

Reproducible builds ecosystem

Where some of us are
and some hints where this might be going...

Holger 'h01ger' Levsen
holger@layer-acht.org

openSUSE Conference 2016 (Nürnberg, DE)
2016-06-23



about me

- B8BF 5413 7B09 D35C F026 FE9D 091A B856 069A AA1C
- Debian user since 1995 - though my very first installation was SuSE :)
- 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)

more about me

- B8BF 5413 7B09 D35C F026 FE9D 091A B856 069A AA1C
- 8F03 B243 8719 BA6B 1A35 0EB6 40C2 DEA2 F56C 7256
- Debian Reproducible builds team member
 - ▶ within in the team I'm mostly working on <https://tests.reproducible-builds.org>
- until April 2016 together with Lunar funded by the Linux Foundation
 - ▶ applied for extended funding in April 2016...



more about me

- B8BF 5413 7B09 D35C F026 FE9D 091A B856 069A AA1C
- 8F03 B243 8719 BA6B 1A35 0EB6 40C2 DEA2 F56C 7256
- Debian Reproducible builds team member
 - ▶ within in the team I'm mostly working on <https://tests.reproducible-builds.org>
- until April 2016 together with Lunar funded by the Linux Foundation
 - ▶ applied for extended funding in April 2016...
- basically no idea about Reproducible SUSE ;-)



Debian reproducible builds team

akira
Alexis Bienvenüe
Andrew Ayer
Asheesh Laroia
Ceridwen
Chris Lamb
Chris West
Christoph Berg
Daniel Kahn Gillmor
Daniel Shahaf
David Suarez
Dhole
Drew Fisher
Esa Peuha
Fabian Wolff

Guillem Jover
Hans-Christoph Steiner
Helmut Grohne
Holger Levsen
HW42
Intrigeri
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
Santiago Vila
Sascha Steinbiss
Satyam Zode
Scarlett Clark
Stefano Rivera
Stéphane Glondu
Steven Chamberlain
Tom Fitzhenry
Valerie Young
Valentin Lorentz
Wookey
Ximin Luo



Debian reproducible builds team

akira
Alexis Bienvenüe
Andrew Ayer
Asheesh Laroia
Ceridwen
Chris Lamb
Chris West
Christoph Berg
Daniel Kahn Gillmor
Daniel Shahaf
David Suarez
Dhole
Drew Fisher
Esa Peuha
Fabian Wolff

Guillem Jover
Hans-Christoph Steiner
Helmut Grohne
Holger Levsen
HW42
Intrigeri
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
Santiago Vila
Sascha Steinbiss
Satyam Zode
Scarlett Clark
Stefano Rivera
Stéphane Glondu
Steven Chamberlain
Tom Fitzhenry
Valerie Young
Valentin Lorentz
Wookey
Ximin Luo



jenkins.debian.net.git contributors

akira

Alexander Couzens

Levente 'anthraxx' Polyak

Antonio Terceiro

Axel Beckert

Bryan Newbold

Chris Lamb

Daniel Kahn Gillmor

Gabriele Giacone

Hans-Christoph Steiner

Helmut Grohne

Holger Levsen

HW42

James McCoy

Joachim Breitner

Johannes 'josch' Schauer

Jérémy Bobbio

Mattia Rizzolo

Niels Thykier

Paul Wise

Petter Reinholdtsen

Philip Hands

Reiner Herrmann

Samuel Thibault

Steven Chamberlain

Tails developers

Ulrike Uhlig

Wolfgang Schweer

Wouter Verhelst



jenkins.debian.net.git contributors

akira

Alexander Couzens

Levente 'anthraxx' Polyak

Antonio Terceiro

Axel Beckert

Bryan Newbold

Chris Lamb

Daniel Kahn Gillmor

Gabriele Giacone

Hans-Christoph Steiner

Helmut Grohne

Holger Levsen

HW42

James McCoy

Joachim Breitner

Johannes 'josch' Schauer

Jérémy Bobbio

Mattia Rizzolo

Niels Thykier

Paul Wise

Petter Reinholdtsen

Philip Hands

Reiner Herrmann

Samuel Thibault

Steven Chamberlain

Tails developers

Ulrike Uhlig

Wolfgang Schweer

Wouter Verhelst



Who are you?

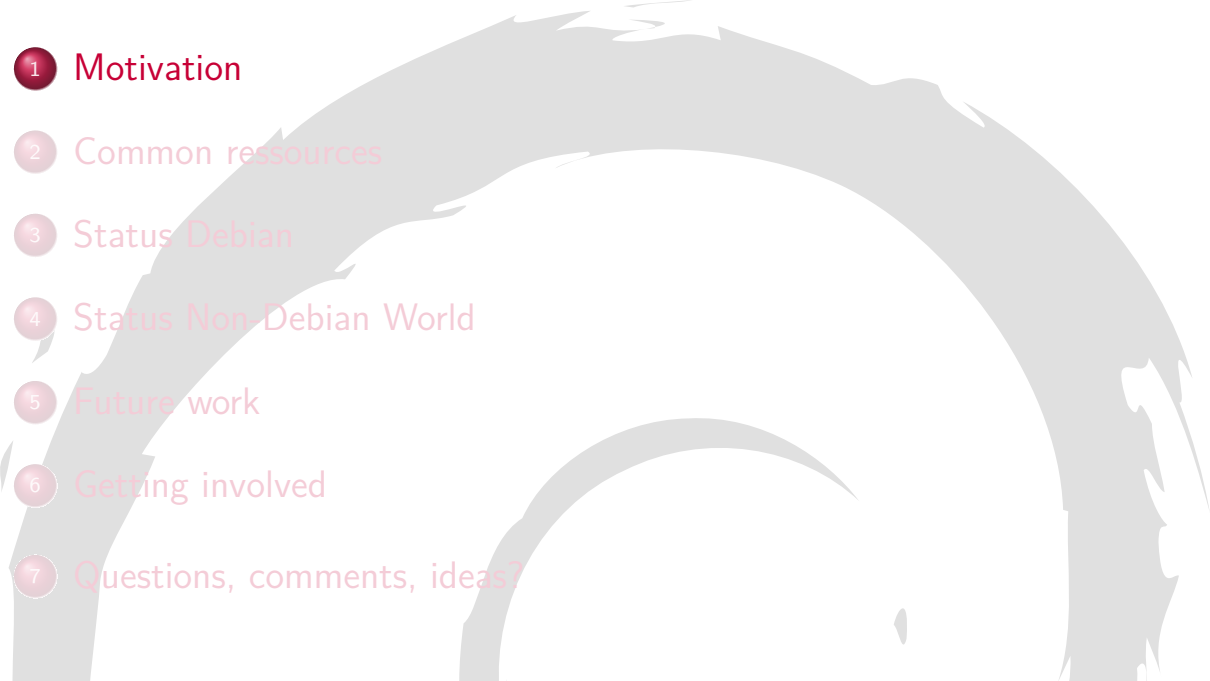
- Contributed to Free Software?



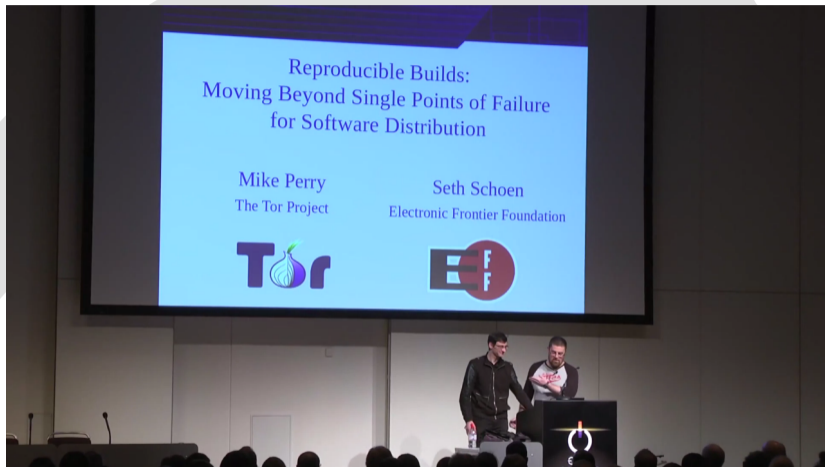
Who are you?

- Contributed to Free Software?
- Seen a talk about reproducible builds?



- 
- 1 Motivation
 - 2 Common resources
 - 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... attack one, 0wn millions.



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... attack one, 0wn millions.
- How can you be sure what's running on your machine or on a build daemon network? Are your computers really always physically safe?



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... attack one, 0wn millions.
- How can you be sure what's running on your machine or on a build daemon network? Are your computers really always physically safe?
- Hacking OBS is very affordable for state sponsored attackers and large criminal organisations and AIUI would expose all openSuSE installations. You are a target because your customers are.



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: ██████████ 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/

Another example from real life: XCode Ghost

At a CIA conference in 2012:

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

(S) Presenter: ██████████, 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/

Summary: the source of the problem...

- Free Software is great!
 - ▶ Use
 - ▶ Share
 - ▶ Study
 - ▶ Modify



Summary: the source of the problem...

- Free Software is great!
 - ▶ Use
 - ▶ Share
 - ▶ Study
 - ▶ Modify
- Free Software is about source code, but noone uses the sources, we all use binaries.



The solution

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



The solution

We call this:

“Reproducible builds”



Demo





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!

This should become the **norm**.

We want to change the meaning of "free software":

it's only free software if it's reproducible!

Because one can only be sure it's free software
if it's reproducible!

a very brief history

- ...
- Bitcoin (2012)
- TorBrowser and Tor (2012)



a very brief history

- ...
- Bitcoin (2012)
- TorBrowser and Tor (2012)
- Debian (2013)
- FreeBSD (2013)



a very brief history

- ...
- Bitcoin (2012)
- TorBrowser and Tor (2012)
- Debian (2013)
- FreeBSD (2013)
- reproducible.debian.net (2014)
- 2015:
 - ▶ 15 talks given
 - ▶ reproducible-builds.org
 - ▶ meeting in Athens with 16 projects



a very brief history

- ...
- Bitcoin (2012)
- TorBrowser and Tor (2012)
- Debian (2013)
- FreeBSD (2013)
- reproducible.debian.net (2014)
- 2015:
 - ▶ 15 talks given
 - ▶ reproducible-builds.org
 - ▶ meeting in Athens with 16 projects



- 
- 1 Motivation
 - 2 Common resources**
 - 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 verify this process.



tests.reproducible-builds.org



tests.reproducible-builds.org

- Continuously testing Debian testing, unstable and experimental



tests.reproducible-builds.org

- Continuously testing Debian testing, unstable and experimental
- on amd64 and i386 and armhf



tests.reproducible-builds.org

- Continuously testing Debian testing, unstable and experimental
- on amd64 and i386 and armhf
- Also testing: OpenWrt, coreboot, NetBSD, FreeBSD.
- Arch Linux, Fedora and F-Droid work in progress...



tests.reproducible-builds.org

- Continuously testing Debian testing, unstable and experimental
- on amd64 and i386 and armhf
- Also testing: OpenWrt, coreboot, NetBSD, FreeBSD.
- Arch Linux, Fedora and F-Droid work in progress...
- testing = build, vary the environment, build again, compare results.



tests.reproducible-builds.org

- Continuously testing Debian testing, unstable and experimental
- on amd64 and i386 and armhf
- Also testing: OpenWrt, coreboot, NetBSD, FreeBSD.
- Arch Linux, Fedora and F-Droid work in progress...
- testing = build, vary the environment, build again, compare results.
- 311 jenkins jobs running on 31 hosts
- 41 scripts with a total of 4k lines of Python and 6k lines of Bash Shell
- 31 contributors for `jenkins.debian.net.git`



tests.reproducible-builds.org

- Continuously testing Debian testing, unstable and experimental
- on amd64 and i386 and armhf
- Also testing: OpenWrt, coreboot, NetBSD, FreeBSD.
- Arch Linux, Fedora and F-Droid work in progress...
- testing = build, vary the environment, build again, compare results.
- 311 jenkins jobs running on 31 hosts
- 41 scripts with a total of 4k lines of Python and 6k lines of Bash Shell
- 31 contributors for `jenkins.debian.net.git`
- static webpages, results available as JSON too



CPU architectures on tests.r-b.org

- amd64 and i386: 106 cores and 282 GB RAM split on 9 VMs
- most resources used for testing Debian...
- sponsored by <https://profitbricks.co.uk> since 2014 (2012)



- armhf: 21 nodes with 80 cores and 41 GB RAM sponsored by Debian
- arm64: coming soon

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/gid	1111	2222
shell	dash	bash
UTS namespace	shared with the host	<i>modified using /usr/bin/unshare --uts</i>
kernel version	Linux 3.16 or 4.X	on amd64 and i386 always varied, on armhf sometimes
32 vs 64 bit kernel	one or the other	only varied on i386
umask	0022	0002
CPU type	Intel and AMD variation for i386 and amd64 (<i>work in progress</i>) on armhf varied a bit	
filesystem	same for both builds on amd64: (tmpfs), on i386 and armhf ext3/4 (<i>and we have disorderfs, but the code is disabled</i>)	
year, month, date	on amd64 and i386: 398 days variation, on armhf not yet	
hour, minute	hour is usually the same... usually, the minute differs...	
<i>everything else</i>	<i>is likely the same...</i>	



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)



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

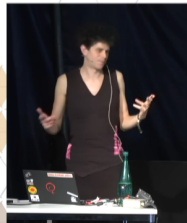
```
int getRandomNumber()
{
    return 4; // chosen by fair dice roll
            // guaranteed to be random.
}
```

XKCD #221

Example

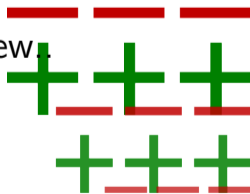
```
$ gcc -flto -c utils.c
$ nm -a utils.o | grep inline
0000000000000000 n .gnu.lto_.inline.381a277a0b6d2a35
```

CCCamp15 29 / 59



Debugging problems: difoscope

- 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), Fedora, Arch Linux, FreeBSD, NetBSD, Guix, Homebrew.
- Maintainers (upstream and in other distros) wanted.
- <https://difoscope.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( 'cd09fb8c2161a 60756INSERT INTO "git_commit" VALUES( 'e78fe5d803208
8d1280b848eaab3b14d35fe3044 ' ); bf6c877dc675cdb4f1b719e7519 ' );
60736COMMIT; 60757COMMIT;
```

install.rdf

Offset 5, 15 lines modified

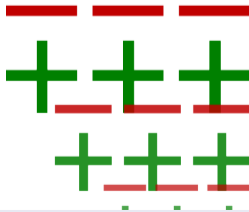
```
5 .....<Description about="urn:mozilla:install-
manifest">
6 .....<em:name>HTTPS-Everywhere</em:name>
7 .....<em:creator>Mike Perry, Peter Eckersley,
&amp; Yan Zhu</em:creator>
8 .....<em:aboutURL>chrome://https-everywhere/
content/about.xul</em:aboutURL>
9 .....<em:id>https-everywhere@eff.org</em:id>
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>
```

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,
&amp; 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>
```

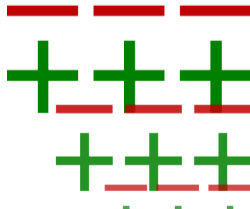

Try diffoscope

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



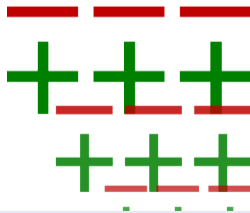
diffoscope is "just" for debugging

- 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`!



SOURCE_DATE_EPOCH

- Build date (timestamps) usually not useful for the user



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)
- SOURCE_DATE_EPOCH can be used instead of current date
- can also be used for random seeds etc.



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)
- SOURCE_DATE_EPOCH can be used instead of current date
- can also be used for random seeds etc.
- in Debian, set from the latest debian/changelog entry
- solution has been adopted by other projects & distributions (NetBSD, FreeBSD, Arch Linux, Guix, Fedora...)



SOURCE_DATE_EPOCH (closed bugs)

- dh-strip-nondeterminism
- gcc (`__DATE__` and `__TIME__` macros)

<https://gcc.gnu.org/ml/gcc-patches/2015-06/msg02210.html>

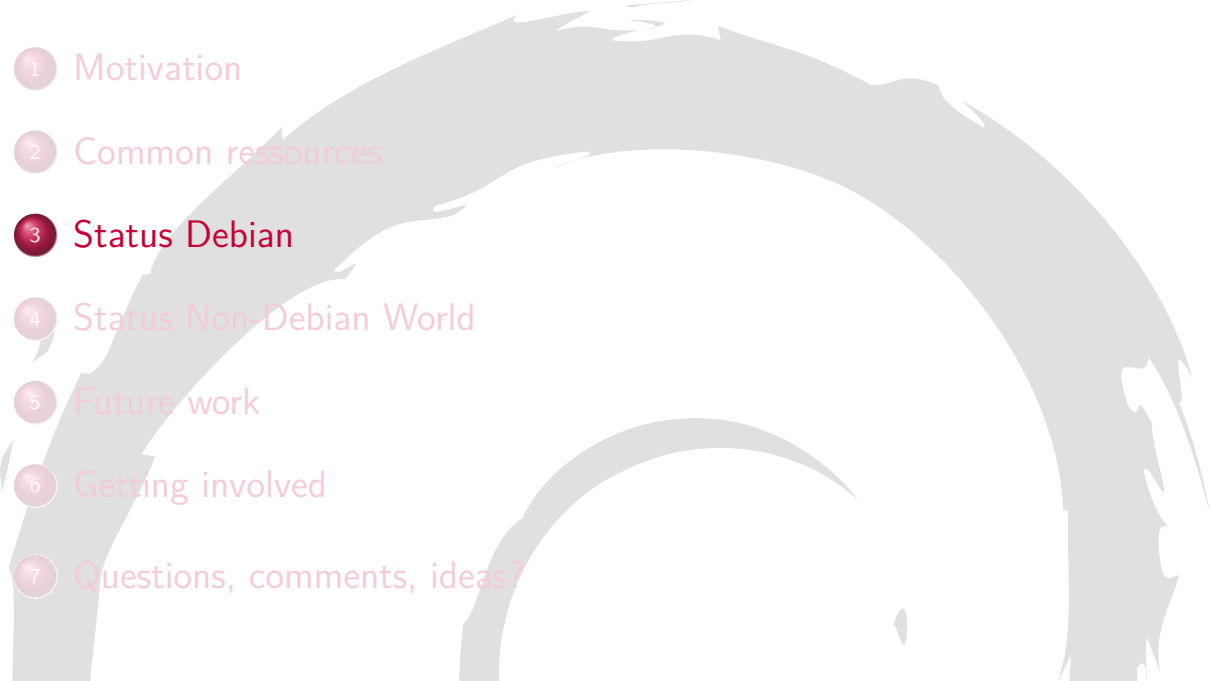
- #791823: debhelper
- #787444: help2man
- #790899: epydoc
- #794004: ghostscript
- #796130: man2html
- #783475: texi2html
- #794586: ocaml-doc
- #792202: texlive-bin
- ...



SOURCE_DATE_EPOCH

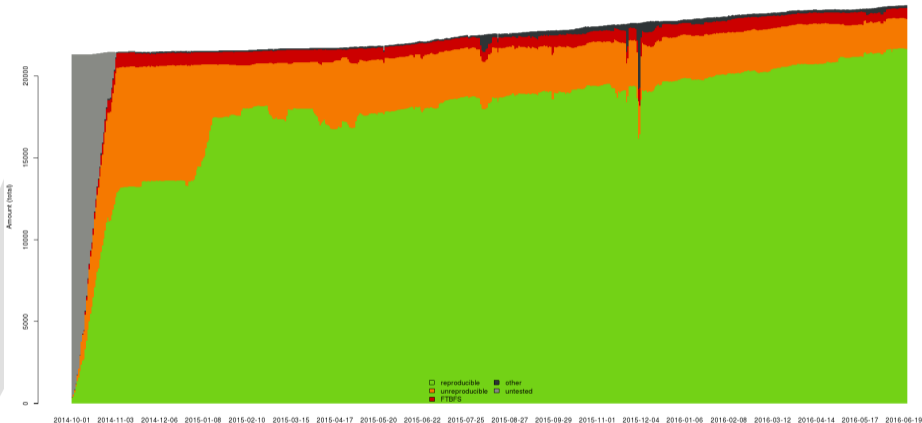
- SOURCE_DATE_EPOCH spec available
- <https://reproducible-builds.org/specs/>



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

Progress in Debian unstable/amd64

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



21,666 (89.0%) out of 24,323 source packages are reproducible
in our test framework (and 90.2% in testing/amd64)



Notes and issues on tests.reproducible-builds.org

- 206 categorised distinct issues
- 3,261 notes



Notes and issues on tests.reproducible-builds.org

- 206 categorised distinct issues
- 3,261 notes
- 1844 unreproducible packages in sid/amd64, but only 211 without a note
- 655 packages failing to build, but only 149 without a note



Notes and issues on tests.reproducible-builds.org

- 206 categorised distinct issues
- 3,261 notes
- 1844 unreproducible packages in sid/amd64, but only 211 without a note
- 655 packages failing to build, but only 149 without a note
- maintained in `notes.git`



Notes and issues on tests.reproducible-builds.org

- 206 categorised distinct issues
- 3,261 notes
- 1844 unreproducible packages in sid/amd64, but only 211 without a note
- 655 packages failing to build, but only 149 without a note
- maintained in `notes.git`
- currently Debian only, but we will turn those into cross distro (and upstream) notes



examples of issues from notes.git

- timestamps_in_cpio_archive
- randomness_in_ocaml_provides
- leaks_path_environment_variable
- python-ply_compiled_parse_tables
- timestamps_in_documentation_generated_by_docbook_dbtimes-tamp
- plist_weirdness
- timestamps_generated_by_eigenbase_resgen
- different_due_to_umask
- timestamps_in_manpages_generated_by_docbook_utils
- different_pot_creation_date_in_gettext_mo_files
- ftbfs_uninvestigated_test_failures



examples of issues from notes.git

- `unsorted_file_glob_by_cmake`
- `timestamps_generated_by_mangosdk_spiprocessor`
- `randomness_in_icc_colour_profiles`
- `undeterministic_symlinking_by_rdfind`
- `random_order_in_ruby_rdoc_indices`
- `timestamps_in_edj_files_generated_by_edje_cc`
- `timestamps_in_documentation_generated_by_asciidocctor`
- `timestamps_in_zip`
- `random_order_in_plexus_comonents_xml`
- `timestamps_in_png`
- `random_order_in_documentation_generated_by_javadoc`

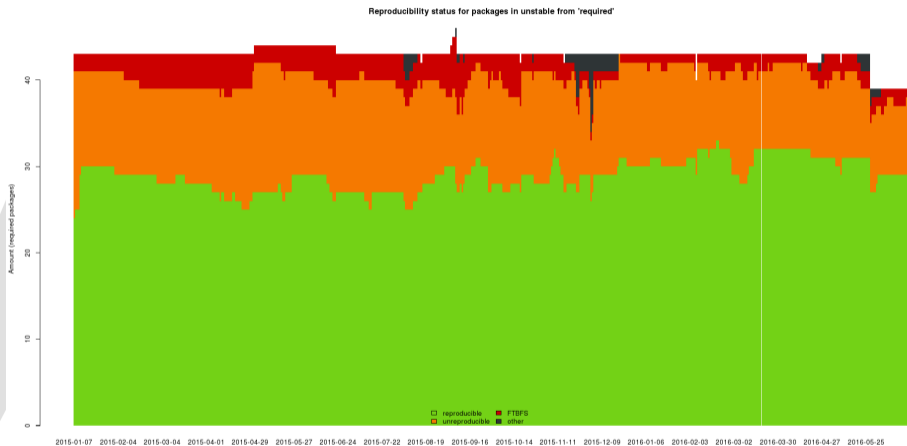


Debian packages on tests.reproducible-builds.org

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



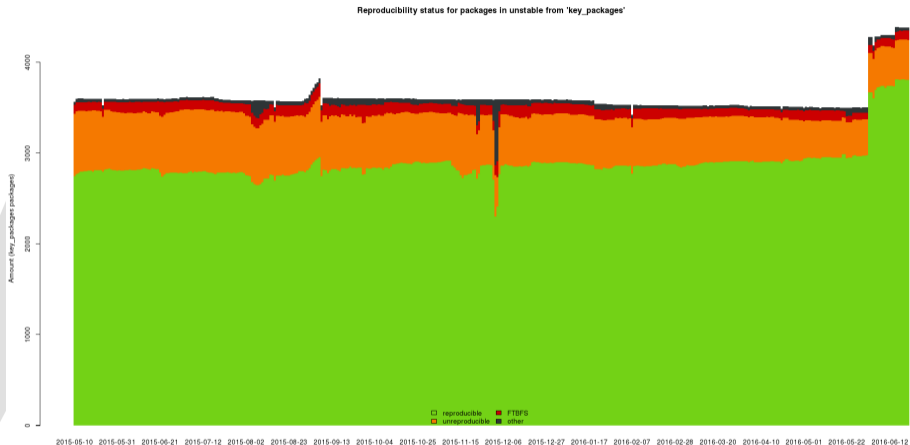
Debian package sets on tests.r-b.org



42 different "package sets", eg. required is only 74.3% reproducible



Debian package sets on tests.r-b.org

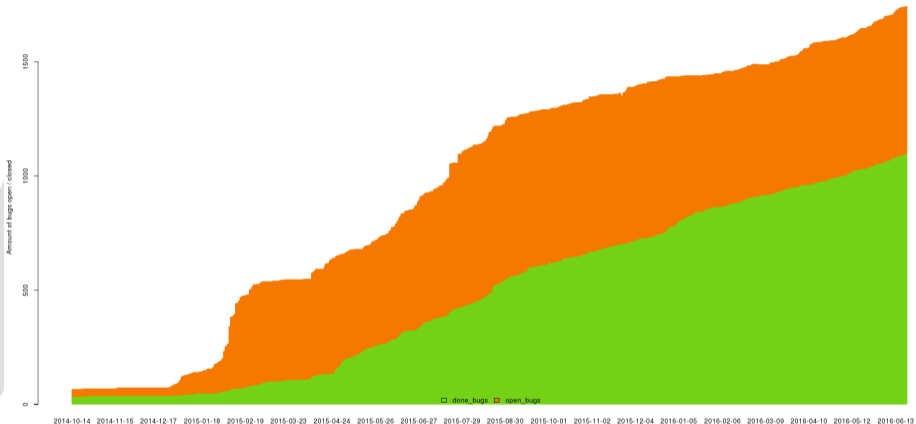


Debian's key_packages are 86.9% reproducible, but 437 packages (10%) will still need to be fixed



Progress in the Debian bug tracker

Open and closed bugs (with all usertags except tagged 'ttbfs')



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



What we did in Debian

- Agreed on using a fixed build path: `/build/`
- Recording the build environment: `.buildinfo`
- `strip-nondeterminism`
- `diffoscope` (formerly `debbindiff`)
- `SOURCE_DATE_EPOCH`
- `disorderfs`
- 1700+ patches: `dpkg`, `debhelper`, `sbuild`, ...
- 2 packages modified to achieve those 89% (90.2%)
- ...



Detour: Reproducible builds demand a defined build environment

- ...and being able to re-create this build environment is mandatory too.
- Without an *sufficiently identical* build environment, reproducible builds will only happen by sheer luck.



Detour: Reproducible builds demand a defined build environment

- ...and being able to re-create this build environment is mandatory too.
- Without an *sufficiently identical* build environment, reproducible builds will only happen by sheer luck.
- I've only verified this works for Debian so far... `koji` is designed for that too, Guix as well...
- I'd very much like to hear about your experiences.



Debian .buildinfo files

- Aggregates in the same file:
 - ▶ Sources (checksums)
 - ▶ Generated binaries (checksums)
 - ▶ Packages used to build (with specific version, checksums coming soon)
- Can be later used to exactly recreate environment
- For Debian, all versions are available from snapshot.debian.org



Example .buildinfo file

```
Format: 1.9
Build-Architecture: amd64
Source: txtorcon
Binary: python-txtorcon
Architecture: all
Version: 0.11.0-1
Build-Path: /build/txtorcon-0.11.0-1
Checksums-Sha256:
  a26549d9...7b 125910 python-txtorcon_0.11.0-1_all.deb
  28f6bcbe...69 2039 txtorcon_0.11.0-1.dsc
Build-Environment:
  base-files (= 8),
  base-passwd (= 3.5.37),
  bash (= 4.3-11+b1),
  ...
```



.buildinfo files elsewhere

- neither used nor specified elsewhere **yet**
- it's clear we need something like them
- it's clear what needs to be specified
- it "just" needs to be done...



.buildinfo files elsewhere

- neither used nor specified elsewhere **yet**
- it's clear we need something like them
- it's clear what needs to be specified
- it "just" needs to be done...
- and it **needs** to be done: we need "API"s to define inputs and outputs, these "API"s will be different in their implementation but the basic principles will be the same. Without .buildinfo files reproducible rebuilds are not doable in practice...!



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



Tell the world & collaborate

- Weekly reports since May 2015
- First Reproducible World Summit in December 2015 (Athens, Greece)
 - ▶ 40 people from 16 projects
 - ▶ another summit in second half 2016, somewhere in Europe



Tell the world & collaborate

- Weekly reports since May 2015
- First Reproducible World Summit in December 2015 (Athens, Greece)
 - ▶ 40 people from 16 projects
 - ▶ another summit in second half 2016, somewhere in Europe
- 2 GSoC students in 2015, totally new contributors, totally rocking
- 4 GSoC and Outreachy students in 2016, also rocking already!



debian-policy

- Section 4.15: “Sources **must** build reproducible binaries.”



debian-policy

- Section 4.15: “Sources **must** build reproducible binaries.”
- We hope this will happen after stretch (Debian 9) release



debian-policy

- Section 4.15: “Sources **must** build reproducible binaries.”
- We hope this will happen after stretch (Debian 9) release
- In 2016: “Sources **shall** build reproducible binaries.” ?



Summary

- This is just a proof-of-concept, Debian is not 90% reproducible, Debian is 0% reproducible.
- Patches still need to be merged (until the end of the year)



Summary

- This is just a proof-of-concept, Debian is not 90% reproducible, Debian is 0% reproducible.
- Patches still need to be merged (until the end of the year)
- I hope that Debian 9, "stretch", will be *partially reproducible in a meaningful way*



Summary

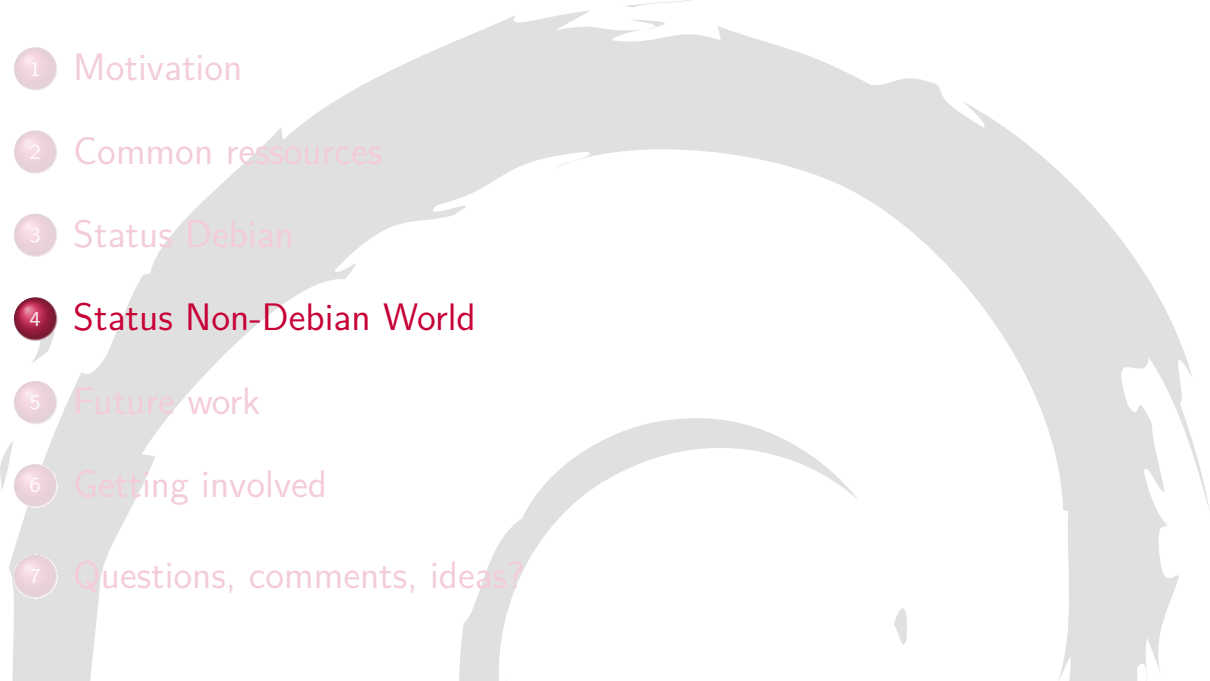
- This is just a proof-of-concept, Debian is not 90% reproducible, Debian is 0% reproducible.
- Patches still need to be merged (until the end of the year)
- I hope that Debian 9, "stretch", will be *partially reproducible in a meaningful way*
- Debian unstable still needs changes to dpkg and ftp.debian.org (for keeping .buildinfo files)



Summary

- This is just a proof-of-concept, Debian is not 90% reproducible, Debian is 0% reproducible.
- Patches still need to be merged (until the end of the year)
- I hope that Debian 9, "stretch", will be *partially reproducible in a meaningful way*
- Debian unstable still needs changes to dpkg and ftp.debian.org (for keeping .buildinfo files)
- What's beyond (rebuilding, .buildinfo file signing and distribution, user tools) mostly still needs *design and code*



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

Status coreboot

- <https://tests.r-b.org/coreboot>
- 100% reproducible with seabios payload
- tests maintained by Alexander 'lynxis' Couzens
- unclear what the next steps are... they don't release binaries...
- needs involvement from coreboot developers



Status OpenWrt

- <https://tests.r-b.org/openwrt>
- selected images are 100% reproducible and selected packages 99.7%
- using 13 patches send upstream on January 25th 2016
- tests maintained by Alexander 'lynxis' Couzens and Bryan Newbold
- recreating the build env: needs to be checked in practice
- user verification tools: not yet



Status NetBSD

- <https://tests.r-b.org/netbsd>
- 42 (77.7%) out of 54 built NetBSD files are reproducible
- tests maintained by Thomas 'wiz' Klausner and h01ger
- MKREPRO=yes
- MK_TIMESTAMP=\$SOURCE_DATE_EPOCH
- recreating the build env: ?



Status FreeBSD

- <https://tests.r-b.org/freebsd>
- base system not yet reproducible, but almost there
- 63% of 15k ports were reproducible in 2013 already, their wiki says
- tests maintained by h01ger so far... but Ed Maste has recently started work
- recreating the build env: ?
- soon testing ports (=packages) too



Status Fedora

- <https://tests.r-b.org/fedora> (23)
- maintained by Dhiru Kholia and h01ger
- rpm repo available by Dhiru, but still **0% reproducible**
- first patch for rpm merged
- rpm format includes build time and build host and signatures...
- recreating the build env: koji
- next: first reproducible rpm, use koji



Status Fedora

- <https://tests.r-b.org/fedora> (23)
- maintained by Dhiru Kholia and h01ger
- rpm repo available by Dhiru, but still **0% reproducible**
- first patch for rpm merged
- rpm format includes build time and build host and signatures...
- recreating the build env: koji
- next: first reproducible rpm, use koji
- help/patches from SuSE? :)



Status Arch Linux

- <https://tests.r-b.org/archlinux>
- maintained by Levente 'anthraxx' Polyak and h01ger
- reproducible patches available for pacman by anthraxx
- recreating the build env: unaddressed

Status F-Droid

- not yet: `https://tests.r-b.org/f-droid`
- maintained by Hans-Christoph Steiner and h01ger
- work has just begun...
- we need help with vagrant. please contact me if you can help...

Status openSUSE

- Watch Bernhard's talk directly after this one!



More projects with known activities

- Bitcoin, Tor,
- Signal
- OpenSUSE (could be tested easily...)
- Ubuntu
- Guix, NixOS
- ElectroBSD
- Qubes, TAILS, Subgraph OS
- commercial, proprietary Software
- ?

Detour: what, reproducible commercial Software???

- Guess which

Detour: what, reproducible commercial Software???

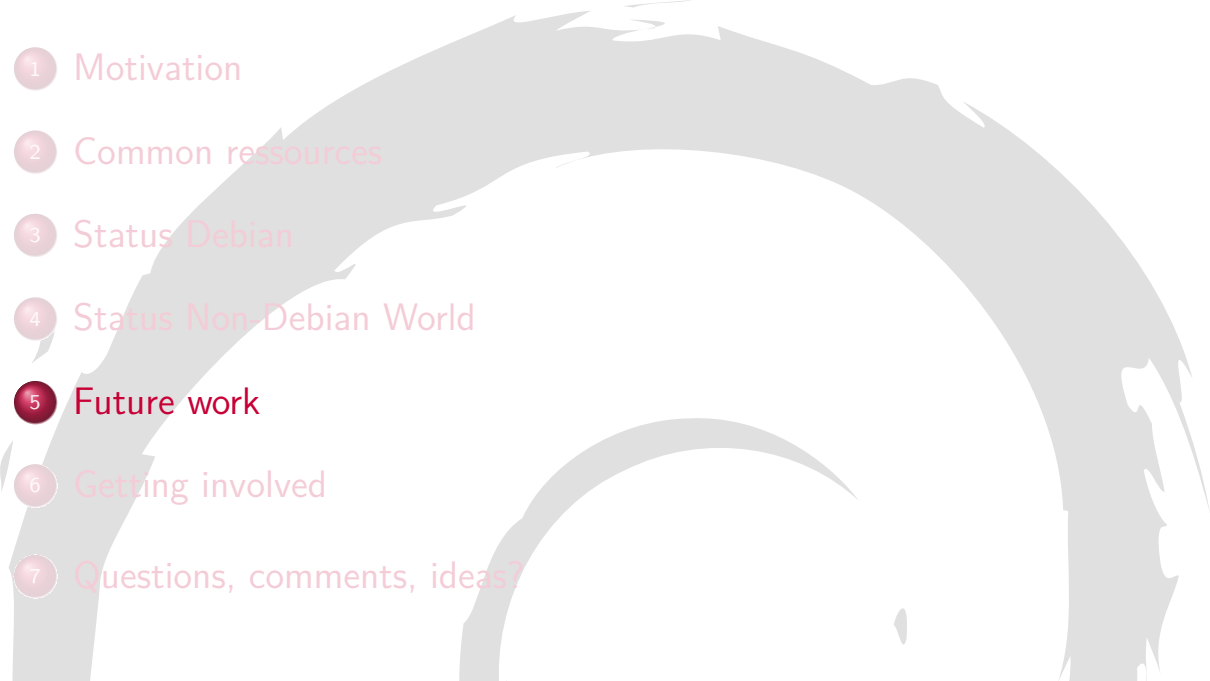
- Guess which
- Microsoft Windows? (the source is available)
- medical devices in your body?
- arms?
- critical infrastructure like in nuclear powerplants?
- cars?

Detour: what, reproducible commercial Software???

- Guess which
- Microsoft Windows? (the source is available)
- medical devices in your body?
- arms?
- critical infrastructure like in nuclear powerplants?
- cars?
- Gambling machines!

Unknown activities?

- OpenBSD
- Gentoo (stage1)
- ?

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

Distributing .buildinfo files

- Probably 100,000 new files per Debian suite; 50% increase per suite
- Mirrors would not be happy, so should not go there



Distributing .buildinfo files

- Probably 100,000 new files per Debian suite; 50% increase per suite
- Mirrors would not be happy, so should not go there
- We'll need more files with detached signatures...
- Revoking signatures?
- ...



Rebuilders and sharing signed checksums

- Almost no work has been done here yet.



Rebuilders and sharing signed checksums

- Almost no work has been done here yet.
- 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...



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?"



Summary

- We've come a long way.
- We've made impressive progress.
- We're still not nearly where we want to be.



Summary

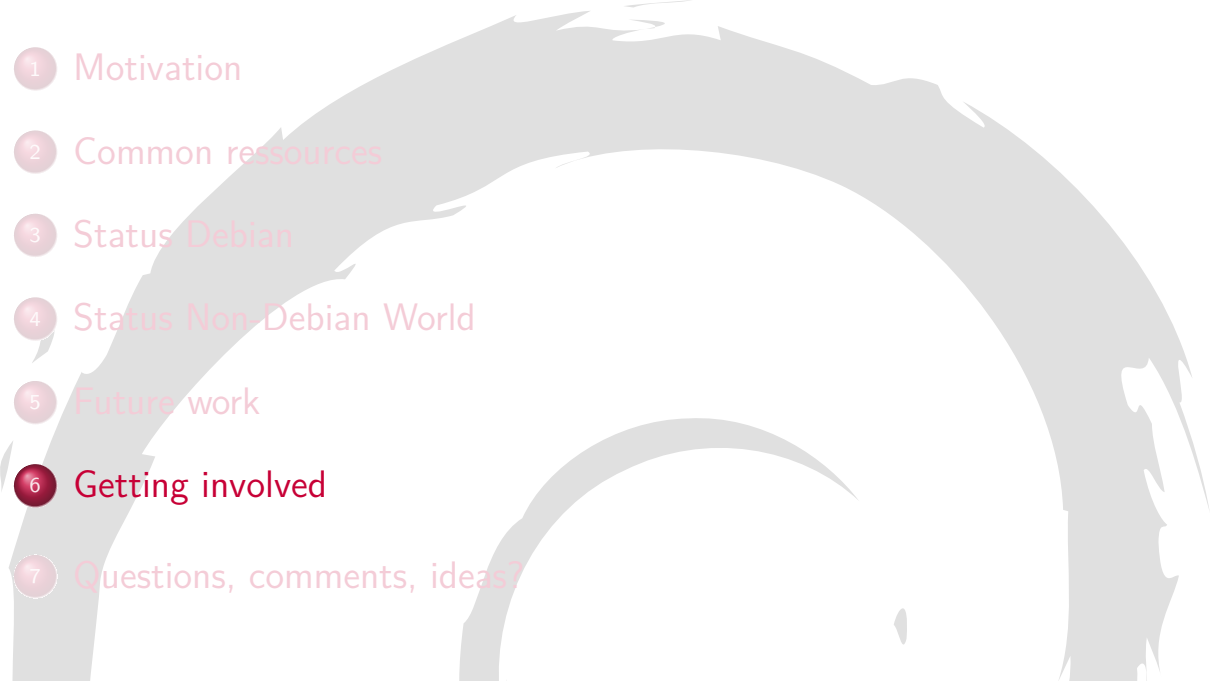
- We've come a long way.
- We've made impressive progress.
- We're still not nearly where we want to be.
- In fact, it's still fully not clear where we need to be going.
- We've shown it's technically feasible, now we need to create policies and processes!



Summary

- We've come a long way.
- We've made impressive progress.
- We're still not nearly where we want to be.
- In fact, it's still fully not clear where we need to be going.
- We've shown it's technically feasible, now we need to create policies and processes!
- Keep up the great work!
- Join the fun! There are many big and small things to do!



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

As a software developer

- Merge our patches



As a software developer

- Merge our patches
- 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 IRC or on our mailing lists



Join the Reproducible builds team(s)!

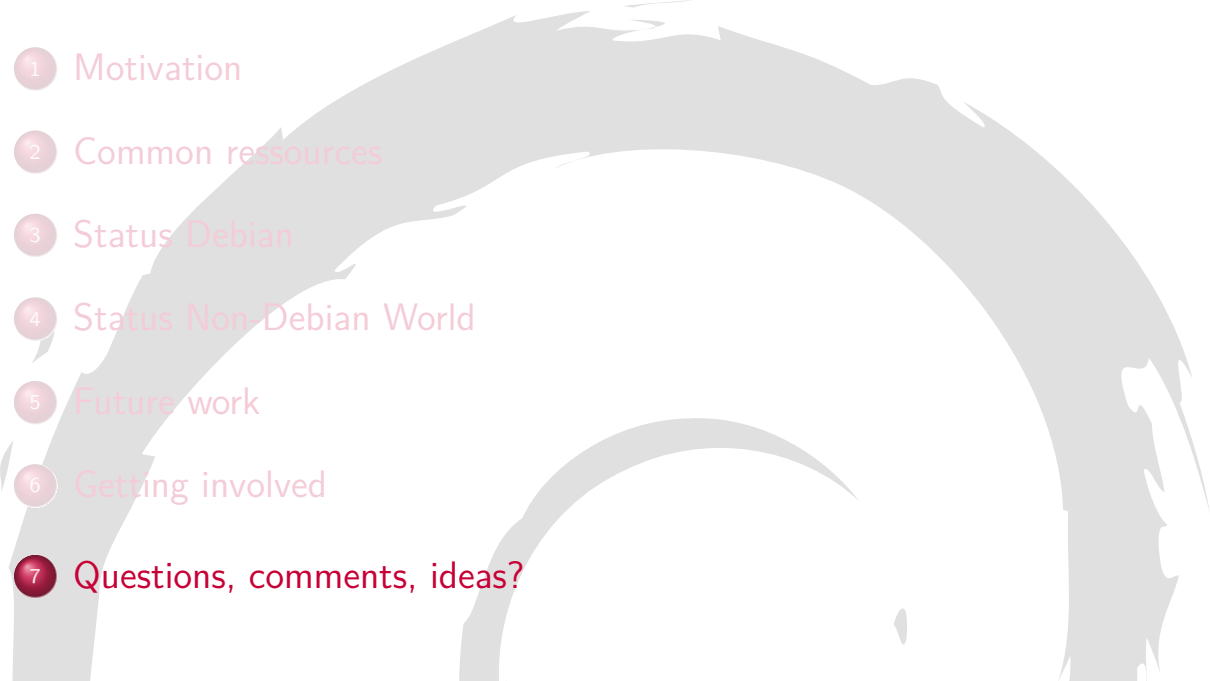
- Why?

- ▶ ♡♡♡ Lovely group of people ♡♡♡
- ▶ Learn something new everyday
- ▶ Change the (software) world!

- What do we do?

- ▶ Review packages
- ▶ Identify issues and document solutions
- ▶ `tests.r-b.o`, `diffoscope`, `strip-nondeterminism`
- ▶ Propose changes for toolchain
- ▶ Submit patches for individual packages
- ▶ Write more general documentation and talk to the world



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

Questions, comments, ideas?



Questions, comments, ideas?

- `https://reproducible-builds.org/docs`
- `https://tests.reproducible-builds.org`
- `#reproducible-builds` on `irc.0FTC.net`
- and/or `#debian-reproducible` too!



Questions, comments, ideas?

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



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](https://creativecommons.org/licenses/by-sa/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>.

