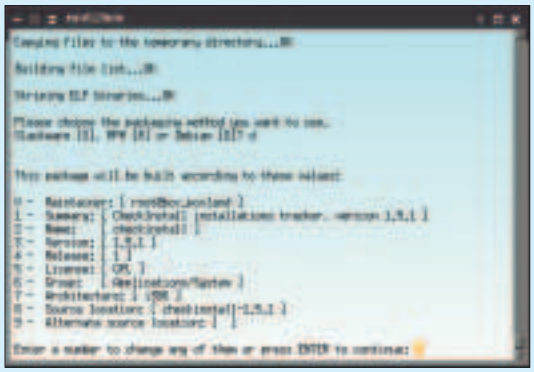

Figure 1: Selecting the package type

Trickery

However, we have not installed *checkinstall* just to remove it again immediately. This treatment should be used though, for any programs in future software packages that prove to be unstable or of too little use. But how does *checkinstall* actually work?

The library *installwatch* replaces all file functions of the Standard-C library with its own. Checkinstall now uses the **Preload** mechanism to give the functions from *installwatch* precedence over the "true" functions. The installed, functions make a note of all write actions and transfer the file list thus obtained to *checkinstall*. This in turn deploys the selected package manager to create from the list a Slackware, Red Hat or Debian package. Finally the freshly constructed package is installed with the distribution's own package manager.

Baking Red Hats

How does this look in practice? As an example let's pick the installation of the fractal generator XaoS, introduced in a previous issue of "Out of the box". Once its source text is unpacked with *tar –xzvf xaos-3.0.tar.gz*, the following steps are taken:

```
cd XaoS-3.0
./configure
make
su (enter root password)
checkinstall
```

After the usual steps of *./configure* and *make*, instead of *make install* you invoke the command *checkinstall*. To the question about the package type, users of rpm-based distributions select *r*, use *6* to change the group to *Applications/Graphics* and add, with *1*, a brief description, for example *realtime fractal zooming for X and console*.

Figure 2: Installation as rpm package

Once installation is complete, uninstallation is done by *rpm –e XaoS-3.0-1* (Figure 2). Also, the program files away the created *rpm* package for later installations in the directory */usr/src/rpm/RPMS/i386*.

So after removal, it is possible at any time to play back in the package with *rpm –i XaoS-3.0-1.i386.rpm* as new. A similar thing applies, too, for Debian packages, except they are filed, not in the */usr/src* hierarchy, but in the current source directory.

Figure 3 shows the rpm database queried for control purposes. With *rpm –qi XaoS* (the version number can be left out for installed packages) the package manager displays the information defined by us or automatically generated about the XaoS package.

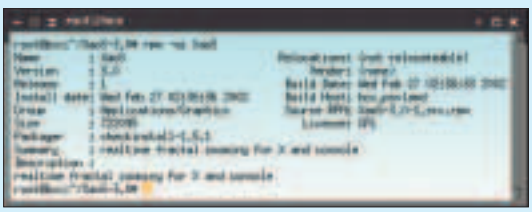

Figure 3: rpm displays the installed package

Standards and specials

Normally one would only use a single package format in a Linux system. But this means it soon becomes a chore to keep querying the package type through *checkinstall*. Fortunately, the program uses a file with standard settings, which you can adapt as you wish. You can find this configuration file at */usr/local/lib/checkinstall/checkinstallrc*. It is well commented and in addition to selection of the package type (*INSTYPE*) also allows you to define a directory in which to save the created packages (*PAK_DIR*) or additional options for the various package managers (*RPM_FLAGS*, *DPKG_FLAGS*).

Checkinstall also includes a range of command line options when invoked. One which has proven very useful is *–si*. This option allows it to supervise interactive mechanisms, which additionally require user inputs during installation. The complete documentation on *checkinstall* can be found, by the way, at */usr/doc/checkinstall-1.5.1/README*.

With many distributions it is good form that the documentation on a package is installed in */usr/doc/Packagename*. Checkinstall tries to keep to this and when invoked it searches the source directory of the directory *doc-pak*. If it exists, its content will be copied during the *checkinstall* run to */usr/doc/Packagename* – and obviously also removed during uninstallation.

Limits

Obviously, every tool has its limits. So *checkinstall* cannot monitor the file accesses of a **static-linked** installation program, because the preload mechanism does not function here. The same restriction applies to programs which, after starting, run with the rights of another user. Such programs are prohibited from preloading on security grounds.

For the future, the author is planning better menu control via dialog boxes, a manpage for quick reference to the options and the cryptographic signing of the created packages. There are also great expectations of an upcoming version 2.0.

Info

XaoS
http://www.gnu.org/software/xaos/xaos.html
Patricia Jung, "Installation without tears", *Linux Magazine*, Issue 18, pages 68-73.

Package manager A management program for smooth installation and uninstallation of program packages. Common package managers are rpm (used by Red Hat, but also SuSE, Mandrake and Caldera) and the Debian package manager dpkg.

Preload If the environment variable LD_PRELOAD is added to the name of a library file, then all the symbols of this library take precedence over those of the libraries loaded later. So for example the C-library function *printf* (formatted output) is replaced by its own version.

Static-linked A static-linked program contains all the necessary functions from the libraries used and no longer needs to read these in at run time. The advantage is the independence from the installed libraries, and the drawback is a considerably larger program file.