

Bit by bit identically reproducible builds for the RPM world

Lightning talk
as if it were possible
to describe this properly in 10 minutes...

Holger 'h01ger' Levsen
holger@debian.org

Devconf.cz (Brno, Czech Republic)
2016-02-06

about me

- B8BF 5413 7B09 D35C F026 FE9D 091A B856 069A AA1C
- Debian user since 1995
- Debian contributor since 2001
- Debian developer since 2007
- DebConf organizer, founded the DebConf video team
 - ▶ <http://video.debian.net>
- Debian-Edu (Debian for education)
- Debian QA (quality assurance)
 - ▶ <https://piuparts.debian.org>
 - ▶ <https://jenkins.debian.net> (1100 jobs continuously testing Debian)
- Debian LTS (Long Term Support)
- Debian Reproducible builds team member
 - ▶ until April 2016 together with Lunar funded by the Linux Foundation

Debian reproducible builds team

akira

Andrew Ayer

Asheesh Laroia

Chris Lamb

Chris West

Christoph Berg

Daniel Kahn Gillmor

David Suarez

Dhole

Drew Fisher

Esa Peuha

Guillem Jover

Hans-Christoph

Steiner

Helmut Grohne

Holger Levsen

Jelmer Vernooij

josch

Juan Picca

Lunar

Mathieu Bridon

Mattia Rizzolo

Nicolas Boulenguez

Niels Thykier

Niko Tyni

Paul Wise

Peter De Wachter

Philip Rinn

Reiner Herrmann

Stefano Rivera

Stéphane Glondou

Steven Chamberlain

Tom Fitzhenry

Valentin Lorentz

Wookey

Ximin Luo



Debian reproducible builds team

akira

Andrew Ayer

Asheesh Laroia

Chris Lamb

Chris West

Christoph Berg

Daniel Kahn Gillmor

David Suarez

Dhole

Drew Fisher

Esa Peuha

Guillem Jover

Hans-Christoph

Steiner

Helmut Grohne

Holger Levsen

Jelmer Vernooij

josch

Juan Picca

Lunar

Mathieu Bridon

Mattia Rizzolo

Nicolas Boulenguez

Niels Thykier

Niko Tyni

Paul Wise

Peter De Wachter

Philip Rinn

Reiner Herrmann

Stefano Rivera

Stéphane Glondou

Steven Chamberlain

Tom Fitzhenry

Valentin Lorentz

Wookey

Ximin Luo



Who are you?



Who are you?

- Seen a talk about reproducible builds?





1

Motivation

2

Common ressources

3

Status Debian

4

Status Non-Debian World

5

Future work

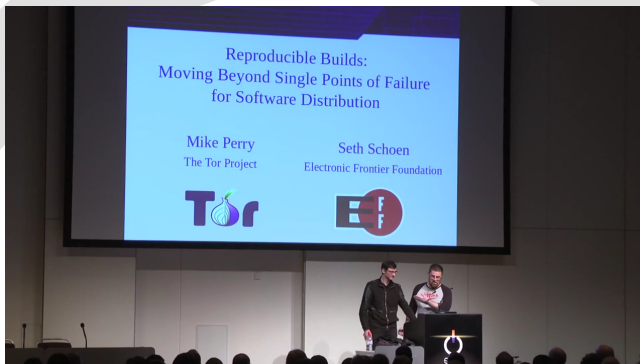
6

Getting involved

7

Questions, comments, ideas?

The problem



Available on media.ccc.de, 31c3



A few examples from that 31c3 talk

- CVE-2002-0083: remote root exploit in sshd, a single bit difference in the binary



A few examples from that 31c3 talk

- CVE-2002-0083: remote root exploit in sshd, a single bit difference in the binary
- 31c3 talk had a live demo with a kernel module modifying source code in memory only



A few examples from that 31c3 talk

- CVE-2002-0083: remote root exploit in sshd, a single bit difference in the binary
- 31c3 talk had a live demo with a kernel module modifying source code in memory only
- financial incentives to crack developer machines or a projects build infrastructure...



A few examples from that 31c3 talk

- CVE-2002-0083: remote root exploit in sshd, a single bit difference in the binary
- 31c3 talk had a live demo with a kernel module modifying source code in memory only
- financial incentives to crack developer machines or a projects build infrastructure...
- how can you be sure what's running on your machine or on a build daemon network? Do you ever leave your USB3 ports alone?



A few examples from that 31c3 talk

- CVE-2002-0083: remote root exploit in sshd, a single bit difference in the binary
- 31c3 talk had a live demo with a kernel module modifying source code in memory only
- financial incentives to crack developer machines or a projects build infrastructure...
- how can you be sure what's running on your machine or on a build daemon network? Do you ever leave your computers alone?



Another example from real life

At a CIA conference in 2012:

[edit] (S//NF) Strawhorse: Attacking the MacOS and iOS Software Development Kit

(S) Presenter: [REDACTED], Sandia National Laboratories

(S//NF) Ken Thompson's gcc attack (described in his 1984 Turing award acceptance speech) motivates the StrawMan work: **what can be done** of benefit to the US Intelligence Community (IC) **if one can make an arbitrary modification to a system compiler** or Software Development Kit (SDK)? A (whacked) SDK can provide a subtle injection vector onto standalone developer networks, or it can modify any binary compiled by that SDK. **In the past, we have watermarked binaries for attribution, used binaries as an exfiltration mechanism, and inserted Trojans into compiled binaries.**

(S//NF) In this talk, we discuss our explorations of the Xcode (4.1) SDK. Xcode is used to compile MacOS X applications and kernel extensions as well as iOS applications. We describe how we use (our whacked) Xcode to do the following things: -Entice all MacOS applications to create a remote backdoor on execution -Modify a dynamic dependency of securityd to load our own library - which rewrites securityd so that no prompt appears when exporting a developer's private key -Embed the developer's private key in all iOS applications -Force all iOS applications to send embedded data to a listening post -Convince all (new) kernel extensions to disable ASLR

(S//NF) We also describe how we modified both the MacOS X updater to install an extra kernel extension (a keylogger) and the Xcode installer to include our SDK whacks.

[firstlook.org/theintercept/2015/03/10/
ispy-cia-campaign-steal-apples-secrets/](http://firstlook.org/theintercept/2015/03/10/ispy-cia-campaign-steal-apples-secrets/)



The solution

Promise that anyone can always generate identical binary packages from a given source



The solution

We call this:

“Reproducible builds”



Demo



Demo

- Build a package 5 times, get 5 .debs with different checksums



Demo

- Build a package 5 times, get 5 .debs with different checksums
- Build a package 5 times, get 5 .debs with the same checksum





This should become the
norm.



This should become the **norm.**

We want to change the meaning of "free software":
it's only free software if it's reproducible!

More benefits than "just" security...

- smaller deltas, thus faster updates possible
- in Debian: lots of QA benefits
- Google does reproducible builds, to save money
- ...





1

Motivation

2

Common ressources

3

Status Debian

4

Status Non-Debian World

5

Future work

6

Getting involved

7

Questions, comments, ideas?

reproducible-builds.org

- <https://reproducible-builds.org>

reproducible-builds.org

Provide a verifiable path from source code to binary.

What is it
about?

Reproducible builds are a set of software development practices which create a **verifiable path from** human readable **source code** to the **binary** code used by computers.

Why does
it matter?

Most aspect of software verification is done on source code, as that is what humans can reasonably understand. But most of the time, computers require software to be first built into long string of numbers to be used. With *reproducible builds*, multiple parties can **redo this process independently** and ensure they **all get exactly the same result**. We can thus **grow confidence** than a



Documentation about common problems

- <https://reproducible-builds.org/docs>
- Lunar's talk from CCCamp 2015 also on <https://media.ccc.de>

Avoid (true) randomness

- Randomness is not deterministic

```
int getRandomNumber()
{
    return 4; // chosen by fair dice roll
            // guaranteed to be random
}
```

XXKD #221

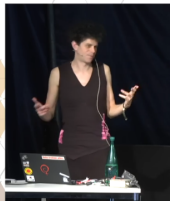
Example

```
$ gcc -flto -c utils.c
$ nm -a utils.o | grep inline
0000000000000000 n .gnu.lto_.inline.381a277a0b6d2a35
```

CCCamp15 29 / 55



CHAOS
COMMUNICATION
CAMP 2015



Common problems

- time stamps



Common problems

- time stamps
- timezones
- locales



Common problems

- time stamps
- timezones
- locales
- everything else (seperated into known issues and the blurry rest)



SOURCE_DATE_EPOCH

- Build date (timestamps) usually not useful for the user
- SOURCE_DATE_EPOCH is defined as the last modification of the source, since the epoch (1970-01-01)
- can be used instead of current date
- can also be used for random seeds etc.
- in Debian, set from the latest debian/changelog entry



SOURCE_DATE_EPOCH

- Build date (timestamps) usually not useful for the user
- SOURCE_DATE_EPOCH is defined as the last modification of the source, since the epoch (1970-01-01)
- can be used instead of current date
- can also be used for random seeds etc.
- in Debian, set from the latest debian/changelog entry
- many upstreams support it already
- has been adopted by other distributions (NetBSD, FreeBSD, Arch Linux, Guix, ...)



SOURCE_DATE_EPOCH

- SOURCE_DATE_EPOCH spec available:
- <https://reproducible-builds.org/specs/>



- Continuously testing Debian testing, unstable and experimental
- Also testing: coreboot, OpenWrt, NetBSD, FreeBSD, Arch Linux, Fedora and soon F-Droid and Guix too



tests.reproducible-builds.org

- Continuously testing Debian testing, unstable and experimental
- Also testing: coreboot, OpenWrt, NetBSD, FreeBSD, Arch Linux, Fedora and soon F-Droid and Guix too
- 230 jenkins jobs running on seven amd64 nodes with 120 cores and 300 GB ram combined and 15 small armhf nodes...
- 42 scripts in Python and Bash, 282 lines of code in average
- 29 contributors for `jenkins.debian.net.git`



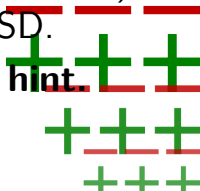
Variations (when testing Debian)

variation	first build	second build
hostname	jenkins	i-capture-the-hostname
domainname	debian.net	i-capture-the-domainname
env TZ	GMT+12	GMT-14
env LANG	C	fr_CH.UTF-8
env LC_ALL	not set	fr_CH.UTF-8
env USER	pbuilder1	pbuilder2
uid	1111	2222
gid	1111	2222
UTS namespace	shared with the host	<i>modified using /usr/bin/unshare --uts</i>
kernel version	Linux 3.16 or 4.X	on amd64 always varied, on armhf sometimes
umask	0022	0002
CPU type	same for both builds on amd64 (<i>work in progress</i>) on armhf varied a bit	
filesystem	same for both builds on amd64: (tmpfs), on armhf ext3/4 (<i>and we have disorderfs, but the code is disabled</i>)	
year, month, date	on amd64: 398 days variation, on armhf not yet	
hour, minute	hour is usually the same... usually, the minute differs...	
everything else	<i>is likely the same...</i>	



Debugging problems: diffoscope

- Examines differences **in depth**.
- Outputs HTML or plain text with human readable differences.
- Recursively unpacks archives, uncompresses PDFs, disassembles binaries, unpacks Gettext files, ...
- Easy to extend to new file formats.
- Falls back to binary comparison.
- Available from git, PyPI, Debian (sid and stretch), Arch Linux, Guix, Homebrew. Works on BSD.
- Maintainers in other distros wanted. **hint, hint.**
- <https://diffoscope.org/>



diffoscope example (HTML output)

51431INSERT INTO "targets" VALUES ('ttu.ee', 13611);	51438INSERT INTO "targets" VALUES ('ttu.ee', 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 ('cd09fb8c2161a8d1280b848eaab3b14d35fe3044');	60756INSERT INTO "git_commit" VALUES ('e78fe5d803208bf6c877dc675cdb4f1b719e7519');
60736COMMIT;	60757COMMIT;

install.rdf

Offset 5, 15 lines modified

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

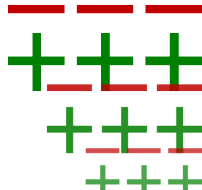
Offset 5, 15 lines modified

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



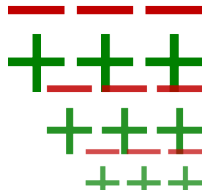
Try diffoscope

- <https://try.diffoscope.org>



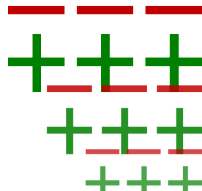
diffoscope is "just" for debugging

- Reminder: diffoscope is for **debugging**



diffoscope is "just" for debugging

- Reminder: diffoscope is for **debugging**
- "reproducible" according to our definition means: **bit by bit identical**. So the tools for testing whether something is reproducible are either `diff` or `sha256sum`!





1

Motivation

2

Common ressources

3

Status Debian

4

Status Non-Debian World

5

Future work

6

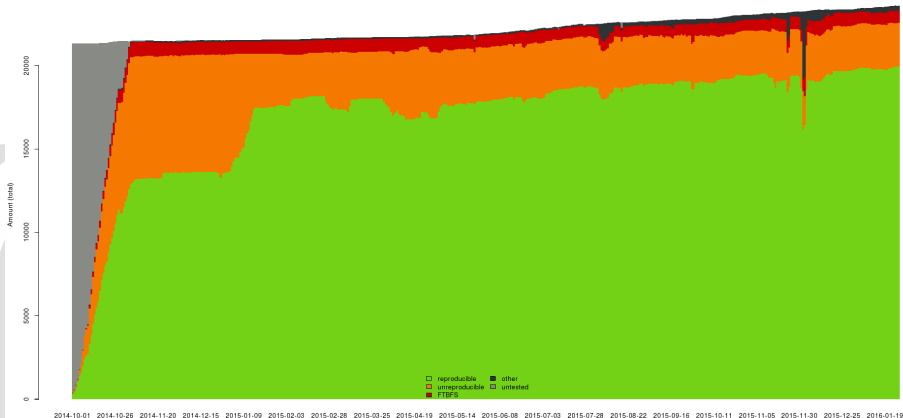
Getting involved

7

Questions, comments, ideas?

Progress in Debian unstable

Reproducibility status for packages in 'unstable' for 'amd64'



20,079 (85%) out of 23,595 source packages are reproducible
in our test framework



Notes and issues on tests.reproducible-builds.org

- 179 categorised distinct issues
- 3,792 notes



Notes and issues on tests.reproducible-builds.org

- 179 categorised distinct issues
- 3,792 notes
- 2549 unreproducible packages in sid, but only 139 without a note
- 728 packages failing to build, but only 74 without a note



Notes and issues on tests.reproducible-builds.org

- 179 categorised distinct issues
- 3,792 notes
- 2549 unreproducible packages in sid, but only 139 without a note
- 728 packages failing to build, but only 74 without a note
- maintained in `notes.git`



Notes and issues on tests.reproducible-builds.org

- 179 categorised distinct issues
- 3,792 notes
- 2549 unreproducible packages in sid, but only 139 without a note
- 728 packages failing to build, but only 74 without a note
- maintained in `notes.git`
- currently Debian only, but cross distro notes are planned



Debian packages on tests.reproducible-builds.org

- `https://reproducible.debian.net/\$src`

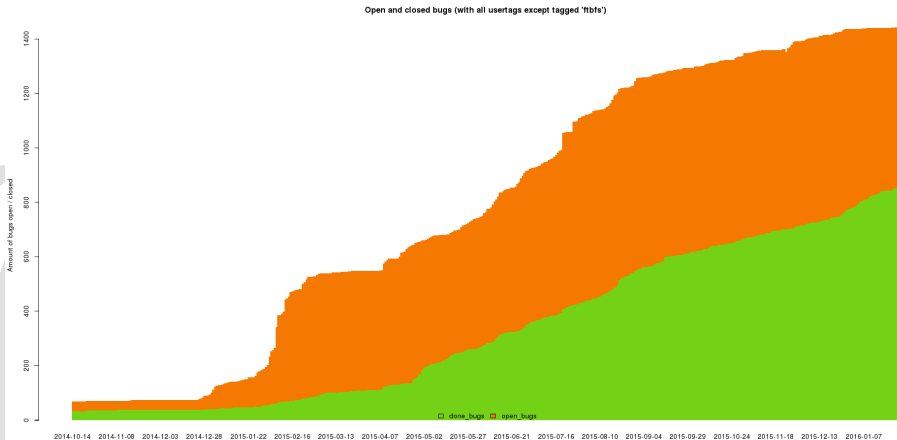


Debian packages on tests.reproducible-builds.org

- `https://reproducible.debian.net/\$src`
- package sets



Progress in the Debian bug tracker



As a rule, we file bugs with patches.
There are very few exceptions.



Tell the world & collaborate

- Weekly reports since May 2015



Tell the world & collaborate

- Weekly reports since May 2015
- First Reproducible World Summit in December 2015 (Athens, Greece)
 - ▶ 40 people from 16 projects
 - ▶ `reproducible.debian.net` has become `tests.reproducible-builds.org`



Tell the world & collaborate

- Weekly reports since May 2015
- First Reproducible World Summit in December 2015 (Athens, Greece)
 - ▶ 40 people from 16 projects
 - ▶ `reproducible.debian.net` has become `tests.reproducible-builds.org`
 - ▶ Planning for next summit, probably in May/June 2016, has just begun
 - ▶ Talk to me if you want to attend.



Tell the world & collaborate

- Weekly reports since May 2015
- First Reproducible World Summit in December 2015 (Athens, Greece)
 - ▶ 40 people from 16 projects
 - ▶ `reproducible.debian.net` has become `tests.reproducible-builds.org`
 - ▶ Planning for next summit, probably in May/June 2016, has just begun
 - ▶ Talk to me if you want to attend.
- "We don't care about Debian (only), we care about free and open source software."



Reminder / Summary

- This is just a proof-of-concept, Debian is not 85% reproducible



Reminder / Summary

- This is just a proof-of-concept, Debian is not 85% reproducible
- Debian "unstable" this summer?!!



Reminder / Summary

- This is just a proof-of-concept, Debian is not 85% reproducible
- Debian "unstable" this summer?!!
- I hope that Debian 9, "stretch", will be partially reproducible in a meaningful way, in 2017.



Reminder / Summary

- This is just a proof-of-concept, Debian is not 85% reproducible
- Debian "unstable" this summer?!!
- I hope that Debian 9, "stretch", will be partially reproducible in a meaningful way, in 2017.
- Will Debian 10, "buster", be 100% reproducible?



Reminder / Summary

- This is just a proof-of-concept, Debian is not 85% reproducible
- Debian "unstable" this summer?!!
- I hope that Debian 9, "stretch", will be partially reproducible in a meaningful way, in 2017.
- Will Debian 10, "buster", be 100% reproducible?
- What's beyond (rebuilding, .buildinfo file handling, user tools) mostly still needs *design and code*



A vintage television set with a dark frame is shown against a stone wall. The screen displays a black and white image of a protest. A large, light-colored sign is held up, featuring the text 'The Whole WORLD'S WATCHING' in a bold, hand-painted font. The sign is tilted. In the background, a dark, angular structure, possibly a building or a bridge, is visible against a light sky. A silhouette of a person holding the sign is partially visible at the bottom of the sign. The television has a small 'DUMHI' logo at the bottom center of its frame.

The
Whole
WORLD'S
WATCHING

DUMHI

- 
- 1 Motivation
 - 2 Common ressources
 - 3 Status Debian
 - 4 Status Non-Debian World**

- 5 Future work

- 6 Getting involved

- 7 Questions, comments, ideas?

Skipped stati

- <https://tests.r-b.org/coreboot>
- <https://tests.r-b.org/openwrt>
- <https://tests.r-b.org/netbsd>
- <https://tests.r-b.org/freebsd>
- <https://tests.r-b.org/archlinux>
- not yet: <https://tests.r-b.org/f-droid>
- not yet: <https://tests.r-b.org/guix>

Status Fedora

- <https://tests.r-b.org/fedora> (23)
- maintained by Dhiru Kholia and h01ger
- rpm repo available by Dhiru, but still **0% reproducible**
- rpm format includes build time, build host and embedded signature
- recreating the build env: koji
- next: 24+rawhide
- next: get more people involved - **you ?**
- next: first reproducible rpm
- next: use koji



Unmentioned, with known activities

- Bitcoin (2011)
- Tor (2013)
- NixOS
- Qubes
- few commercial, proprietary Software
- ?



1

Motivation

2

Common ressources

3

Status Debian

4

Status Non-Debian World

5

Future work

6

Getting involved

7

Questions, comments, ideas?

Rebuilders and sharing signed checksums

- Almost no work has been done here yet. We are just at the first step: being able to rebuild reproducibly...



Rebuilders and sharing signed checksums

- Almost no work has been done here yet. We are just at the first step: being able to rebuild reproducibly...
- Different projects, different solutions?



Rebuilders and sharing signed checksums, cont.

- Individually signed checksums (think web of trust) could work in the Debian case (we have a gpg web of trust), but IMO won't scale.



Rebuilders and sharing signed checksums, cont.

- Individually signed checksums (think web of trust) could work in the Debian case (we have a gpg web of trust), but IMO won't scale.
- Another idea: rebuilders, run by large organisations (ACLU, CCC, CERN, Deutsche Bank, EDF, EON, Greenpeace, NASA, NSA, XYZ).



Rebuilders and sharing signed checksums, cont.

- Individually signed checksums (think web of trust) could work in the Debian case (we have a gpg web of trust), but IMO won't scale.
- Another idea: rebuilders, run by large organisations (ACLU, CCC, CERN, Deutsche Bank, EDF, EON, Greenpeace, NASA, NSA, XYZ).
- Fedora rebuilds Debian, Debian rebuilds OpenSUSE, OpenSUSE rebuilds NetBSD, etc...



Rebuilders and sharing signed checksums, cont.

- Individually signed checksums (think web of trust) could work in the Debian case (we have a gpg web of trust), but IMO won't scale.
- Another idea: rebuilders, run by large organisations (ACLU, CCC, CERN, Deutsche Bank, EDF, EON, Greenpeace, NASA, NSA, XYZ).
- Fedora rebuilds Debian, Debian rebuilds OpenSUSE, OpenSUSE rebuilds NetBSD, etc...
- Big customers could just rebuild everything themselves.



Integration in user tools

- "Do you really want to install this unreproducible software (y/N)"



Integration in user tools

- "Do you really want to install this unreproducible software (y/N)"
- "Do you want to build those packages which unconfirmed checksums, before installing? (Y/n)"



Integration in user tools

- "Do you really want to install this unreproducible software (y/N)"
- "Do you want to build those packages which unconfirmed checksums, before installing? (Y/n)"
- "How many signed checksums do you require to call a package 'reproducible'?"



Integration in user tools

- "Do you really want to install this unreproducible software (y/N)"
- "Do you want to build those packages which unconfirmed checksums, before installing? (Y/n)"
- "How many signed checksums do you require to call a package 'reproducible'?"
- "Which rebuilders do you want to trust?"





1

Motivation

2

Common ressources

3

Status Debian

4

Status Non-Debian World

5

Future work

6

Getting involved

7

Questions, comments, ideas?

As a software developer

- Stop using build dates
- Use `SOURCE_DATE_EPOCH` instead
- See <https://reproducible-builds.org/specs/>



Getting involved - learning by doing

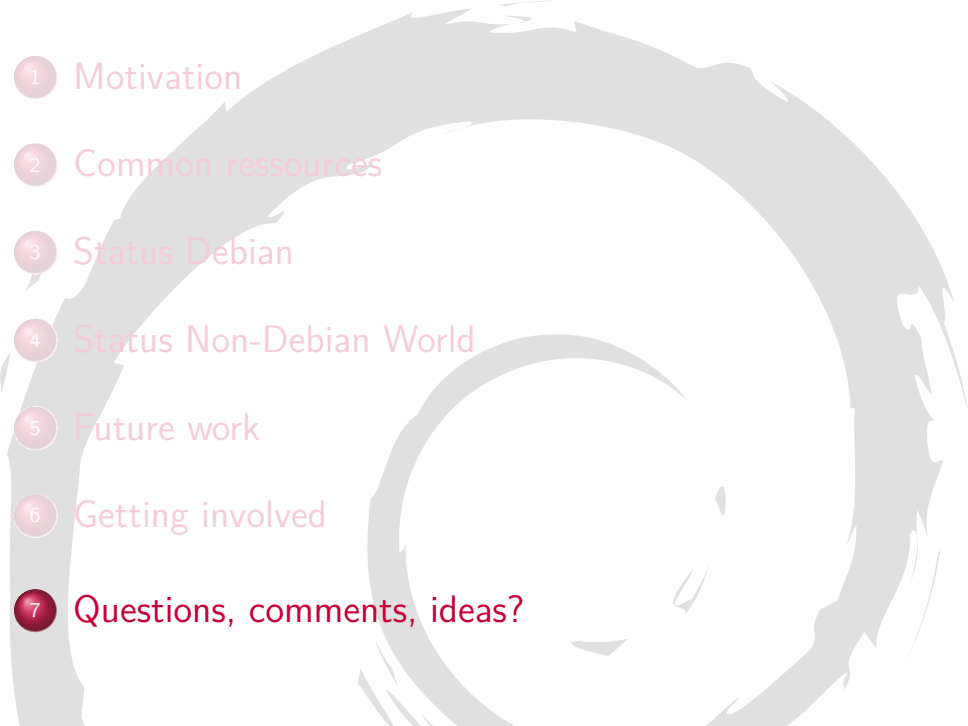
- Test for yourself:
 - ▶ Build something twice, run diffoscope on the results
 - ★ For better results use our “reproducible” repository, pbuilder and a custom config
- Docs on the web:
<https://reproducible-builds.org/docs/>
<https://wiki.debian.org/ReproducibleBuilds/ExperimentalToolchain>
- Ask for help on #reproducible-builds or on mailing list



Form the Fedora reproducible builds team!

- Why?
 - ▶ Every distribution should be reproducible!
 - ▶ Learn something new everyday
 - ▶ Change the (software) world!
 - ▶ <https://tests.reproducible-builds.org/fedora> needs **your** help
- How to get started?
 - ▶ Talk to me here or talk to us on IRC or via mail.
 - ▶ RTFM, there is lots of documentation
 - ▶ Experiment - learning by doing
 - ▶ Use+debug existing patches to rpm-build



- 
- 1 Motivation
 - 2 Common ressources
 - 3 Status Debian
 - 4 Status Non-Debian World
 - 5 Future work
 - 6 Getting involved
 - 7 Questions, comments, ideas?

Questions, comments, ideas?

- <https://reproducible-builds.org/docs>
- <https://tests.reproducible-builds.org>
- #reproducible-builds on irc.0FTC.net



Questions, comments, ideas?

- <https://reproducible-builds.org/docs>
- <https://tests.reproducible-builds.org>
- #reproducible-builds on irc.OFTC.net
- <https://lists.reproducible-builds.org>
- <https://twitter.com/ReproBuild>



Questions, comments, ideas?

- <https://reproducible-builds.org/docs>
- <https://tests.reproducible-builds.org>
- #reproducible-builds on irc.OFTC.net
- <https://lists.reproducible-builds.org>
- <https://twitter.com/ReproBuild>
- Mike and Seth's talk from 31c3 about motivations
- Lunar's talk about fixing reproducible issues from CCCamp 15
- my talk "the Reproducible builds ecosystem" from FOSDEM 16



Thanks to...! ...and thank **you**, too!

- Debian “Reproducible Builds” team
(you are just **so** awesome!)
- Linux Foundation and the Core Infrastructure Initiative



holger@debian.org B8BF 5413 7B09 D35C F026
FE9D 091A B856 069A AA1C



Thanks to...! ...and thank **you**, too!

- All “Reproducible Builds” teams
(you are just **so** awesome!)
- Linux Foundation and the Core Infrastructure Initiative



holger@debian.org B8BF 5413 7B09 D35C F026
FE9D 091A B856 069A AA1C



Copyright © 2014–2016
Holger Levsen holger@layer-acht.org and others.

Copyright of images included in this document are 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 <http://creativecommons.org/licenses/by-sa/3.0/> or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.

The source of this document is available from
<https://anonscm.debian.org/git/reproducible/presentations.git>.

