Reproducible builds ecosystem

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

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

ripe72 (Copenhagen, DK)
2016-05-24
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](http://video.debian.net)
- Debian-Edu (Debian for education)
- Debian QA (quality assurance)
  - [https://piuparts.debian.org](https://piuparts.debian.org)
  - [https://jenkins.debian.net](https://jenkins.debian.net) (1100 jobs continuously testing Debian)
- Debian-LTS (Long Term Support)
- `sudo (apt|dnf) install torbrowser-launcher`
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...
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 Levens
Jelmer Vernooij
josch
Juan Picca
Lunar
Mathieu Bridon
Mattia Rizzolo
Nicolas Bouleuguez
Niels Thykier
Niko Tyni
Paul Wise
Peter De Wachter
Philip Rinn
Reiner Herrmann
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 Levens
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
Sascha Steinbiss
Satyam Zode
Scarlett Clark
Stefano Rivera
Stéphane Glondu
Steven Chamberlain
Tom Fitzhenry
Valerie Young
Valentin Lorentz
Wookey
Ximin Luo

Holger 'h01ger' Levens (Debian)
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 Levens
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...
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...
- 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...
- 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: [removed], 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/
The solution

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

We call this:

“Reproducible builds”
Demo
Demo - unreproducible builds

- db4c5c4d6eaec2268eeab750920e34004292ec3a giftrans_1.12.2-19.dsc
- a931a19e832024f509f7902b2b5560f8e46f004b giftrans_1.12.2-19.debian.tar.xz
- 816067762fe7d41f2b73f0acd2da0e51a1b93f27 giftrans_1.12.2.orig.tar.gz
- 97f656caf73a5c73bd86e7e0d7f134c55ad83fc8 .1/giftrans_1.12.2-19_amd64.deb
- 08211b176c889e8270aa87a1a753b3bc24c6aed0 .2/giftrans_1.12.2-19_amd64.deb
- 1365e56a2217fa44afe3594333b9aa70fc0dd8d4 .3/giftrans_1.12.2-19_amd64.deb
- b486c9337968b04b7159c2500c03434cbac6f50b .4/giftrans_1.12.2-19_amd64.deb
- 50aef605a36eec64c307812a32553d38f30d1672 .5/giftrans_1.12.2-19_amd64.deb
Demo - reproducible builds

- db4c5c4d6eaec2268eeab750920e34004292ec3a
giftrans_1.12.2-19.dsc
- a931a19e832024f509f7902b2b5560f8e46f004b
giftrans_1.12.2-19.debian.tar.xz
- 816067762fe7d41f2b73f0acd2da0e51a1b93f27
giftrans_1.12.2.orig.tar.gz
- 2a7c368a7fb1857b964a53fd53fd39d466e81d3a
  .1/giftrans_1.12.2-19_amd64.deb
- 2a7c368a7fb1857b964a53fd53fd39d466e81d3a
  .2/giftrans_1.12.2-19_amd64.deb
- 2a7c368a7fb1857b964a53fd53fd39d466e81d3a
  .3/giftrans_1.12.2-19_amd64.deb
- 2a7c368a7fb1857b964a53fd53fd39d466e81d3a
  .4/giftrans_1.12.2-19_amd64.deb
- 2a7c368a7fb1857b964a53fd53fd39d466e81d3a
  .5/giftrans_1.12.2-19_amd64.deb
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!
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.

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
Continuously testing Debian testing, unstable and experimental on amd64 and i386 and armhf
Also testing: coreboot, OpenWrt, NetBSD, FreeBSD, Arch Linux, Fedora and soon F-Droid and Guix too
283 jenkins jobs running on 28 hosts
41 scripts with a total of 4k lines of Python and 6k lines of Bash Shell
31 contributors for jenkins.debian.net.git
Holger ‘h01ger’ Levsen (Debian)
Reproducible builds ecosystem
tests.reproducible-builds.org

- Continuously testing Debian testing, unstable and experimental
Continuously testing Debian testing, unstable and experimental
on amd64 and i386 and armhf
Continuous testing Debian testing, unstable and experimental
on amd64 and i386 and armhf
Also testing: coreboot, OpenWrt, NetBSD, FreeBSD, Arch Linux, Fedora and soon F-Droid and Guix too
Continuously testing Debian testing, unstable and experimental
- on amd64 and i386 and armhf
- Also testing: coreboot, OpenWrt, NetBSD, FreeBSD, Arch Linux, Fedora and soon F-Droid and Guix too
- 283 jenkins jobs running on 28 hosts
- 41 scripts with a total of 4k lines of Python and 6k lines of Bash Shell
- 31 contributors for jenkins.debian.net.git
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: 18 nodes with 70 cores and 35 GB RAM
  sponsored by Debian
- arm64: coming soon
## Variations (when testing Debian)

<table>
<thead>
<tr>
<th>variation</th>
<th>first build</th>
<th>second build</th>
</tr>
</thead>
<tbody>
<tr>
<td>hostname</td>
<td>jenkins</td>
<td>i-capture-the-hostname</td>
</tr>
<tr>
<td>domainname</td>
<td>debian.net</td>
<td>i-capture-the-domainname</td>
</tr>
<tr>
<td>env TZ</td>
<td>GMT+12</td>
<td>GMT-14</td>
</tr>
<tr>
<td>env LANG</td>
<td>C</td>
<td>fr_CH.UTF-8</td>
</tr>
<tr>
<td>env LC_ALL</td>
<td>not set</td>
<td>fr_CH.UTF-8</td>
</tr>
<tr>
<td>env USER</td>
<td>pbuilder1</td>
<td>pbuilder2</td>
</tr>
<tr>
<td>uid</td>
<td>1111</td>
<td>2222</td>
</tr>
<tr>
<td>gid</td>
<td>1111</td>
<td>2222</td>
</tr>
<tr>
<td>shell</td>
<td>dash</td>
<td>bash</td>
</tr>
<tr>
<td>UTS namespace</td>
<td>shared with the host</td>
<td>modified using /usr/bin/unshare --uts</td>
</tr>
<tr>
<td>kernel version</td>
<td>Linux 3.16 or 4.X</td>
<td>on amd64 and i386 always varied, on armhf sometimes</td>
</tr>
<tr>
<td>32 vs 64 bit kernel</td>
<td>one or the other</td>
<td>only varied on i386</td>
</tr>
<tr>
<td>umask</td>
<td>0022</td>
<td>0002</td>
</tr>
<tr>
<td>CPU type</td>
<td>Intel and AMD variation for i386 and amd64 (work in progress)</td>
<td>on armhf varied a bit</td>
</tr>
<tr>
<td>filesystem</td>
<td>same for both builds on amd64: (tmpfs), on i386 and armhf ext3/4</td>
<td>(and we have disorderfs, but the code is disabled)</td>
</tr>
<tr>
<td>year, month, date</td>
<td>on amd64 and i386: 398 days variation, on armhf not yet</td>
<td>hour is usually the same... usually, the minute differs...</td>
</tr>
<tr>
<td>hour, minute</td>
<td>hour is usually the same... usually, the minute differs...</td>
<td>is likely the same...</td>
</tr>
</tbody>
</table>
Common problems

- time stamps
Common problems

- time stamps
- time zones
- locales
Common problems

- time stamps
- time zones
- locales
- everything else (separated 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

Example

```sh
$ gcc -f1to -c utils.c
$ nm -a utils.o | grep inline
0000000000000000 n .gnu.lto_.inline.381a277a0b6d2a35
```
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), Fedora, Arch Linux, FreeBSD, NetBSD, Guix, Homebrew..
- Maintainers (upstream and in other distros) wanted.
- [https://diffoscope.org/](https://diffoscope.org/)
diffoscope example (HTML output)
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.
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...)

Holger ‘h01ger’ Levsen (Debian)
SOURCE_DATE_EPOCH (closed bugs)

- dh-strip-nondeterminism
- gcc (\_\_DATE\_\_ and \_\_TIME\_\_ macros)
- #791823: debhelper
- #787444: help2man
- #790899: epydoc
- #794004: ghostscript
- #796130: man2html
- #783475: texi2html
- #794586: ocamlcmdoc
- #795942: wheel
- #792202: texlive-bin
- ...

Holger 'h01ger' Levsen (Debian)
Reproducible builds ecosystem
• 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?
21,365 (88.5%) out of 24,135 source packages are reproducible in our test framework. 90.1% in testing/amd64.
Notes and issues on tests.reproducible-builds.org

- 194 categorised distinct issues
- 3,085 notes
Notes and issues on tests.reproducible-builds.org

- 194 categorised distinct issues
- 3,085 notes
- 1879 unreproducible packages in sid/amd64, but only 227 without a note
- 516 packages failing to build, but only 92 without a note
Notes and issues on tests.reproducible-builds.org

- 194 categorised distinct issues
- 3,085 notes
- 1879 unreproducible packages in sid/amd64, but only 227 without a note
- 516 packages failing to build, but only 92 without a note
- maintained in notes.git
Notes and issues on tests.reproducible-builds.org

- 194 categorised distinct issues
- 3,085 notes
- 1879 unreproducible packages in sid/amd64, but only 227 without a note
- 516 packages failing to build, but only 92 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
35 different "package sets", eg. required is only 73.1% reproducible
Debian package sets on tests.r-b.org

Debian’s key_packages are 84.5% reproducible, but 435 packages (12.8%) will still need to be fixed.
Progress in the Debian bug tracker

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
- 1600+ patches: dpkg, debhelper, sbuild, ...
- 4 packages modified to achieve those 88% (90.1%)
- ...
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 for Debian so far... *koji* is designed for that too, *Guix* as well...
- I’d very much like to be corrected here, with tests.
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
.buildinfo files elsewhere

- neither used nor specified elsewhere
- 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
- 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 rebuild are not doable in practice…
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
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

Holger 'h01ger' Levsen (Debian)
Reproducible builds ecosystem
ripe72 42 / 69
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
Section 4.15: “Sources must build reproducible binaries.”
Section 4.15: “Sources **must** build reproducible binaries.”

We hope this will happen after stretch (Debian 9) release
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*
Status coreboot

- https://tests.r-b.org/coreboot
- 99.2% 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/coreboot
- selected images are 100% reproducible and selected packages 99.7%
- using 13 patches send upstream on January 25th
- tests maintained by Alexander ’lynxis’ Couzens and Bryan Newbold
- recreating the build env: needs to checked in practice
- user verification tools: not yet
Status NetBSD

- https://tests.r-b.org/netbsd
- 21 (38.8%) 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...

- Recreating the build env: ?
Status Fedora

- [https://tests.r-b.org/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? :)

Holger 'h01ger' Levsen (Debian)
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...
Unmentioned, 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

• Microsoft Windows
• Medical devices in your body
• Critical infrastructure like in nuclear power plants
• Cars
• Gambling machines!
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!
Unmentioned, unknown activities?

- OpenBSD
- Gentoo (stage1)
- ?
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.

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!

Holger ‘h01ger’ Levsen (Debian)  Reproducible builds ecosystem  ripe72  61 / 69
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!
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!
As a software developer

- Merge our patches

See https://reproducible-builds.org/specs/
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
Questions, comments, ideas?
Questions, comments, ideas?

- https://reproducible-builds.org/docs
- https://tests.reproducible-builds.org
- #reproducible-builds on irc.OFTC.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
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