History — reproducible-builds.org

11-14 minutes

A history of *reproducible builds* in and around Debian.

An old idea

The idea of *reproducible builds* is not very new. It was implemented for GNU tools in the early 1990s (which we learned, much later in 2017). In Debian world, it was mentioned first in 2000, and then more explicitly in 2007 on `debian-devel`: "*I think it would be really cool if the Debian policy required that packages could be rebuild bit-identical from source.*" The reactions were unfortunately not really enthusiastic both times.

Private property + Snowden effect

The interest on reproducible builds picked up again with Bitcoin. Users of bitcoins needed a way to trust that they were not downloading corrupted software. Initial versions of Gitian were written in 2011 to solve the problem. It drives builds using virtual machines and Git.

The global surveillance disclosures in 2013 raised the interest even further. Mike Perry worked on making the Tor Browser build reproducibly in fear of a "*malware that attacks the software development and build processes themselves to distribute copies of itself to tens or even hundreds of millions of machines in a single, officially signed, instantaneous update*".

Kick-off

The success of making such a large piece of software build reproducibly proved that it was feasible for other projects. This prompted Lunar to organize a discussion at DebConf13 happening July 2013. Even scheduled at the last minute, there was still about thirty attendees who were very much interested, amongst them members of the technical committee and a few other core teams. Minutes are available.

After some more research during the conference, a wiki page was created. The initial approach was to get Debian to "buy-in" on the idea by making five packages from different maintainers build reproducibly. However, it quickly appeared that before fixing issues in the toolchain, it would not be possible to even get a single package to be reproducible.

First mass-rebuilds

Lunar came up with the first patches for DebianPts:dpkg during August 2013. This enabled DebianPts:hello to build reproducibly. The first large scale rebuild was performed soon after by David Suárez, with variations on time and build path. 24% of 5240 source packages were identified as reproducible. The first version of a "smart" comparison script was written to help reviewing differences.

A second mass rebuild was made before the presentation in the distro devroom at FOSDEM'14. It used a slightly different approach regarding build paths and had `binutils` built in deterministic mode. 67% of 6887 source packages were found reproducible. A result applauded by the FOSDEM crowd.

The presentation sparked interest and woke up the mailing-list created some months ago. Tomasz Buchert wrote a lintian check for gzip files. Stéphane Glondu worked on sorting logs and experimenting with alternatives for build path issues.

.buildinfo control files

In parallel, several approaches on where and how to record the build environment were considered. The first idea was to use the .changes control file through a substitution variable (Bug:719854). Instead, Guillem Jover suggested to add new fields by passing `--changes-option="-DBuild-Env=...` to `dpkg-buildpackage`. As for the value, we discovered `dh-buildinfo` written by Yann Dirson, described as a “debhelper addon to track package versions used to build a package”. Fit for reproducible builds!

What happened for a year was presented at DebConf14. The reception was unexpectedly good and the follow-up BoF truly productive. For one thing, a suitable way to record the build environment was sketched out.

One issue about using `.changes` files is that they are not kept in the archive. So to be used as a way to record the environment, they would need to be distributed with the archive. But this would be a misunderstanding of their purpose. As their name implies, `.changes` control files represent *changes* to archive. They were inherently designed to be transient.

So instead, we had the idea of a new `.buildinfo` control file which would be added to the archive alongside binary packages — and be uploaded by referencing them in `.changes`. We quickly drafted a specification, and a couple of days later Niko Tyni came up with an addition to debhelper which created a `.buildinfo` using the output of the aforementioned `dh-buildinfo`.

strip-nondeterminism

Before DebConf14, an explicit *timestamp* was given during rebuilds, extracted from the `.changes` file. However, during the discussions, there was a consensus that the date of the latest entry in the debian/changelog file could be used as the reference timestamp when needed.

This helped another idea: a generic tool that would post-process different file formats to remove timestamps or other source of non-determinism. Andrew Ayer further took the task of creating strip-nondeterminism. The first released version handled files created by gzip, Zip, Jar, Javadoc, and `.a` files.

Giving up on build paths

Initially we though that variations happening when building the package from different build path should be eliminated. This has proven difficult. The main problem that has been identified is that full path to source files are written in debug symbols of ELF files.

First attempt used the -fdebug-prefix-map option which allows to map the current directory to a canonical one in what gets recorded. But compiler options get written to debug file as well. So it has to be doubled with `-gno-record-gcc-switches` to be used for reproducibility. The first large scale rebuild has proven that it was also hard to determine what the actual build path has been accurately.

Second attempt used `debugedit` which is used by Fedora and other to change the source paths to a canonical location after the build. Unfortunately, `gcc` write debug strings in a *hashtable*. `debugedit` will not reorder the table after patching the strings, so the result is still unreproducible. Adding this feature to `debugedit` looked difficult. We can still make the approach work by passing `-fno-merge-debug-strings` but this is space expensive. The second large scale rebuild used the latter approach. It was still difficult to guess the initial build path properly. Stéphane Glondu was the first to suggest to using a canonical build path to solve the issue.

During discussions at DebConf14, we revisited the idea, and felt it was indeed appropriate to decide on a canonical build path. It has an added benefit of making it easier to use debug packages: one simply has to unpack the source in the right place, no extra configuration required.

Finally, it was agreed to add a `Build-Path` field to `.buildinfo` as it made it easier to reproduce the initial build if the canonical build location would change.

After strip-nondeterminism initial upload and integrating some more changes discussed during DebConf14 in DebianPts:dpkg and DebianPts:debhelper, Lunar experimented with 172 core packages. 30% were reproduced without further modifications.

As the current tools to understand differences between builds were slow and hard to read, Lunar wrote `debbindiff`. It replaced inefficient shell scripts by structured Python with a HTML output.

Continuous integration

At the end of September 2014, Holger Levsen started to work on jenkins.debian.net to perform continuous integration for build reproducibility. Packages from *sid* started to be built and rebuild. This initially introduced variations for time and file ordering, and was extended later on to also use different users, groups, hostnames, and locales.

The results were visible through a new reproducible.debian.net website. The process of analyzing reproducibility failures could now be more easily shared. New contributors indeed showed up and started submitting sorting out common issues and providing patches.

In July of 2015, Vagrant begins hosting ARM boards for reproducibility testing the armhf architecture. They were added to jenkins in August of 2015, and by December, nearly all packages on armhf had been tested at least once.

dpkg-genbuildinfo

The turn of 2015 saw the replacement of the prototype `.buildinfo` generator by a new implementation suitable for proper inclusion in `dpkg`. Previously, only packages using dh could generate `.buildinfo` and could thus be considered reproducible. After updating the experimental toolchain, the change allowed to reach the mark of 80% source packages reproducible.

FOSDEM 2015 and aftermath

The presentation Stretching out for trustworthy reproducible builds was well received at FOSDEM 2015 and was followed up by

- tracker.debian.org inclusion, see 739497
- Debian Developer's Packages Overview (DDPO) inclusion
- debbindiff gained .rpm support
- Debian Maintainer Dashbord inclusion

Finally, for now, not even two weeks after FOSDEM 2015 a mail with the subject "Reproducible Builds — proof of concept successful for 83% of all sources in main" was send to debian-devel-announce@lists.ddebian.org officially anouncing the project to the Debian developer community at large.

To be sorted out

- 2015-03-26: `binutils 2.25-6` is built with `--enable-deterministic-archives`
- testing `testing`* and `experimental`* now, pkg sets available too.
- 2015-05-27: `iceweasel 38.0.1-5` is reproducible.

Google Summer of Code 2015

During the summer of 2015 akira and Dhole will be working on moving forward reproducible builds as a Google Summer of Code project. Follow the links to check the accepted akira's application and Dhole's application. Dhole also made a blog post about how Dhole got into GSoC 2015.

CCCamp 2015

Short mention of Lunar's talk to be written here. Add links.

DebConf15

To be written: the first real life meeting of the Debian team. Talk given, roundstable discussion, hacking session. Mentioned in several talks, incl DPL key note. SOURCE_DATE_EPOCH was invented around this time too.

Continous tests for Coreboot, OpenWrt, NetBSD, FreeBSD, Arch Linux and Fedora

to be written: tests for these six projects have been added between June and December 2015…

Reproducible World Summit, December 1-3, 2015, Athens, Greece

to be written, maybe some photos to be shared, pointers to reports, new mailinglists, new irc channel, an even wider community has started to grow, website.

2016 and 2017

Are largely missing here, we should fix this, rather sooner than later.

In January 2017 we learned, that John Gilmore wrote an interesting mail about how Cygnus.com worked on reproducible builds in the early 1990s. It's eye opening to see how the dealt with basically the very same problems we're dealing with today, how they solved them and then to realize that most of this has been forgotten and bit-rotted in the last 20 years. How will we prevent history repeating itself here?

On August 21st 2017 reproducible-builds where first mentioned in Debian Policy, 4.1.0.

Archive wide rebuilds

- 2013-09-07 by David Suárez. 24% of 5240 source packages reproducible. Variations: time, build path.
- 2014-01-26 by David Suárez. 67% of 6887 source packages reproducible. Variations: time, build path.
- 2014-09-19 by Lunar, 30% of 172 source core packages reproducible. Variations: time, file order.
- Updated daily since 2014-09-28 by jenkins.debian.net. On 2014-11-11, 13213 (61.4%) out of 21448 packages are reproducible.

Publicity

please see the Publicity page in the Debian wiki which we still need to migrate to reproducible-builds.org.

Contributors

See this page for an incomplete list of contributors so far.

← Documentation index