Reproducible builds everywhere
eg. in Debian, OpenWrt and LEDE

Bit by bit identical binaries
from a given source

Alexander 'lynxis' Couzens
Holger 'h01ger' Levsen

OpenWrt Summit in Berlin, Germany
2016-10-13

about h01ger

- B8BF 5413 7B09 D35C F026 FE9D 091A B856 069A AA1C
- Debian user since 1995
- Debian contributor since 2001
- OpenWrt user since 2006
- 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 (1200 jobs continously testing Debian)
- Debian Reproducible builds team member
  - since April 2015 funded by the Linux Foundation

about lynxis

- 390D CF78 8BF9 AA50 4F8F F1E2 C29E 9DA6 A0DF 8604
- Debian user since 2003
- OpenWrt user since 2006
- LEDE founding member
- coreboot hacker
- tests.reproducible-builds.org contributor
- CCC member

about OpenWrt and LEDE

- In this talk we'll ignore the distinction between the two:
- when we say "OpenWrt" me mean "LEDE and OpenWrt",
- when we say "LEDE" me mean "OpenWrt and LEDE",
- when we say "OpenWrt and LEDE" we mean "LEDE and OpenWrt".

OpenWrt
Wireless Freedom

about OpenWrt and LEDE

- In this talk we'll ignore the distinction between the two:
- when we say "OpenWrt" me mean "LEDE and OpenWrt",
- when we say "LEDE" me mean "OpenWrt and LEDE",
- when we say "OpenWrt and LEDE" we mean "LEDE and OpenWrt".
- They are two projects though and when there are differences we'll mention them.

Who are you?

Who are you?

- Seen a talk about reproducible builds?

Who are you?

- Seen a talk about reproducible builds?
- Contributed to the efford?

Who are you?

- Seen a talk about reproducible builds?
- Contributed to the efford?
- Uses Debian or a Debian based system?

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
Emmanuel Bourg
Emanuel Bronshtein
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
Emmanuel Bourg
Emanuel Bronshtein
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

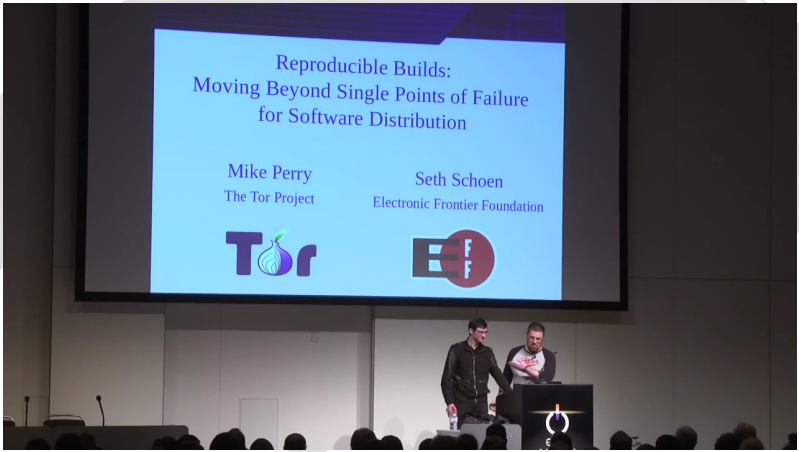
jenkins.debian.net.git contributors

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

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
- How can you be sure what's running on your machine or on a build daemon network connected to the net? Do you ever leave your computers physically 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
- How can you be sure what's running on your machine or on a build daemon network connected to the net? Do you ever leave your computers physically alone?
- Huge financial incentives to crack developer machines or a project's 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
- How can you be sure what's running on your machine or on a build daemon network connected to the net? Do you ever leave your computers physically alone?
- Huge financial incentives to crack developer machines or a project's build infrastructure...

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/`

The solution

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

The solution

We call this:

“Reproducible builds”

Debian demo (skipped)

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

Debian demo (skipped)

- Build a package 5 times, get 5 .debs with different checksums
- Build a package 5 times, get 5 .debs with the same checksum
  Yes, it's really this simple.

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...

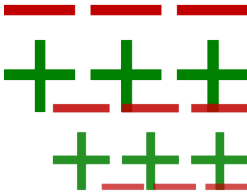
- smaller deltas, thus faster updates possible
- in Debian: lots of QA benefits
- Google does reproducible builds, to save money
- ...

reproducible-builds.org

- `https://reproducible-builds.org`
- git repositories, IRC channels, mailinglists, webspace

Debugging problems: https://try.diffoscope.org

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



diffoscope example (HTML output)

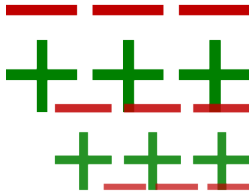
51431	INSERT INTO "targets" VALUES('ttu.ee',13611);	51438	INSERT INTO "targets" VALUES('ttu.ee',13542);
51432	INSERT INTO "targets" VALUES('ttu.ee',13611);	51439	INSERT INTO "targets" VALUES('ttu.ee',13542);
51433	[9300 lines removed]	51440	[9314 lines removed]
60733	CREATE TABLE git_commit	60754	CREATE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

install.rdf

Offset 5, 15 lines modified		Offset 5, 15 lines modified	
5	<Description about="urn:mozilla:install-manifest">	5	<Description about="urn:mozilla:install-manifest">
6	<em:name>HTTPS-Everywhere</em:name>	6	<em:name>HTTPS-Everywhere</em:name>
7	<em:creator>Mike Perry, Peter Eckersley, & Yan Zhu</em:creator>	7	<em:creator>Mike Perry, Peter Eckersley, & Yan Zhu</em:creator>
8	<em:aboutURL>chrome://https-everywhere/content/about.xul</em:aboutURL>	8	<em:aboutURL>chrome://https-everywhere/content/about.xul</em:aboutURL>
9	<em:id>https-everywhere@eff.org</em:id>	9	<em:id>https-everywhere@eff.org</em:id>
10	<em:type>2</em:type> <!-- type: Extension -->	10	<em:type>2</em:type> <!-- type: Extension -->
11	<em:description>Encrypt the Web! Automatically use HTTPS security on many sites.</em:description>	11	<em:description>Encrypt the Web! Automatically use HTTPS security on many sites.</em:description>
12	<em:version>5.0.6</em:version>	12	<em:version>5.0.7</em:version>

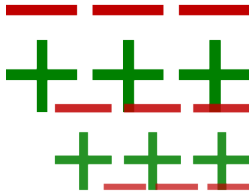
`diffoscope` is "just" for debugging

- 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`!



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`!
- `https://try.diffoscope.org`



tests.reproducible-builds.org

- Continuously testing Debian `testing`, `unstable` and `experimental`
- Also testing: coreboot, OpenWrt, LEDE, NetBSD, FreeBSD, Arch Linux, Fedora and soon F-Droid too
- 8-12 `amd64` nodes, 150 cores and soon 500 GB RAM - thanks to Profitbricks.com!
- 22 `armhf` nodes, 98 cores and 53 GB RAM
- 329 jenkins jobs running on jenkins.debian.net
- 43 scripts in Python and Bash, 283 lines of code in average
- 37 contributors for `jenkins.debian.net.git`

Variations (when testing Debian)

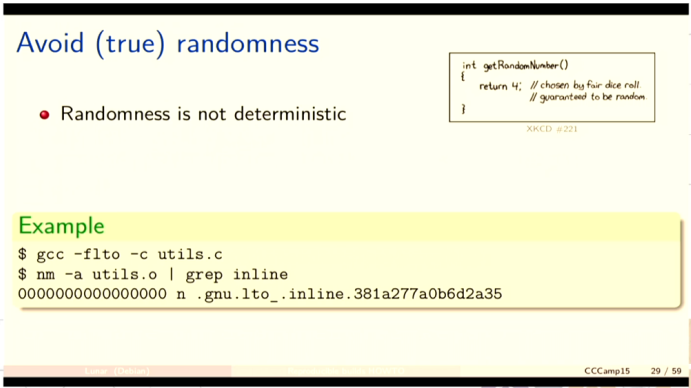
variation	first build	second build
hostname	`jenkins`	`i-capture-the-hostname`
domainname	`debian.net`	`i-capture-the-domainname`
`env TZ`	`GMT+12`	`GMT-14`
`env LANG`	`C`	`fr_CH.UTF-8`
`env LC_ALL`	not set	`fr_CH.UTF-8`
`env USER`	`pbuilder1`	`pbuilder2`
uid	1111	2222
gid	1111	2222
UTS namespace	shared with the host	*modified using `/usr/bin/unshare --uts`*
kernel version	Linux 3.16 or 4.X	on amd64 always varied, on armhf sometimes
umask	0022	0002
CPU type	varied on i386	
	on armhf varied a bit, not on amd64	
filesystem	same for both builds on amd64: (`tmpfs`), on armhf `ext3/4`	
		(and we have `disorderfs`, but the code is disabled)
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	*is likely the same...*	

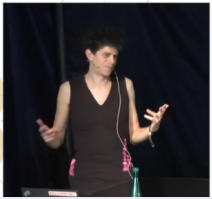
Common problems

- time stamps
- timezones
- locales
- build paths
- 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`





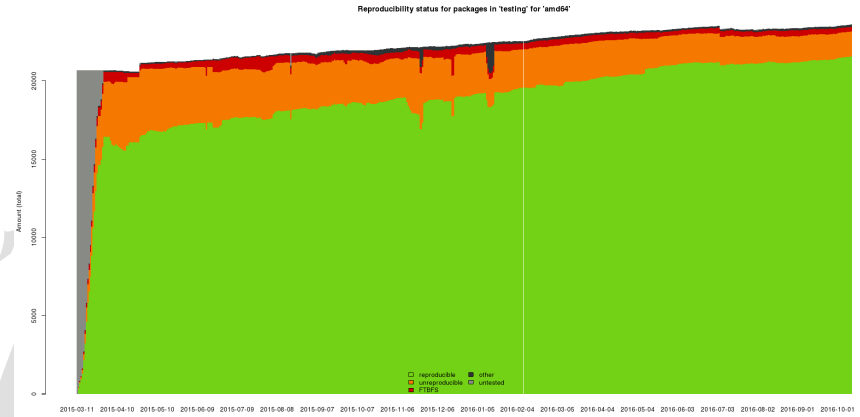
SOURCE_DATE_EPOCH

- 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
- can be set to the latest git commit too or the latest file modification date

SOURCE_DATE_EPOCH

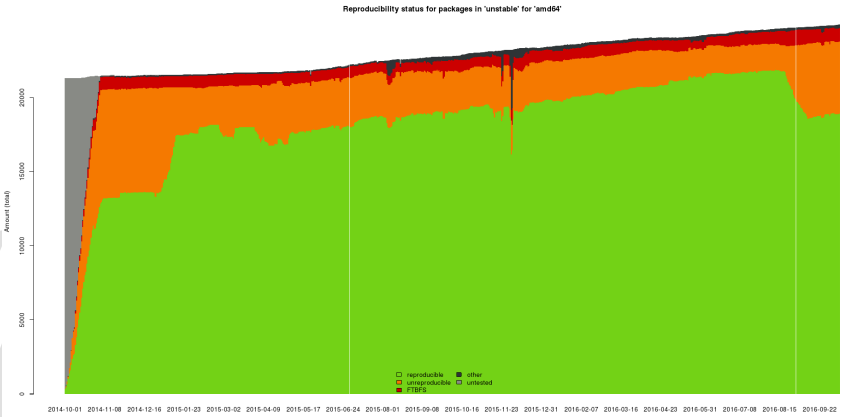
- SOURCE_DATE_EPOCH spec available:
- https://reproducible-builds.org/specs/
- many upstreams support it already
- has been adopted by other distributions (OpenWrt, LEDE, NetBSD, FreeBSD, Arch Linux, coreboot, Guix, ...) and many many upstreams (GCC, dpkg, rpm, mkisofs, ghostscript, libxslt, sphinx, texlive-bin, ...)

Progress in Debian `testing` ("stretch")



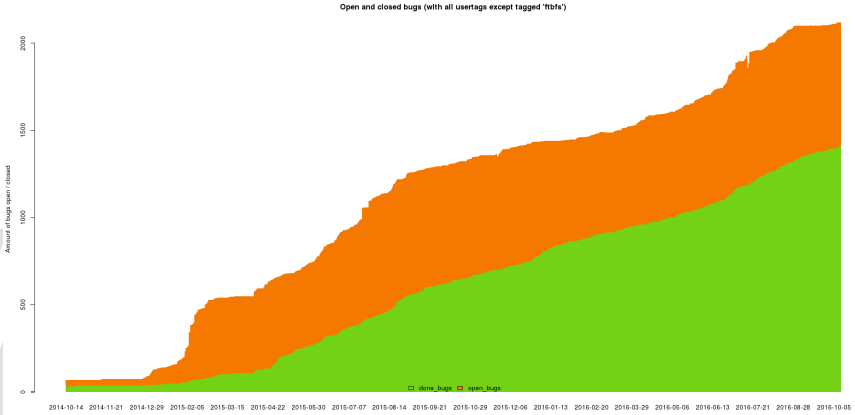
21,527 (91.2%) out of 23,597 source packages are reproducible in our test framework on `amd64`

Progress in Debian unstable



18,898 (75.8%) out of 24,931 source packages are reproducible
in our test framework on amd64 (difference due to build path variations)

Progress in the Debian bug tracker



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

Details on tests.reproducible-builds.org

- `https://reproducible.debian.net/$src`
- 43 package sets
- 250 categorised distinct issues
- 6,944 notes
- 1,894 unreproducible packages in `stretch` (testing), but only 177 without a note (5,777 in `unstable` but also only 277 without a note)
- maintained in `notes.git` by 47 contributors
- currently Debian only, but cross distro notes are planned

Summary / What's left to do

- This is a proof-of-concept, Debian is neither 91.2% reproducible nor 75.8%. (and 10% $>$ 2,300 sources packages!)

Summary / What's left to do

- This is a proof-of-concept, Debian is neither 91.2% reproducible nor 75.8%. (and 10% > 2,300 sources packages!)
- All our required changes are finally in Debian now, except `dpkg` and `.buildinfo` file support on the archive side.

Summary / What's left to do

- This is a proof-of-concept, Debian is neither 91.2% reproducible nor 75.8%. (and 10% > 2,300 sources packages!)
- All our required changes are finally in Debian now, except `dpkg` and `.buildinfo` file support on the archive side.
- We hope that Debian 9, "stretch", will be partially reproducible in a meaningful way, in 2017.
- What's beyond (rebuilding, `.buildinfo` file handling, user tools) still needs *design and code.*

Summary / What's left to do

- This is a proof-of-concept, Debian is neither 91.2% reproducible nor 75.8%. (and 10% > 2,300 sources packages!)
- All our required changes are finally in Debian now, except `dpkg` and `.buildinfo` file support on the archive side.
- We hope that Debian 9, "stretch", will be partially reproducible in a meaningful way, in 2017.
- What's beyond (rebuilding, `.buildinfo` file handling, user tools) still needs *design and code.*
- Will Debian 10, "buster", be 100% reproducible?

Tell the world & collaborate

- "We don't care about Debian (only), we care about free and open source software."

Tell the world & collaborate

- "We don't care about Debian (only), we care about free and open source software."
- Weekly reports since May 2015

Tell the world & collaborate

- "We don't care about Debian (only), we care about free and open source software."
- 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

- "We don't care about Debian (only), we care about free and open source software."
- 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`
- Second Reproducible World Summit in December 2016 in Berlin
  - Talk to h01ger if you want to attend.

Skipping some...

- https://tests.r-b.org/coreboot
- https://tests.r-b.org/netbsd
- https://tests.r-b.org/freebsd
- paused: https://tests.r-b.org/archlinux
- paused: https://tests.r-b.org/fedora
- not yet: https://tests.r-b.org/f-droid




Skipping some more...

- Bitcoin (2011)
- Tor (2013)
- NixOS, Guix, ElectroBSD
- Qubes, Tails
- very few commercial, propietary software (guess where!)
- ?

Skipping some more...

- Bitcoin (2011)
- Tor (2013)
- NixOS, Guix, ElectroBSD
- Qubes, Tails
- very few commercial, propietary software (gamblingmachines!)
- ?

OpenWrt and LEDE tested for reproducible builds

OpenWrt
Wireless Freedom

OpenWrt and LEDE tested for reproducible builds

- `https://tests.r-b.org/openwrt`
- `https://tests.r-b.org/lede`
- reproducible_(openwrt_common|openwrt|lede).sh scripts in `jenkins.debian.net.git`
- 1,073/1,089 packages and 12/1 (OpenWrt/LEDE) images tested each week
- variations: TZ, LANG, LC_ALL, PATH, (umask), make -j, linux64 –uname-2.6, CAPTURE_ENVIRONMENT

OpenWrt Wireless Freedom

Thanks to these OpenWrt / LEDE reproducible builds contributors

Alexander Couzens
Bryan Newbold
Dirk Neukirchen
Felix Fietkau
Jonas Gorski
Jo-Philipp Wich
Nathan Hintz
Reiner Herrmann

TODO for tests.r-b.org/(openwrt|lede)

- we should add more variations (date, time, build path, hostname, domain, use disorderfs, CPU type, kernel, USER, HOME, SHELL, the base system).
- we should test more targets.

OpenWrt Wireless Freedom

TODO for tests.r-b.org/(openwrt|lede)

- we should add more variations (date, time, build path, hostname, domain, use disorderfs, CPU type, kernel, USER, HOME, SHELL, the base system).
- we should test more targets.
- we could build other branches too...
- we could build OpenWrt + LEDE at least every day, thanks again to Profitbricks.com.

OpenWrt Wireless Freedom

TODO for tests.r-b.org/(openwrt|lede)

- we should add more variations (date, time, build path, hostname, domain, use disorderfs, CPU type, kernel, USER, HOME, SHELL, the base system).
- we should test more targets.
- we could build other branches too...
- we could build OpenWrt + LEDE at least every day, thanks again to Profitbricks.com.
- we want to make **you** look at these pages every day!

OpenWrt Wireless Freedom

TODO: design `.buildinfo` files for OpenWrt and LEDE

- rfc822 format
- needs to define the environment
- needs to define the sources (input)
- needs to define the binaries (output)

TODO: design `.buildinfo` files for OpenWrt and LEDE

- rfc822 format
- needs to define the environment
- needs to define the sources (input)
- needs to define the binaries (output)
- Debian has only .deb files as output, while OpenWrt/LEDE have packages and images...

Rebuilds 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?

Rebuilds 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?
  - something like `.buildinfo` files (defining the environment, the input and the output(s)) will be needed everywhere, but so far we only have them for Debian...

Rebuilders and sharing signed checksums, cont.

- Individuelly 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 have 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 have unconfirmed checksums, before installing? (Y/n)"
- "How many signed checksums do you require to call a package 'reproducible'?" - and whom do you trust?

As a software developer

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

Form your reproducible builds team!

- Why?
  - Every distribution should be reproducible!
  - Learn something new everyday
  - Change the (software) world!
  - `https://tests.reproducible-builds.org/openwrt` needs **your** help
  - `https://tests.reproducible-builds.org/lede` needs **your** help
- How to get started?
  - Build something twice, run diffoscope on the results.
  - Talk to lynxis or h01ger here or talk to us on IRC or via mail.
  - RTFM, there is lots of documentation
  - Experiment - learning by doing

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

- All “Reproducible Builds” contributors
  (you are just **so** awesome!)
- OpenWrt Summit and ELCE

holger@debian.org B8BF 5413 7B09 D35C F026
FE9D 091A B856 069A AA1C
lynxis@fe80.eu 390D CF78 8BF9 AA50 4F8F
F1E2 C29E 9DA6 A0DF 8604

Questions, comments, ideas?

- https://reproducible-builds.org/
- #reproducible-builds on irc.OFTC.net
- https://lists.reproducible-builds.org
- twitter: @ReproBuild

Questions, comments, ideas?

- `https://reproducible-builds.org/`
- `#reproducible-builds` on `irc.OFTC.net`
- `https://lists.reproducible-builds.org`
- twitter: @ReproBuild
- Mike and Seth's talk from 31c3 about motivations
- Lunar's talk about fixing reproducible issues from CCCamp 15
- h01ger's talk "the Reproducible builds ecosystem" from FOSDEM 16