There and Back Again, Reproducibly!

Vagrant Cascadian and Holger Levsen

linuxdev-br, 2019-08-04

Who are we

	Vagrant	Holger
debian user	2001	1995
debian developer	2010	2007
reproducible builds	2015	2014

When we say reproducible

https://reproducible-builds.org/docs/definition/

A build is reproducible if given the same source code, build environment and build instructions, any party can recreate bit-by-bit identical copies of all specified artifacts.

Reproducible Builds

Humble beginnings

Unexpected guests

First Breakfast! Second Breakfast? Elevensies?

Elevensies!

openSUSE

QUBES OS

BuildRoot
Making Embedded Linux Easy

openembedded

OpenWrt
Wireless Freedom

GuixSD

Tails

yocto
PROJECT

NetBSD

Symfony

ElectroBSD

MONERO

Webconverger

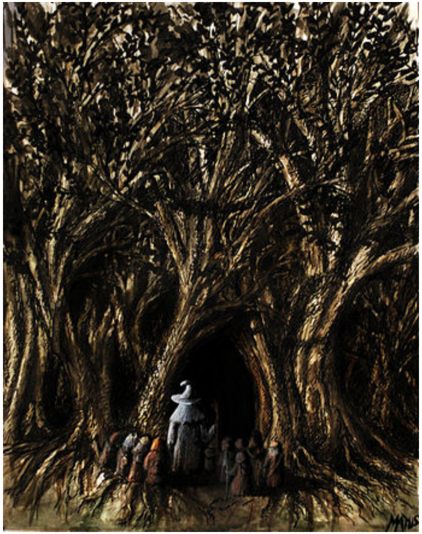
Dangerous Journeys

trolls

caves

Who does your hardware serve?

Dark and forboding places



spiders

Under the mountain

And back again

Who watches

The End . . . Or the Beginning?

Reproducible Builds

`https://reproducible-builds.org`

Once upon a time

- A list mail in 1997, very few more in 2001 and 2003.

Once upon a time

- A list mail in 1997, very few more in 2001 and 2003.
- Then, in 2011 and 2012, Bitcoin and Torbrowser were made reproducible.

Once upon a time

- A list mail in 1997, very few more in 2001 and 2003.
- Then, in 2011 and 2012, Bitcoin and Torbrowser were made reproducible.
- Wow.

Why unreproducibilities exist (historically)

- Historically software was reproducible! Every bit counted.

Why unreproducibilities exist (historically)

- Historically software was reproducible! Every bit counted.
- And every bit was known.

Why unreproducibilities exist (historically)

- Historically software was reproducible! Every bit counted.
- And every bit was known.
- Bit for bit reproducible GNU toolchain in the early 90s on 10(?) architectures.

Why unreproducibilities exist (historically)

- Historically software was reproducible! Every bit counted.
- And every bit was known.
- Bit for bit reproducible GNU toolchain in the early 90s on 10(?) architectures.
- And then we all forgot.

Motivation for reproducible builds

- Why do we care?

Motivation for reproducible builds

- Why do we care?
- reproducible builds:

Motivation for reproducible builds

- Why do we care?
- reproducible builds:
    - can detect backdoored build environments on developer systems or project build machines

Motivation for reproducible builds

- Why do we care?
- reproducible builds:
    - can detect backdoored build environments on developer systems or project build machines
    - cannot detect flaws in sources

Motivation for reproducible builds

- Why do we care?
- reproducible builds:
  - can detect backdoored build environments on developer systems or project build machines
  - cannot detect flaws in sources
- see our previous talks where we explain motivation in greater detail, eg. look for 'a tale of three developers'

Debian

- In 2013 some people in Debian began to investigate this.

Debian

- In 2013 some people in Debian began to investigate this.
- And kicked it off in 2014 by introducing systematic testing, classifications and weekly blogs.

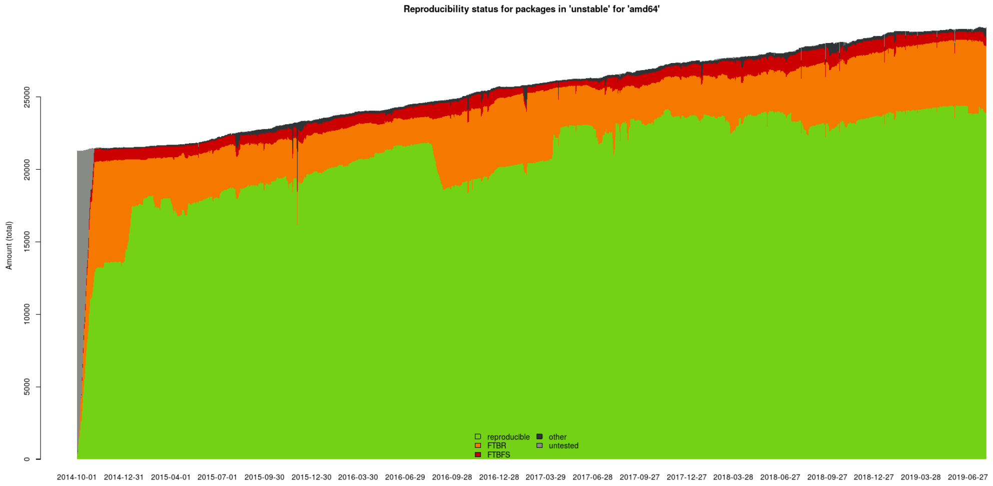
Debian

- In 2013 some people in Debian began to investigate this.
- And kicked it off in 2014 by introducing systematic testing, classifications and weekly blogs.
- Since 2017 in Debian Policy, as a "should" directive, not "must".

Debian

- In 2013 some people in Debian began to investigate this.
- And kicked it off in 2014 by introducing systematic testing, classifications and weekly blogs.
- Since 2017 in Debian Policy, as a "should" directive, not "must".
- 2023 with "must"?

Debian main unstable/amd64, since October 2014



Shared research and developments / WIP

- Test/research setup for many but not all projects.

Shared research and developments / WIP

- Test/research setup for many but not all projects.
- Since end of 2018 shared database for some of those.

Shared research and developments / WIP

- Test/research setup for many but not all projects.
- Since end of 2018 shared database for some of those.
- Sharing issues, patches and upstreaming them.

Shared research and developments / WIP

- Test/research setup for many but not all projects.
- Since end of 2018 shared database for some of those.
- Sharing issues, patches and upstreaming them.
- Shared public blog, now called monthly report.

Shared research and developments / WIP

- Test/research setup for many but not all projects.
- Since end of 2018 shared database for some of those.
- Sharing issues, patches and upstreaming them.
- Shared public blog, now called monthly report.
- More collaboration is possible!

What's causing unreproducibilities

- timestamps

What's causing unreproducibilities

- timestamps
- timestamps

What's causing unreproducibilities

- timestamps
- timestamps
- timestamps

What's causing unreproducibilities

- timestamps
- timestamps
- timestamps
- build paths

What's causing unreproducibilities

- timestamps
- timestamps
- timestamps
- build paths
- timezones, locales

What's causing unreproducibilities

- timestamps
- timestamps
- timestamps
- build paths
- timezones, locales
- hundreds of classes of causes !

What's causing unreproducibilities

- timestamps
- timestamps
- timestamps
- build paths
- timezones, locales
- hundreds of classes of causes !
- It's fun to discover these! Well, mostly.

A light at the end of the forest!

`https://diffoscope.org`

- Recursive and human-readable "diff"

A light at the end of the forest!

`https://diffoscope.org`

- Recursive and human-readable "diff"
  - locates and diagnoses reproducibility issues

A light at the end of the forest!

`https://diffoscope.org`

- Recursive and human-readable "diff"
  - locates and diagnoses reproducibility issues
  - not used for determining whether something is reproducible!

A light at the end of the forest!

`https://diffoscope.org`

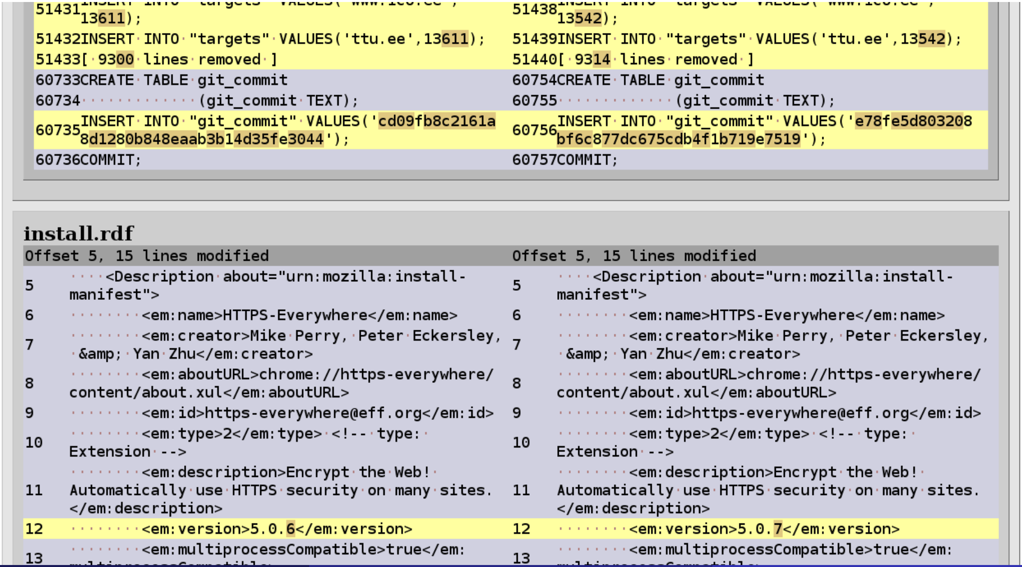
- Recursive and human-readable "diff"
  - locates and diagnoses reproducibility issues
  - not used for determining whether something is reproducible!
  - used for analysing why

A light at the end of the forest!

`https://diffoscope.org`

- Recursive and human-readable "diff"
    - locates and diagnoses reproducibility issues
    - not used for determining whether something is reproducible!
    - used for analysing why
- available for Debian, Fedora, OpenSUSE, Archlinux, GNU Guix, NixOS, FreeBSD, NetBSD, Homebrew, PypI, ...

diffoscope example



```
51431                                                        51438
     13611);                                                      13542);
51432INSERT·INTO·"targets"·VALUES('ttu.ee',13611);           51439INSERT·INTO·"targets"·VALUES('ttu.ee',13542);
51433[·9300·lines·removed·]                                  51440[·9314·lines·removed·]
60733CREATE·TABLE·git_commit                                 60754CREATE·TABLE·git_commit
60734·············(git_commit·TEXT);                         60755·············(git_commit·TEXT);
60735INSERT·INTO·"git_commit"·VALUES('cd09fb8c2161a          60756INSERT·INTO·"git_commit"·VALUES('e78fe5d803208
     8d1280b848eaab3b14d35fe3044');                               bf6c877dc675cdb4f1b719e7519');
60736COMMIT;                                                  60757COMMIT;
```

install.rdf

```
Offset 5, 15 lines modified                                  Offset 5, 15 lines modified
5    ····<Description·about="urn:mozilla:install-            5    ····<Description·about="urn:mozilla:install-
     manifest">                                                   manifest">
6    ········<em:name>HTTPS-Everywhere</em:name>             6    ········<em:name>HTTPS-Everywhere</em:name>
7    ········<em:creator>Mike·Perry,·Peter·Eckersley,        7    ········<em:creator>Mike·Perry,·Peter·Eckersley,
     ·&amp;·Yan·Zhu</em:creator>                                 ·&amp;·Yan·Zhu</em:creator>
8    ········<em:aboutURL>chrome://https-everywhere/         8    ········<em:aboutURL>chrome://https-everywhere/
     content/about.xul</em:aboutURL>                              content/about.xul</em:aboutURL>
9    ········<em:id>https-everywhere@eff.org</em:id>         9    ········<em:id>https-everywhere@eff.org</em:id>
10   ········<em:type>2</em:type>·<!--·type:·                10   ········<em:type>2</em:type>·<!--·type:·
     Extension·-->                                                Extension·-->
     ········<em:description>Encrypt·the·Web!·                    ········<em:description>Encrypt·the·Web!·
11   Automatically·use·HTTPS·security·on·many·sites.         11   Automatically·use·HTTPS·security·on·many·sites.
     </em:description>                                            </em:description>
12   ········<em:version>5.0.6</em:version>                  12   ········<em:version>5.0.7</em:version>
13   ········<em:multiprocessCompatible>true</em:            13   ········<em:multiprocessCompatible>true</em:
```

diffoscope, supported file types

Android APK files, Android boot images, Ar(1) archives, Berkeley DB database files, Bzip2 archives, Character/block devices, ColorSync colour profiles (.icc), Coreboot CBFS filesystem images, Cpio archives, Dalvik .dex files, Debian .buildinfo files, Debian .changes files, Debian source packages (.dsc), Device Tree Compiler blob files, Directories, ELF binaries, Ext2/ext3/ext4/btrfs filesystems, FreeDesktop Fontconfig cache files, FreePascal files (.ppu), Gettext message catalogues, GHC Haskell .hi files, GIF image files, Git repositories, GNU R database files (.rdb), GNU R Rscript files (.rds), Gnumeric spreadsheets, Gzipped files, ISO 9660 CD images, Java .class files, JavaScript files, JPEG images, JSON files, LLVM IR bitcode files, MacOS binaries, Microsoft Windows icon files, Microsoft Word .docx files, Mono 'Portable Executable' files, Ogg Vorbis audio files, OpenOffice .odt files, OpenSSH public keys, OpenWRT package archives (.ipk), PDF documents, PGP signed/encrypted messages, PNG images, PostScript documents, RPM archives, Rust object files (.deflate), SQLite databases, SquashFS filesystems, Statically-linked binaries, Symlinks, Tape archives (.tar), Tcpdump capture files (.pcap), Text files, TrueType font files, XML binary schemas (.xsb), XML files, XZ compressed files, etc.

Try diffoscope!

`https://try.diffoscope.org`

- diffoscope is useful beyond reproducible builds, eg.

Try diffoscope!

`https://try.diffoscope.org`

- diffoscope is useful beyond reproducible builds, eg.
  - for checking security updates only change what should be changed

Try diffoscope!

`https://try.diffoscope.org`

- diffoscope is useful beyond reproducible builds, eg.
  - for checking security updates only change what should be changed
  - for development too

A barrel in the river

reprotest: builds something twice with many variations

- `https://salsa.debian.org/reproducible/reprotest`

A barrel in the river

reprotest: builds something twice with many variations

- `https://salsa.debian.org/reproducible/reprotest`
- if unreproducible: reduce variations until (hopefully) the cause has been identified

A barrel in the river

reprotest: builds something twice with many variations

- `https://salsa.debian.org/reproducible/reprotest`
- if unreproducible: reduce variations until (hopefully) the cause has been identified
- Please help!

Theory vs Praxis

- 93% is a lie.

Theory vs Praxis

- 93% is a lie.
- Getting software reproducible in theory is 33% of the way.

Theory vs Praxis

- 93% is a lie.
- Getting software reproducible in theory is 33% of the way.
- The next 33% are about reproducible builds in practice, which means changing distro tools and workflows. Technically easy. . .

Theory vs Praxis

- 93% is a lie.
- Getting software reproducible in theory is 33% of the way.
- The next 33% are about reproducible builds in practice, which means changing distro tools and workflows. Technically easy. . .
- The last 33% are again different for each distro and divided into these questions:

Theory vs Praxis

- 93% is a lie.
- Getting software reproducible in theory is 33% of the way.
- The next 33% are about reproducible builds in practice, which means changing distro tools and workflows. Technically easy. . .
- The last 33% are again different for each distro and divided into these questions:
  - distributing trust

Theory vs Praxis

- 93% is a lie.
- Getting software reproducible in theory is 33% of the way.
- The next 33% are about reproducible builds in practice, which means changing distro tools and workflows. Technically easy. . .
- The last 33% are again different for each distro and divided into these questions:
  - distributing trust
  - how to "Enable everyone to independently. . . " in practice. (eg for Debian there are two designs with code, but. . .)

Four summits so far

- Athens 2015

Four summits so far

- Athens 2015
- Berlin 2016

Four summits so far

- Athens 2015
- Berlin 2016
- Berlin 2017

Four summits so far

- Athens 2015
- Berlin 2016
- Berlin 2017
- Paris 2018

Four summits so far

- Athens 2015
- Berlin 2016
- Berlin 2017
- Paris 2018
- Marrakesh 2019

Four summits so far

- Athens 2015
- Berlin 2016
- Berlin 2017
- Paris 2018
- Marrakesh 2019
- . . .

Collaboration is so great, again.

Collaboration, again.

- We stand on the shoulders of giants.

Collaboration, again.

- We stand on the shoulders of giants.
- And women, men and others,

Collaboration, again.

- We stand on the shoulders of giants.
- And women, men and others,
- And elves and dwarves,

Collaboration, again.

- We stand on the shoulders of giants.
- And women, men and others,
- And elves and dwarves,
- And wizards and hobbits,

Collaboration, again.

- We stand on the shoulders of giants.
- And women, men and others,
- And elves and dwarves,
- And wizards and hobbits,
- And beings beyond our current imagination,

Collaboration, again.

- We stand on the shoulders of giants.
- And women, men and others,
- And elves and dwarves,
- And wizards and hobbits,
- And beings beyond our current imagination,
- And we welcome you.

Collaboration, again.

- We stand on the shoulders of giants.
- And women, men and others,
- And elves and dwarves,
- And wizards and hobbits,
- And beings beyond our current imagination,
- And we welcome you.
- And we welcome Free Software.

The end / summary

- We made 93% of the first 33%.

The end / summary

- We made 93% of the first 33%.
- Sounds good, but 7% of 30000 source packages means 2100 unreproducible source packages.

The end / summary

- We made 93% of the first 33%.
- Sounds good, but 7% of 30000 source packages means 2100 unreproducible source packages.
- Currently. There's new software every hour.

The end / summary

- We made 93% of the first 33%.
- Sounds good, but 7% of 30000 source packages means 2100 unreproducible source packages.
- Currently. There's new software every hour.
- The 2nd 33% are more blurry, some small projects made it, no big one yet.

The end / summary

- We made 93% of the first 33%.
- Sounds good, but 7% of 30000 source packages means 2100 unreproducible source packages.
- Currently. There's new software every hour.
- The 2nd 33% are more blurry, some small projects made it, no big one yet.
- There are ideas and even code for the last 33%, but we can't go on that path without the first 66%...

The end / summary

- We made 93% of the first 33%.
- Sounds good, but 7% of 30000 source packages means 2100 unreproducible source packages.
- Currently. There's new software every hour.
- The 2nd 33% are more blurry, some small projects made it, no big one yet.
- There are ideas and even code for the last 33%, but we can't go on that path without the first 66%...
- There is a lot to do. Please. Help.

Questions?

Thank you for your time and contributions.

It's been a long journey but we will get there. And back again, on to new journeys!

https://reproducible-builds.org
https://try.diffoscope.org

Copyright

Still hungry?

We could offer pizza. . .