reproducible and user-controlled software management in HPC

with GNU Guix

Ricardo Wurmus + Pjotr Prins + Ben Woodcroft + Altuna Akalin
BOSC 2017, Prague
System Admins
prefer **mature** software, no variants, only apply unavoidable updates

Users
want **fresh** software, multiple variants, latest tools, **flexibility**

use **stable** software for **systems**

manage **user software** stack on their own
There is no way to reproduce the environment, even on the same machine at a different point in time.

Due to a lack of isolation, the environment will change or even break when the host system changes.

No safe upgrades or roll-backs. No separation for different workflows. Unportable.
Installing software is easy now!

...but reproducibility is still out of reach

boegel opened this issue on Nov 5, 2013 · 0 comments

boegel commented on Nov 5, 2013

It seems like the GCC libraries (e.g. *libiberty.a*) sometimes end up being built with `-fPIC` (e.g. on SL5), and sometimes not (e.g. on SL6), while `eb` is performing the exact same build procedure.

This causes problems for cairo (see) and ExtraE (part of UNITE), which require *libiberty.a* to be built with `-fPIC`. The cairo builds works fine on SL5, but doesn't work on SL6 (see also [hpcugent/easybuild-easyconfigs#494](https://github.com/hpcugent/easybuild-easyconfigs/issues/494) (comment)).
Installing software is easy now!

...but reproducibility is still out of reach

Good news, I ran into this problem too. But only on SOME computers... I don't yet know why some but not all. Anyway... look in the generated `spconfig.py` files, I see the following:

```python
env['PATH'] = "":".join(cmdlist"
   /gpfs/dnb53/rpfische/spack3/opt/spack/linux-SuSE11-x86_64/gcc-5.3.0/cmake-3.6.1-xfz
   /gpfs/dnb53/rpfische/spack3/opt/spack/linux-SuSE11-x86_64/gcc-5.3.0/python-3.5.2-d5i
   /gpfs/dnb53/rpfische/spack3/opt/spack/linux-SuSE11-x86_64/gcc-5.3.0/netcdf-cxx4-4.3.
   /gpfs/dnb53/rpfische/spack3/opt/spack/linux-SuSE11-x86_64/gcc-5.3.0/py-numpy-1.11.1-
   /gpfs/dnb53/rpfische/spack3/opt/spack/linux-SuSE11-x86_64/gcc-5.3.0/udunits2-2.2.20-
   /gpfs/dnb53/rpfische/spack3/opt/spack/linux-SuSE11-x86_64/gcc-5.3.0/proj-4.9.2-f6543
   /gpfs/dnb53/rpfische/spack3/lib/spack/env
   /gpfs/dnb53/rpfische/spack3/lib/spack/env/case-insensitive
   /gpfs/dnb53/rpfische/spack3/lib/spack/env/gcc
   /gpfs/dnb53/rpfische/spack3/opt/spack/linux-SuSE11-x86_64/gcc-5.3.0/binutils-2.27-vd
   /home/rpfische/git/modele-control/bin
   /usr/local/other/SLES11.3/openmpi/1.10.1/gcc-5.3.0/bin
   /usr/local/other/SLES11.3/gcc/5.3.0/bin
   /usr/local/other/SLES11.3/git/2.7.4/libexec/git-core
   /usr/local/other/SLES11.3/git/2.7.4/bin
```
There can be non-trivial issues with glibc that make it hard to distribute truly portable packages. I've been in situations where I couldn't easily deal with the range of Linux distributions that my colleagues were using.

After activating an environment `libselinux.so.1: cannot open shared object file` #5640

Nagasaki45 opened this issue 16 days ago - 1 comment

Nagasaki45 commented 16 days ago - edited

I have an environment.yml file as pasted below. When I create an environment from it, and activate it, I see some weird bugs:

```bash
(myenv) $ vim
vim: error while loading shared libraries: libselinux.so.1: cannot open shared object file:
```

libgfortran broken? #686

swryan opened this issue on Mar 7, 2016 - 35 comments

swryan commented on Mar 7, 2016

Started getting this error a couple days ago:

```python
ImportError: /home/travis/miniconda/lib/python2.7/site-packages/scipy/special/.../.../.../libgfortran.so.3: version GFORTRAN_1.4 not found (required by /usr/lib/libiapack.so.3gf)```
100% reproducible
We have all the bits!
100% reproducible
We have all the bits!

100% stateful
We only have the bits!
App bundles are convenient but lack means of abstraction. They don't compose well and are like giant statically linked binaries.
headers
sources
build tools
libraries
...

$\text{cabba9e-emacs-24.5/}$
- bin
  - Emacs
- lib
  - ...

$\text{f(x)}$
Functional packaging

Same inputs? Same output!

Different inputs? Different outputs.

cabba9e-emacs-24.5/
  └── bin
      └── Emacs

  └── lib
      └── ...

dedbeef-emacs-24.5/
  └── bin
      └── Emacs

  └── lib
      └── ...

Different inputs?
guix gc --references
/gnu/store/...-foo-0.9

/gnu/store/...-glibc-2.25
/gnu/store/...-gcc-4.9.3-lib
/gnu/store/...-bar-0.7b
/gnu/store/...-baz-1.4.9
/gnu/store/...-foo-0.9
guix package -i baz
   --with-input=foo=bar

guix package -i baz
   --with-source=baz-1.0.tgz
guix pack
samtools bedtools

/gnu/store/...-pack.tar.gz
guix pack -f docker
samtools bedtools

/gnu/store/
...-docker-pack.tar.gz

LOL
guix package
--manifest=GeneNetwork

All you need is:
  Guix version + package manifest
(+ source code)
1. The level of abstraction matters
2. Guix enables reproducible and safe experimentation
3. Guix makes environment sharing easy
4. There are ways to use Guix without root access
5. Like Conda?
   guix package -i conda
Learn more!

Poster A-142

#guix on irc.freenode.net

http://gnu.org/s/guix

ricardo.wurmus@mdc-berlin.de
pjotr.public34@thebird.nl