Beyond Trusting Open Source Software

Vagrant Cascadian <vagrant@reproducible-builds.org>

LinuxFest Northwest 2026-04-26

Who am I

	Vagrant
debian user	2001
debian developer	2010
reproducible builds	2015

Free and Open Source Software

Free and Open Source Software

- Use

Free and Open Source Software

Free and Open Source Software

- Use
- Study

Free and Open Source Software

Free and Open Source Software

- Use
- Study
- Change

Free and Open Source Software

Free and Open Source Software

- Use
- Study
- Change
- Share

Free and Open Source Software

Free and Open Source Software

- Use
- Study
- Change
- Share
- Community

A taste of source

from bash 5.0 assoc.c:

```
assoc_insert (hash, key, value)
     HASH_TABLE *hash;
     char *key;
     char *value;
{
  BUCKET_CONTENTS *b;

  b = hash_search (key, hash, HASH_CREATE);
  if (b == 0)
    return -1;
  /* If we are overwriting an existing element's value, we're not going to
     use the key.  Nothing in the array assignment code path frees the key
     string, so we can free it here to avoid a memory leak. */
  if (b->key != key)
```

Building the software

```
./configure
make
make install
```

A resulting binary might look like

```
$ head /bin/bash
ELF&@@8 @@88TTTDDPtdDDQtdRtd0<0</lib/ld-linux-aarch64.so.1GNUy;OgUQGNU 04
                                                                    #!Jzd
NL@@@AB
OIq(h   @(
          H &RD!D
                    $DP‘
 @A4@ABf L0 dPCDDBE % 32BX@TD$
 @A%

!O‘O@@bBh
         HBH
Xq@ Y       ‘1B
BdH(O"BB1@
```

Reproducible Builds

`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.

Debian

- The Universal Operating System

Reproducible Builds At Scale

Debian

- The Universal Operating System
- ~37000 source packages . . . and counting

Reproducible Builds At Scale

Debian

- The Universal Operating System
- ~37000 source packages . . . and counting
- 380 million lines of code . . . and counting!

Reproducible Builds At Scale

Debian

- The Universal Operating System
- ~37000 source packages . . . and counting
- 380 million lines of code . . . and counting!
- ~96% reproducible

Chaos and Freinds

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

- Timestamps

Chaos and Freinds

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

- Timestamps
- User Information

Chaos and Freinds

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

- Timestamps
- User Information
- Host system information

Chaos and Freinds

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

- Timestamps
- User Information
- Host system information
- Randomness

Chaos and Freinds

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

- Timestamps
- User Information
- Host system information
- Randomness
- So many more!

Chaos and Freinds

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

- Timestamps
- User Information
- Host system information
- Randomness
- So many more!
- Especially Timestamps!

Deterministic time?

SOURCE_DATE_EPOCH (seconds since 1970-01-01)
`https://reproducible-builds.org/docs/source-date-epoch/`
Supported in GCC, Clang, and more!

So you want Reproducible builds

`https://reproducible-builds.org/docs/recording/`
Providing sufficient information for independent verification:

- ...

So you want Reproducible builds

`https://reproducible-builds.org/docs/recording/`
Providing sufficient information for independent verification:

- ...
- "toolchain" packages at specific versions

So you want Reproducible builds

`https://reproducible-builds.org/docs/recording/`
Providing sufficient information for independent verification:

- ...
- "toolchain" packages at specific versions
- SOURCE_DATE_EPOCH

So you want Reproducible builds

`https://reproducible-builds.org/docs/recording/`
Providing sufficient information for independent verification:

- ...
- "toolchain" packages at specific versions
- SOURCE_DATE_EPOCH
- Works best with Free and Open Source Software!

So you want Reproducible builds

`https://reproducible-builds.org/docs/recording/`
Providing sufficient information for independent verification:

- ...
- "toolchain" packages at specific versions
- SOURCE_DATE_EPOCH
- Works best with Free and Open Source Software!
- Automated testing (QA, CI, etc.)

Reprotest

reprotest

- builds something twice with many variations

Reprotest

reprotest

- builds something twice with many variations
- displays the differences between results

Reprotest

reprotest

- builds something twice with many variations
- displays the differences between results
- `https://salsa.debian.org/reproducible/reprotest`

Reprotest

reprotest

- builds something twice with many variations
- displays the differences between results
- `https://salsa.debian.org/reproducible/reprotest`
- if unreproducible: "bisect" the variations

diffocope

`https://diffoscope.org`

- Recursive and human-readable "diff"

diffocope

`https://diffoscope.org`

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

diffocope

`https://diffoscope.org`

- Recursive and human-readable "diff"
- locates and highlights reproducibility issues
- Supported on many distributions

diffoscope example

51431	13611);	51438	13542);
51432	INSERT INTO "targets" VALUES('ttu.ee',13611);	51439	INSERT INTO "targets" VALUES('ttu.ee',13542);
51433	[9300 lines removed]	51440	[9314 lines removed]
60733	CREATE TABLE git_commit	60754	CREATE TABLE git_commit
60734	(git_commit TEXT);	60755	(git_commit TEXT);
60735	INSERT INTO "git_commit" VALUES('cd09fb8c2161a8d1280b848eaab3b14d35fe3044');	60756	INSERT INTO "git_commit" VALUES('e78fe5d803208bf6c877dc675cdb4f1b719e7519');
60736	COMMIT;	60757	COMMIT;

install.rdf

Offset 5, 15 lines modified		Offset 5, 15 lines modified	
5	<Description about="urn:mozilla:install-manifest">	5	<Description about="urn:mozilla:install-manifest">
6	<em:name>HTTPS-Everywhere</em:name>	6	<em:name>HTTPS-Everywhere</em:name>
7	<em:creator>Mike Perry, Peter Eckersley, & Yan Zhu</em:creator>	7	<em:creator>Mike Perry, Peter Eckersley, & Yan Zhu</em:creator>
8	<em:aboutURL>chrome://https-everywhere/content/about.xul</em:aboutURL>	8	<em:aboutURL>chrome://https-everywhere/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: Extension -->	10	<em:type>2</em:type> <!-- type: Extension -->
11	<em:description>Encrypt the Web! Automatically use HTTPS security on many sites.</em:description>	11	<em:description>Encrypt the Web! Automatically use HTTPS security on many sites.</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:

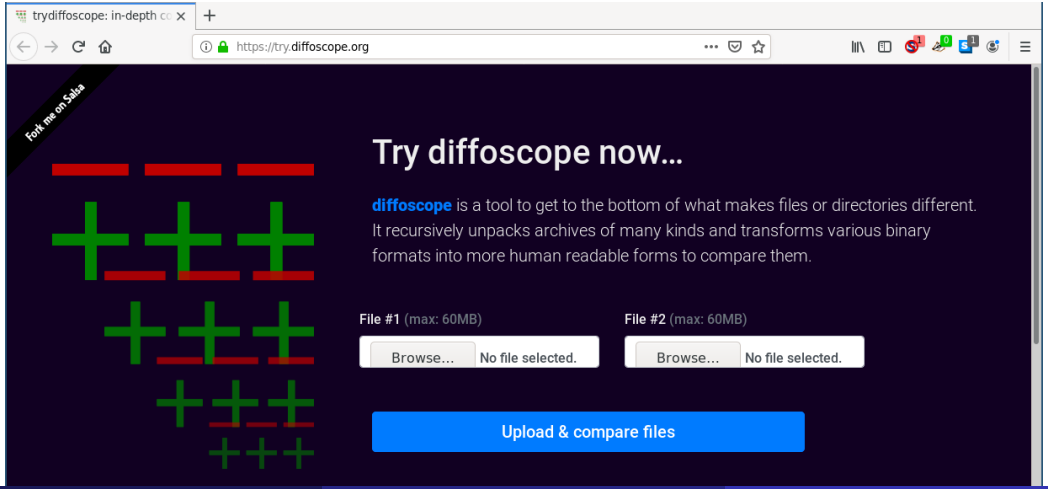
diffoscope, supported file types

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 online

And on the World Wide Web!

`https://try.diffoscope.org`



What you get with Reproducible Builds

Reproducible Builds provides...

- strong confidence...

What you get with Reproducible Builds

Reproducible Builds provides...

- strong confidence...
- that a binary was produced from a given source...

What you get with Reproducible Builds

Reproducible Builds provides...

- strong confidence...
- that a binary was produced from a given source...
- ...probably!

Trust

Different levels of trust:

- curl `http://example.net/hackme` | sudo sh

Trust

Different levels of trust:

- curl `http://example.net/hackme` | sudo sh
- curl –proto '=https' –tlsv1.2 -sSf `https://sh.rustup.rs` | sh

Trust

Different levels of trust:

- curl `http://example.net/hackme` | sudo sh
- curl –proto '=https' –tlsv1.2 -sSf `https://sh.rustup.rs` | sh
- download files, verify signatures . . . run code

Trust

Different levels of trust:

- curl `http://example.net/hackme` | sudo sh
- curl –proto '=https' –tlsv1.2 -sSf `https://sh.rustup.rs` | sh
- download files, verify signatures . . . run code
- download source, verify signature, compile from source

Trust

Different levels of trust:

- curl `http://example.net/hackme` | sudo sh
- curl –proto '=https' –tlsv1.2 -sSf `https://sh.rustup.rs` | sh
- download files, verify signatures . . . run code
- download source, verify signature, compile from source
- emerge –emptytree @world

Trust

Different levels of trust:

- curl `http://example.net/hackme` | sudo sh
- curl –proto '=https' –tlsv1.2 -sSf `https://sh.rustup.rs` | sh
- download files, verify signatures . . . run code
- download source, verify signature, compile from source
- emerge –emptytree @world
- rewrite everything in assembly

Trust

Different levels of trust:

- curl `http://example.net/hackme` | sudo sh
- curl –proto '=https' –tlsv1.2 -sSf `https://sh.rustup.rs` | sh
- download files, verify signatures . . . run code
- download source, verify signature, compile from source
- emerge –emptytree @world
- rewrite everything in assembly
- build it up from transitors

Trust

Different levels of trust:

- curl `http://example.net/hackme` | sudo sh
- curl –proto '=https' –tlsv1.2 -sSf `https://sh.rustup.rs` | sh
- download files, verify signatures . . . run code
- download source, verify signature, compile from source
- emerge –emptytree @world
- rewrite everything in assembly
- build it up from transitors
- I have a beach, some wood, abundant sunshine, and a lot of time

Trusting Trust

Ken Thompson
Reflections on Trusting Trust, 1984
https://archive.org/details/reflections-on-trusting-trust
https://research.swtch.com/nih

Building on a solid foundation of turtles

`https://bootstrappable.org`
Compiling your C compiler with a C compiler
And a C compiler to compile the other C compiler
...Ad infinitum

Rust bootstrap

Rust bootstrapping

- rust 1.95 needs...

Rust bootstrap

Rust bootstrapping

- rust 1.95 needs...
- rust 1.94 which needs...

Rust bootstrap

Rust bootstrapping

- rust 1.95 needs...
- rust 1.94 which needs...
- rust 1.93 which needs...

Rust bootstrap

Rust bootstrapping

- rust 1.95 needs...
- rust 1.94 which needs...
- rust 1.93 which needs...
- ...

Rust bootstrap

Rust bootstrapping

- rust 1.95 needs...
- rust 1.94 which needs...
- rust 1.93 which needs...
- ...
- rust 1.54 can be built with mrustc

Rust bootstrap

Rust bootstrapping

- rust 1.95 needs...
- rust 1.94 which needs...
- rust 1.93 which needs...
- ...
- rust 1.54 can be built with mrustc
- mrustc is written in C++

Rust bootstrap

Rust bootstrapping

- rust 1.95 needs...
- rust 1.94 which needs...
- rust 1.93 which needs...
- ...
- rust 1.54 can be built with mrustc
- mrustc is written in C++
- breaking news, newer mrustc can bootstrap 1.90!

Diverse Double Compiling

David A. Wheeler
Fully Countering Trusting Trust through Diverse Double-Compiling, 2009
`https://dwheeler.com/trusting-trust/dissertation/html/wheeler-trusting-trust-ddc.html`

A beautiful Mes

GNU Mes is a Scheme interpreter and C compiler for bootstrapping the GNU System.
`https://www.gnu.org/software/mes/`

We made the same Mes

Bit-for-bit identical Mes built on three different distributions
https://reproducible-builds.org/news/2019/12/21/reproducible-bootstrap-of-mes-c-compiler/

Beginning with a Mes

GNU Guix: The Reduced Binary Seed Bootstrap
`https://guix.gnu.org/en/manual/devel/en/guix.html#Reduced-Binary-Seed-Bootstrap`

- ...

Beginning with a Mes

GNU Guix: The Reduced Binary Seed Bootstrap
`https://guix.gnu.org/en/manual/devel/en/guix.html#Reduced-Binary-Seed-Bootstrap`

- ...
- Reduced to 145MB of bootstrap binaries (from 250MB)

Beginning with a Mes

GNU Guix: The Reduced Binary Seed Bootstrap
`https://guix.gnu.org/en/manual/devel/en/guix.html#Reduced-Binary-Seed-Bootstrap`

- ...
- Reduced to 145MB of bootstrap binaries (from 250MB)
- Using Mes and guile...

Beginning with a Mes

GNU Guix: The Reduced Binary Seed Bootstrap
`https://guix.gnu.org/en/manual/devel/en/guix.html#Reduced-Binary-Seed-Bootstrap`

- ...
- Reduced to 145MB of bootstrap binaries (from 250MB)
- Using Mes and guile...
- Builds from source GCC, binutils, glibc, etc.

Beginning with a Mes

GNU Guix: The Reduced Binary Seed Bootstrap
`https://guix.gnu.org/en/manual/devel/en/guix.html#Reduced-Binary-Seed-Bootstrap`

- ...
- Reduced to 145MB of bootstrap binaries (from 250MB)
- Using Mes and guile...
- Builds from source GCC, binutils, glibc, etc.
- 145MB of binaries is still not really auditable...

Before The Mes and Beyond

GNU Guix: The Full-Source Bootstrap
`https://guix.gnu.org/en/blog/2023/the-full-source-bootstrap-building-from-source-all-the-way-down/`
Now available via guix pull!

- hex0 (357-byte binary)

Before The Mes and Beyond

GNU Guix: The Full-Source Bootstrap
`https://guix.gnu.org/en/blog/2023/the-full-source-bootstrap-building-from-source-all-the-way-down/`
Now available via guix pull!

- hex0 (357-byte binary)
- hex1

Before The Mes and Beyond

GNU Guix: The Full-Source Bootstrap
`https://guix.gnu.org/en/blog/2023/the-full-source-bootstrap-building-from-source-all-the-way-down/`
Now available via guix pull!

- hex0 (357-byte binary)
- hex1
- hex2

Before The Mes and Beyond

GNU Guix: The Full-Source Bootstrap
`https://guix.gnu.org/en/blog/2023/the-full-source-bootstrap-building-from-source-all-the-way-down/`
Now available via guix pull!

- hex0 (357-byte binary)
- hex1
- hex2
- M0

Before The Mes and Beyond

GNU Guix: The Full-Source Bootstrap
`https://guix.gnu.org/en/blog/2023/the-full-source-bootstrap-building-from-source-all-the-way-down/`
Now available via guix pull!

- hex0 (357-byte binary)
- hex1
- hex2
- M0
- cc_x86

Before The Mes and Beyond

GNU Guix: The Full-Source Bootstrap
`https://guix.gnu.org/en/blog/2023/the-full-source-bootstrap-building-from-source-all-the-way-down/`
Now available via guix pull!

- hex0 (357-byte binary)
- hex1
- hex2
- M0
- cc_x86
- M2-Planet

Before The Mes and Beyond

GNU Guix: The Full-Source Bootstrap
`https://guix.gnu.org/en/blog/2023/the-full-source-bootstrap-building-from-source-all-the-way-down/`
Now available via guix pull!

- hex0 (357-byte binary)
- hex1
- hex2
- M0
- cc_x86
- M2-Planet
- mescc-tools

Before The Mes and Beyond

GNU Guix: The Full-Source Bootstrap
`https://guix.gnu.org/en/blog/2023/the-full-source-bootstrap-building-from-source-all-the-way-down/`
Now available via guix pull!

- hex0 (357-byte binary)
- hex1
- hex2
- M0
- cc_x86
- M2-Planet
- mescc-tools
- Mes

Before The Mes and Beyond

GNU Guix: The Full-Source Bootstrap
`https://guix.gnu.org/en/blog/2023/the-full-source-bootstrap-building-from-source-all-the-way-down/`
Now available via guix pull!

- hex0 (357-byte binary)
- hex1
- hex2
- M0
- cc_x86
- M2-Planet
- mescc-tools
- Mes
- TinyCC (patched)

Before The Mes and Beyond

GNU Guix: The Full-Source Bootstrap
`https://guix.gnu.org/en/blog/2023/the-full-source-bootstrap-building-from-source-all-the-way-down/`
Now available via guix pull!

- hex0 (357-byte binary)
- hex1
- hex2
- M0
- cc_x86
- M2-Planet
- mescc-tools
- Mes
- TinyCC (patched)
- old versions of GCC, binutils, glibc, gzip, tar ...

Before The Mes and Beyond

GNU Guix: The Full-Source Bootstrap
`https://guix.gnu.org/en/blog/2023/the-full-source-bootstrap-building-from-source-all-the-way-down/`
Now available via guix pull!

- hex0 (357-byte binary)
- hex1
- hex2
- M0
- cc_x86
- M2-Planet
- mescc-tools
- Mes
- TinyCC (patched)
- old versions of GCC, binutils, glibc, gzip, tar ...
- modern GCC and almost everything

Make it live

`https://github.com/fosslinux/live-bootstrap`

- A live environment

Make it live

`https://github.com/fosslinux/live-bootstrap`

- A live environment
- From kernel and a bit of source code

Make it live

`https://github.com/fosslinux/live-bootstrap`

- A live environment
- From kernel and a bit of source code
- To a reproducibly bootstrapped toolchain

Make it live

`https://github.com/fosslinux/live-bootstrap`

- A live environment
- From kernel and a bit of source code
- To a reproducibly bootstrapped toolchain
- no pregenerated "source" code shortcuts

Under that Turtle

How about...

...Without an operating system?

- ...

Under that Turtle

How about...

...Without an operating system?

- ...
- UEFI `https://git.stikonas.eu/andrius/stage0-uefi`

Under that Turtle

How about...

...Without an operating system?

- ...
- UEFI `https://git.stikonas.eu/andrius/stage0-uefi`
- Bare Metal `https://git.savannah.nongnu.org/cgit/stage0.git/tree/`

Forget Trust, Verify

No need to Trust, all we need is:

- Free/Libre and Open Source Software

Forget Trust, Verify

No need to Trust, all we need is:

- Free/Libre and Open Source Software
- Reproducible Builds

Forget Trust, Verify

No need to Trust, all we need is:

- Free/Libre and Open Source Software
- Reproducible Builds
- Bootstrapping

Forget Trust, Verify

No need to Trust, all we need is:

- Free/Libre and Open Source Software
- Reproducible Builds
- Bootstrapping
- Diverse compilation

Forget Trust, Verify

No need to Trust, all we need is:

- Free/Libre and Open Source Software
- Reproducible Builds
- Bootstrapping
- Diverse compilation
- ... and lots of compile cycles

Thanks

Help make it happen!
https://reproducible-builds.org/contribute/
https://reproducible-builds.org/donate/
https://reproducible-builds.org/who/sponsors/

Copyright and attributions