Installing GNU Guix 0.6, easily

Arne Babenhauserheide

May 17, 2014

> *"Got a power-outage while updating? No problem: Everything still works"*

GNU Guix is the new functional package manager from the GNU Project which complements the Nix-Store with a nice Guile Scheme based package definition format.

What sold it to me was "Got a power-outage while updating? No problem: Everything still works" from the Guix talk of Ludovico at the GNU Hacker Meeting 2013. My son once found the on-off-button of our power-connector while I was updating my Gentoo box. It took me 3 evenings to get it completely functional again. This would not have happened with Guix.

Installation is straightforward, except if you follow the docs, but it's not as if we're not used to that from other GNU utilities, which often terribly short-sell their quality with overly general documentation :)

So I want to provide a short guide how to setup and run GNU Guix with ease. My system natively runs Gentoo, My system natively runs Gentoo, so some details might vary for you. *If you use Gentoo, you can simply copy the commands here into the shell, but better copy them to a text-file first to ensure that I do not try to trick you into doing evil things with the root access you need.*

Update (2014-05-17): Thanks to zerwas from IRC @ freenode for the patch to guix 0.6 and nice cleanup!

Getting GNU Guix

```
mkdir guix && cd guix
wget http://alpha.gnu.org/gnu/guix/guix-0.6.tar.gz
wget http://alpha.gnu.org/gnu/guix/guix-0.6.tar.gz.sig
gpg --verify guix-0.?.tar.gz.sig
```

Installing GNU Guix

```
tar xf guix-0.?.tar.gz
cd guix-0.?
./configure && make -j16
sudo make install
```

Setting up GNU Guix

Build users

Build-users allow for strong separation of build processes: They cannot affect each other, because they actually run as different users.

```
sudo screen
groupadd guix-builder
for i in `seq 1 10`;
  do
    useradd -g guix-builder -G guix-builder           \
            -d /var/empty -s `which nologin`          \
            -c "Guix build user $i" --system          \
            guix-builder$i;
  done
exit
```

(if you do not have GNU screen yet, you should get it. It makes working on remote servers enjoyable.

Add user work folder.

Also we want to run guix as regular user. We need to pre-create the user-specific build-directory. Note: This should really be done automatically.

```
sudo mkdir -p /usr/local/var/nix/profiles/per-user/$USER
sudo chown -R $USER:$USER /usr/local/var/nix/profiles/per-user/$USER
```

Fix store permissions

```
chgrp 1002 /nix/store; chmod 1775 /nix/store
```

Starting the guix daemon and making it launch at startup

this might be quite Gentoo-specific.

```
sudo screen
echo "#\!/bin/sh" >> /etc/local.d/guix-daemon.start
echo "guix-daemon --build-users-group=guix-builder &" >> /etc/local.d/guix-daemon.start
echo "#\!/bin/sh" >> /etc/local.d/guix-daemon.stop
echo "pkill guix-daemon" >> /etc/local.d/guix-daemon.stop
chmod +x /etc/local.d/guix-daemon.start
chmod +x /etc/local.d/guix-daemon.stop
exit
```

(the pkill is not the nice way of killing the daemon. Ideally the daemon should have a –kill option)

To avoid having to restart, we just launch the daemon once, now.

```
sudo /etc/local.d/guix-daemon.start
```

Adding the guix-installed programs to your PATH

Guix installs each state of the system in its own directory, which actually enables roll-backs. The current state is made available via ~/.guix-profile/, and so we need ~/.guix-profile/bin in our path:

```
echo "export PATH=$PATH:~/.guix-profile/bin" >> ~/.bashrc
. ~/.bashrc
```

Using guix

Guix comes with a quite complete commandline interface. The basics are

- Update the package listing: guix pull
- List available packages: guix package -A
- Install a package: guix package -i PACKAGE
- Update all packages: guix package -u

Experience

For a new distribution-tool, Guix is quite nice. Remember, though, that it builds on Nix: It is not a complete reinvention but rather "stands on the shoulders of giants".

The download speeds are abysmal, though. `http://hydra.gnu.org` seems to have a horribly slow internet connection...

And what I direly missed is a short command explanation in the help output:

```
$ guix --help
Usage: guix COMMAND ARGS...
Run COMMAND with ARGS.

COMMAND must be one of the sub-commands listed below:

   build
   download
   gc
   hash
   import
   package
   pull
   refresh
   substitute-binary
```

Also I miss the categories I know from Gentoo: Having package-names like grue-hunter seems very unorganized compared to the games-text/grue-hunter which I know from Gentoo.

And it would be nice to have shorthands for the command names:

- "guix pa -i" instead of "guix package -i" (though there is a namespace clash with guix pull :( )
- "guix pu" for "guix pull"

and so on.

But anyway: A very interesting project which I plan to keep tracking. It might allow me to do less risky local package installs of stuff I need, like small utilities I wrote myself.

The big advantage of that would be, that I could actually take them with me when I have to use different distros (though I've been a happy Gentoo user for ~10 years and I don't see it as likely that I'll switch completely: Guix would have to include all the roughly 30k packages in Gentoo to actually be a full-fledged alternative - and provide USE flags and all the convenient configurability which makes Gentoo such a nice experience).

Using guix for such small stuff would allow me to decouple experiments from my production environment (which has to keep working).

But enough talk: Have fun with GNU Guix and Happy Hacking!