Reproducible Toolchains For The Win!

Vagrant Cascadian

Cauldron 2019, 2019-09-13

Who

	Vagrant
debian user	2001
debian developer	2010
reproducible builds	2015

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.

Why unreproducibilities exist (prehistorically)

- Historically software was reproducible! Every bit counted.

Why unreproducibilities exist (prehistorically)

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

Why unreproducibilities exist (prehistorically)

- 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 (prehistorically)

- 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 about reproducible builds?

Motivation for reproducible builds

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

Motivation for reproducible builds

- Why do we care about reproducible builds?
  - Can detect backdoored build environments on developer systems or project build machines
  - Optimize build cache, limiting rebuilds

Motivation for reproducible builds

- Why do we care about reproducible builds?
  - Can detect backdoored build environments on developer systems or project build machines
  - Optimize build cache, limiting rebuilds
  - License compliance verification

Motivation for reproducible builds

- Why do we care about reproducible builds?
  - Can detect backdoored build environments on developer systems or project build machines
  - Optimize build cache, limiting rebuilds
  - License compliance verification
  - It just feels right

Motivation for reproducible builds

- Why do we care about reproducible builds?
  - Can detect backdoored build environments on developer systems or project build machines
  - Optimize build cache, limiting rebuilds
  - License compliance verification
  - It just feels right
- What will reproducible builds not do for you?

Motivation for reproducible builds

- Why do we care about reproducible builds?
  - Can detect backdoored build environments on developer systems or project build machines
  - Optimize build cache, limiting rebuilds
  - License compliance verification
  - It just feels right
- What will reproducible builds not do for you?
  - Cannot detect flaws in sources

License compliance: GPL

GPL compliance

- source code is what it used to write free software

License compliance: GPL

GPL compliance

- source code is what it used to write free software
- binary code is what is actually used

License compliance: GPL

GPL compliance

- source code is what it used to write free software
- binary code is what is actually used
- How can you prove that the binaries used are the result of the source code?

License compliance: hardware

`https://www.fsf.org/resources/hw/endorsement/respects-your-freedom`

- Firmware

License compliance: hardware

https://www.fsf.org/resources/hw/endorsement/respects-your-freedom

- Firmware
- Operating system

License compliance: hardware

`https://www.fsf.org/resources/hw/endorsement/respects-your-freedom`

- Firmware
- Operating system
- Other software

The problem is one of Time

https://reproducible-builds.org/docs/source-date-epoch/

gcc: time

Support for SOURCE_DATE_EPOCH added to gcc 2019-04:
`https://gcc.gnu.org/git/?p=gcc.git;a=commitdiff;h=e3e8c48c4a494d9da741c1c8ea6c4c0b7c4ff934`

gzip: time

gzip

- –no-name (a.k.a. -n)
- [PATCH] Do not store mtime when compressing stdin
  `https://debbugs.gnu.org/cgi/bugreport.cgi?bug=32342`

Build Paths

https://reproducible-builds.org/specs/build-path-prefix-map/

gcc: build paths: debug symbols

debug symbols

- -fdebug-prefix-map

gcc: build paths: debug symbols

debug symbols

- -fdebug-prefix-map
- Debian: dpkg support 2016-05, 1.18.5

gcc: moar build paths

-fmacro-prefix-map -ffile-prefix-map
`https://gcc.gnu.org/bugzilla/show_bug.cgi?id=70268`

gcc: build paths: results

https://tests.reproducible-builds.org/debian/issues/unstable/gcc_captures_build_path_issue.html
618: reproducible (possibly due to -ffile-prefix-map)
1015: still unreproducible

gcc: build paths: unresolved

`https://gcc.gnu.org/ml/gcc-patches/2017-07/msg01315.html`

- Use BUILD_PATH_PREFIX_MAP environment variable instead of -ffile-prefix-map

gcc: build paths: unresolved

https://gcc.gnu.org/ml/gcc-patches/2017-07/msg01315.html

- Use BUILD_PATH_PREFIX_MAP environment variable instead of -ffile-prefix-map
- commandline arguments sometimes get embedded in build results

gcc: build paths: unresolved

`https://gcc.gnu.org/ml/gcc-patches/2017-07/msg01315.html`

- Use BUILD_PATH_PREFIX_MAP environment variable instead of -ffile-prefix-map
- commandline arguments sometimes get embedded in build results
- REJECTED on premise of taking data from environment variable

gcc: build paths: way forward

gcc build path proposals?

- -ffile-prefix-map-from-env BUILD_PATH_PREFIX_MAP

gcc: build paths: way forward

gcc build path proposals?

- -ffile-prefix-map-from-env BUILD_PATH_PREFIX_MAP
- Workaround build in same directory

gcc: LTO

Report LTO-induced indeterminism from global constructors
`https://gcc.gnu.org/bugzilla/show_bug.cgi?id=91307`

Sort your wildcards

GNU make

- wildcard/glob should be sorted
  `https://savannah.gnu.org/bugs/index.php?52076`

Sort your wildcards

GNU make

- wildcard/glob should be sorted
  `https://savannah.gnu.org/bugs/index.php?52076`
- src/read.c (parse_file_seq): [SV 52076] Sort wildcard results.
  `https://git.savannah.gnu.org/cgit/make.git/commit/?id=eedea52afb2069e54188508cd87cb7724b30dd6a`

There is so much more to think about

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

- Volatile inputs can disappear

There is so much more to think about

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

- Volatile inputs can disappear
- Value initialization

There is so much more to think about

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

- Volatile inputs can disappear
- Value initialization
- Locales

There is so much more to think about

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

- Volatile inputs can disappear
- Value initialization
- Locales
- Archive metadata

There is so much more to think about

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

- Volatile inputs can disappear
- Value initialization
- Locales
- Archive metadata
- Stable order for outputs

There is so much more to think about

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

- Volatile inputs can disappear
- Value initialization
- Locales
- Archive metadata
- Stable order for outputs
- Randomness

There is so much more to think about

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

- Volatile inputs can disappear
- Value initialization
- Locales
- Archive metadata
- Stable order for outputs
- Randomness
- Build path

There is so much more to think about

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

- Volatile inputs can disappear
- Value initialization
- Locales
- Archive metadata
- Stable order for outputs
- Randomness
- Build path
- System images

There is so much more to think about

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

- Volatile inputs can disappear
- Value initialization
- Locales
- Archive metadata
- Stable order for outputs
- Randomness
- Build path
- System images
- Time

There is so much more to think about

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

- Volatile inputs can disappear
- Value initialization
- Locales
- Archive metadata
- Stable order for outputs
- Randomness
- Build path
- System images
- Time
- Timezones

There is so much more to think about

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

- Volatile inputs can disappear
- Value initialization
- Locales
- Archive metadata
- Stable order for outputs
- Randomness
- Build path
- System images
- Time
- Timezones
- Time

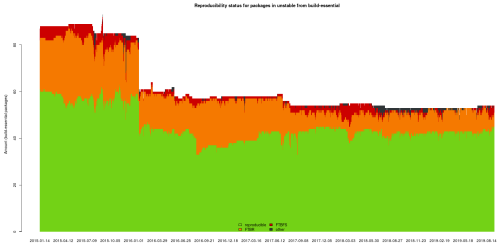
There is so much more to think about

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

- Volatile inputs can disappear
- Value initialization
- Locales
- Archive metadata
- Stable order for outputs
- Randomness
- Build path
- System images
- Time
- Timezones
- Time
- Time again

build essential: debian unstable

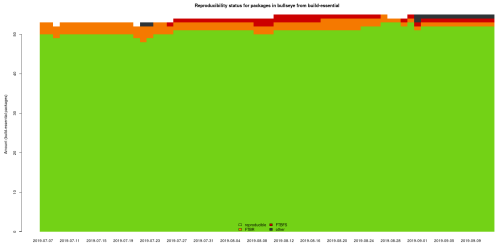
https://tests.reproducible-builds.org/debian/unstable/amd64/pkg_set_build-essential.html



of 54 packages:
6 (11.1%) unreproducible: bash+ linux perl# gmp gcc-9 binutils
3 (5.6%) failed to build: pcre2 glibc xz-utils
45 (83.3%) reproducible: ...

build essential: debian bullseye

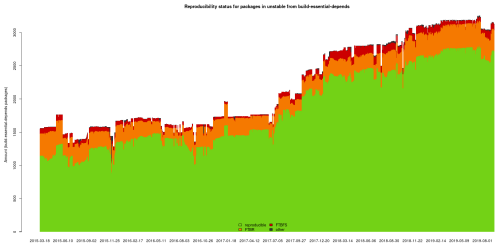
https://tests.reproducible-builds.org/debian/bullseye/amd64/pkg_set_build-essential.html



53 packages:
1 (1.9%) unreproducible: gcc-9
1 (1.9%) failed to build: xz-utils
1 (1.9%) other problems: libgcrypt20
50 (94.3%) reproducible:: ...

build essential build depends: debian unstable

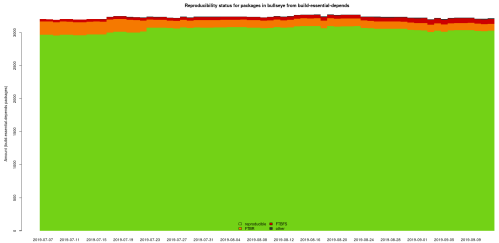
https://tests.reproducible-builds.org/debian/unstable/amd64/pkg_set_build-essential-depends.html



of 3061 packages:
312 (10.2%) unreproducible
83 (2.7%) failed to build
4 (0.1%) misc issues
2662 (87.0%) reproducible

build essential build depends: debian bullseye

https://tests.reproducible-builds.org/debian/bullseye/amd64/pkg_set_build-essential-depends.html



of 3144 packages:
100 (3.2%) unreproducible
69 (2.2%) failed to build
13 (0.4%) misc issues
2962 (94.2%) reproducible

bootstrapping

`https://bootstrappable.org/`
What compiler do you use to compile your compiler?

Trusting Trust

Reflections on Trusting Trust by Ken Thompson 1984

- https://www.ece.cmu.edu/~ganger/712.fall02/papers/p761-thompson.pdf

Diverse Double-Compilation

Diverse Double-Compilation by David A. Wheeler 2005/2009

- https://www.dwheeler.com/trusting-trust/

Untangling the bootstraping Mes

`https://savannah.gnu.org/projects/mes`

GNU Mes

Mutual self-hosting Scheme interpreter written in ~5,000 LOC of simple C and a Nyacc-based C compiler written in Scheme.

diffoscope

`https://diffoscope.org`

- Recursive and human-readable "diff"

diffoscope

`https://diffoscope.org`

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

diffoscope

`https://diffoscope.org`

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

diffoscope

`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

diffoscope

`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, Pypl, ...

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`

Reprotest

reprotest: builds something twice with many variations

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

Reprotest

reprotest: builds something twice with many variations

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

Reprotest

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!

How to help

`https://reproducible-builds.org/contribute/`

- rb-general@lists.reproducible-builds.org

How to help

`https://reproducible-builds.org/contribute/`

- rb-general@lists.reproducible-builds.org
- irc.oftc.net #reproducible-builds

How to help

`https://reproducible-builds.org/contribute/`

- rb-general@lists.reproducible-builds.org
- irc.oftc.net #reproducible-builds
- code hosting: `https://salsa.debian.org/reproducible-builds`

How to help

`https://reproducible-builds.org/contribute/`

- rb-general@lists.reproducible-builds.org
- irc.oftc.net #reproducible-builds
- code hosting: `https://salsa.debian.org/reproducible-builds`
- test infrastructure: `https://tests.reproducible-builds.org`

Copyright