Reproducible Builds

An independently-verifiable path from source code to software

Reproducible
Builds

Frédéric Pierret

Oct. 7th 2022

Frédéric Pierret (fepitre)

- PhD in Applied Mathematics,
- Modeling of dynamical systems,
- Desiging build systems.

- *github.com/fepitre*
- *frederic@invisiblethingslab.com*
- *frederic.pierret@qubes-os.org*

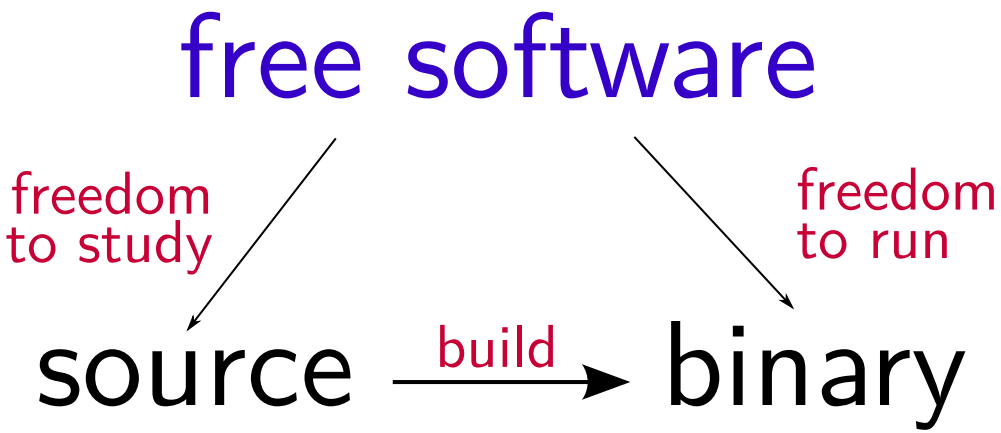
What?

What are reproducible builds?

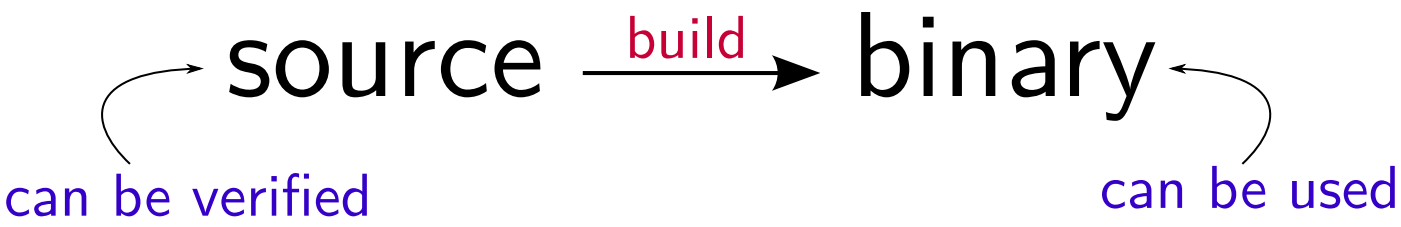
Reproducible Builds
enable anyone to reproduce
identical binary packages
from a given source

What?

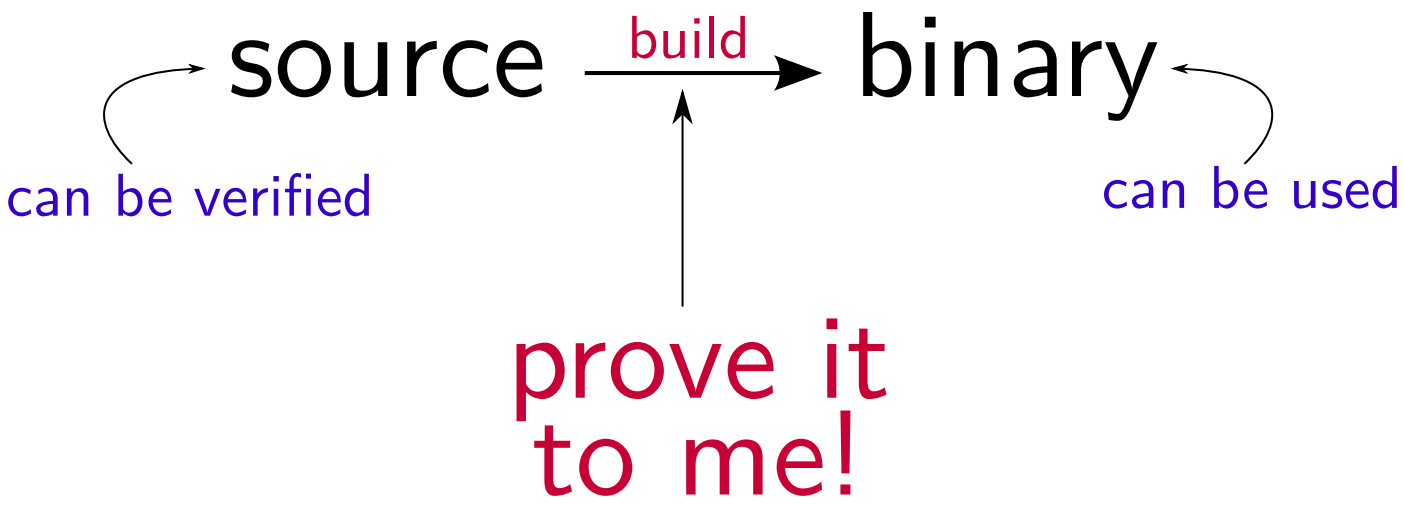
What?


free software
freedom
to study
freedom
to run
source
build
binary

What?



What?


source
build
binary
can be verified
can be used
prove it
to me!

Why?

Why?

Reproducible builds allow for **independent verifications** that a binary matches what the source intended to produce.

. . . and other nice things.

But I'm the developer!

"*I know what's in the binary because I compiled it myself!*"

"*I'm an upstanding, careful, and responsible individual!*"

"*Why should I have to worry about hypothetical risks about the contents of my binaries?*"

But the build machines are secure

- How can you be sure?

But the distribution packagers released it!

- What's in between build artifacts and signed-build artifacts?

Unpleasant thoughts

- We think of software development as a fundamentally benign activity,
  - "*I'm not that interesting.*"
- Users can be targeted through developers,
- Known successful attacks against infrastructure used by Linux (2003), FreeBSD (2013), PHP (2021) and some undisclosed.

Seriously. . .

During a CIA conference in 2012[1]:

[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.

[1] https://firstlook.org/theintercept/2015/03/10/ispy-cia-campaign-steal-apples-secrets/

And yes...

- SolarWinds[2] was exactly what would be **prevented by reproducible builds**!
- The compromission was on build servers!

[2] https://www.sans.org/blog/what-you-need-to-know-about-the-solarwinds-supply-chain-attack/

How small can a backdoor be?

OpenSSH 3.0.2 (CVE-2002-0083) – exploitable security bug (privilege escalation: user can get root)

```
 {
    Channel *c;
-   if (id < 0 || id > channels_alloc) {
+   if (id < 0 || id >= channels_alloc) {
      log("channel_lookup: %d: bad id", id);
      return;
    }
```

Result of fixing the bug (asm)

```
cmpl $0x0,0x8(%ebp)
js 16
mov 0x4,%eax
cmp %eax,0x8(%ebp)
jle 30
mov 0x8(%ebp),%eax
mov %eax,0x4(%esp)
movl $0x4c,(%esp)
call 25
```

```
cmpl $0x0,0x8(%ebp)
js 16
mov 0x4,%eax
cmp %eax,0x8(%ebp)
jl 30
mov 0x8(%ebp),%eax
mov %eax,0x4(%esp)
movl $0x4c,(%esp)
call 25
```

Result of fixing the bug (hex)

Vulnerable	Fixed
55 89 e5 83 ec	55 89 e5 83 ec
28 83 7d 08 00	28 83 7d 08 00
78 0a a1 04 00	78 0a a1 04 00
00 00 39 45 08	00 00 39 45 08
7e 1a 8b 45 08	7c 1a 8b 45 08
89 44 24 04 c7	89 44 24 04 c7
04 24 4c 00 00	04 24 4c 00 00
00 e8 fc ff ff	00 e8 fc ff ff
ff b8 00 00 00	ff b8 00 00 00
00 eb 35	00 eb 35

Resulting difference in the binary

What's the difference between `if (a > b)` and `if (a >= b)` in x86 assembly?

assembly:	JLE	JL
opcode:	0x7E	0x7C
binary:	01111110	01111100

A single bit!

Other corresponding opcode pairs also differ by just a single bit (JGE=0x7D, JG=0x7F)

Do not blame developers (all the times)

- Malicious modifications to binaries could result in irrevocable unwanted actions,
- Individual developers could be blamed for such modifications,
- Reproducible builds therefore *protect* developers

Nothing new though

From: Martin Uecker <muecker@gmx.de>
Cc: debian-devel@lists.debian.org
Date: Sun, 23 Sep 2007 23:32:59 +0200

I think it would be really cool if the Debian policy required that packages could be rebuild bit-identical from source. At the moment, it is impossible to independly verify the integricity of binary packages.

https://lists.debian.org/debian-devel/2007/09/msg00746.html

Wouldn't it be cool?

- Debian is the largest collection of free software
- More than 25,000 source packages
- "Our priorities are our **users** and **free software**"

How?

How?

- Record the build environment
- Reproduce the build environment
- Eliminate unneeded variations

How:

Record the build environment

*.buildinfo

New control file `*.buildinfo` which records:

- Versions of build dependencies
  - ... and their dependencies
- Checksum of the source package.
- Checksums of the binary packages.

Example *.buildinfo

```
Format: 1.0
Source: apt
Binary: apt apt-dbgsym apt-utils apt-utils-dbgsym libapt-pkg-dev libapt-p
Architecture: amd64
Version: 2.2.4
Build-Origin: Debian
Build-Architecture: amd64
Build-Date: Thu, 10 Jun 2021 09:12:36 +0000
Build-Path: /build/apt-oBIw5E/apt-2.2.4
Installed-Build-Depends:
 autoconf (= 2.69-14),
 automake (= 1:1.16.3-2),
 autopoint (= 0.21-4),
 autotools-dev (= 20180224.1+nmu1),
 base-files (= 11.1),
 base-passwd (= 3.5.50),
...
```

How:

Eliminate unneeded variations

Eliminate unneeded variations

Make the build process deterministic:

Same input

=

Same output

Dealing with variations: two approaches

- If a build differs because of X, you have two ways of dealing with it:
  - either make X always the same,
  - or make the build independent of X.
- Good example is build path:
  - Debian tries to not embed it,
  - Fedora or OpenSUSE always builds with the same path (e.g. Mock).

Investigating packages[3]

- diffoscope
  - figure out what makes files or directories different.
  - recursively unpack archives of many kinds and transform various binary formats into more human-readable forms for comparison.
  - It can compare two tarballs, ISO images, or PDFs just as easily.
- reprotest
  - builds the same source code in **different environments**,
  - checks the binaries produced by the builds,
  - see if changing the environment, without changing the source code, changed the generated binaries.

[3] https://reproducible-builds.org/tools/

How:

Reproduce the build

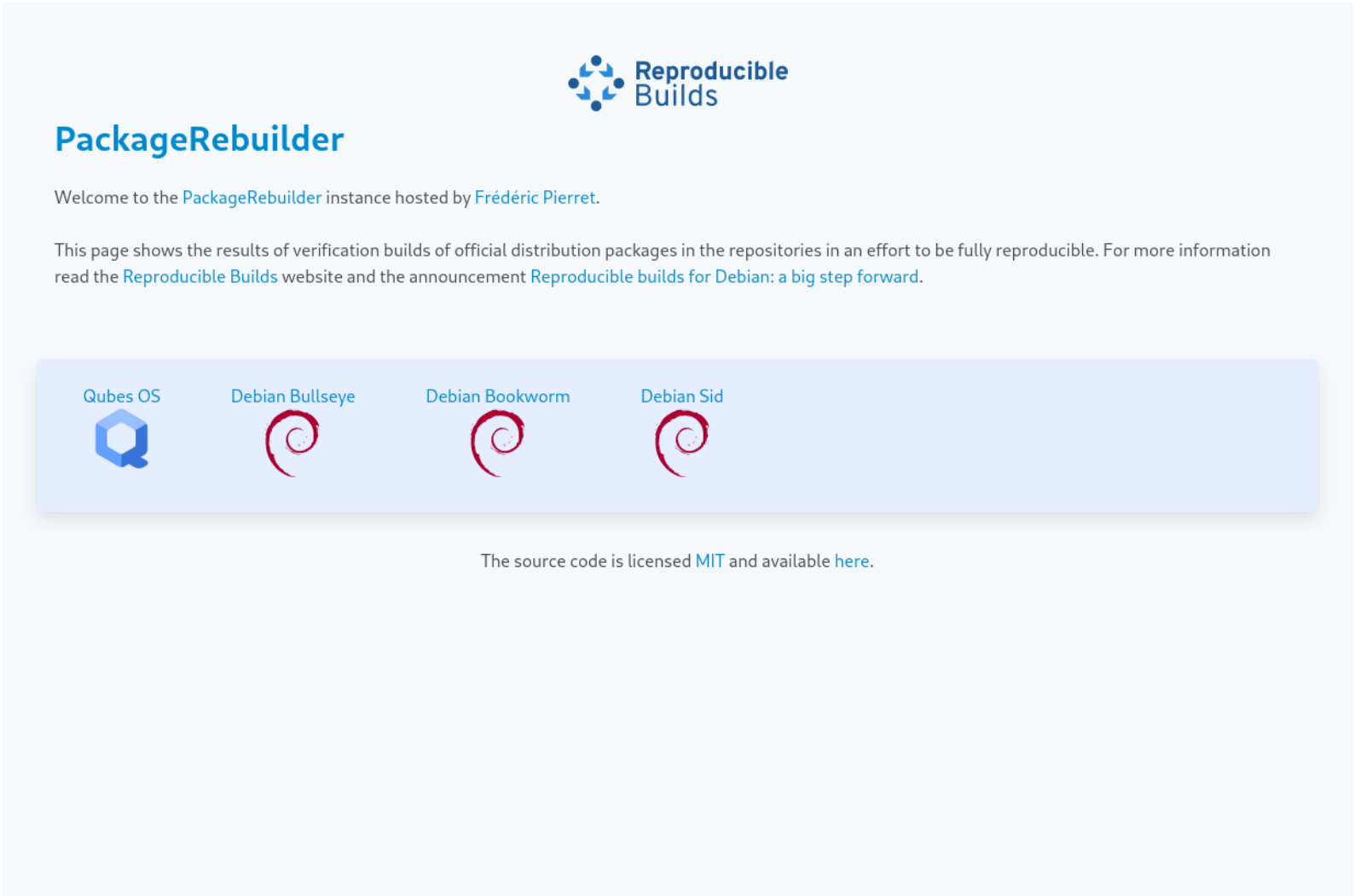
Reproduce the build for Debian aka **rebuild**

- `snapshot.notset.fr`[4]
  - A **working** replacement of `snapshot.debian.org`,
  - History of Debian repositories from 2017 to now,
  - amd64, all and sources (upcoming arm64)
- debrebuild[5]
  - Find the right archive snapshot,
  - Install packages listed in the `*.buildinfo` file,
  - Do the rebuild.

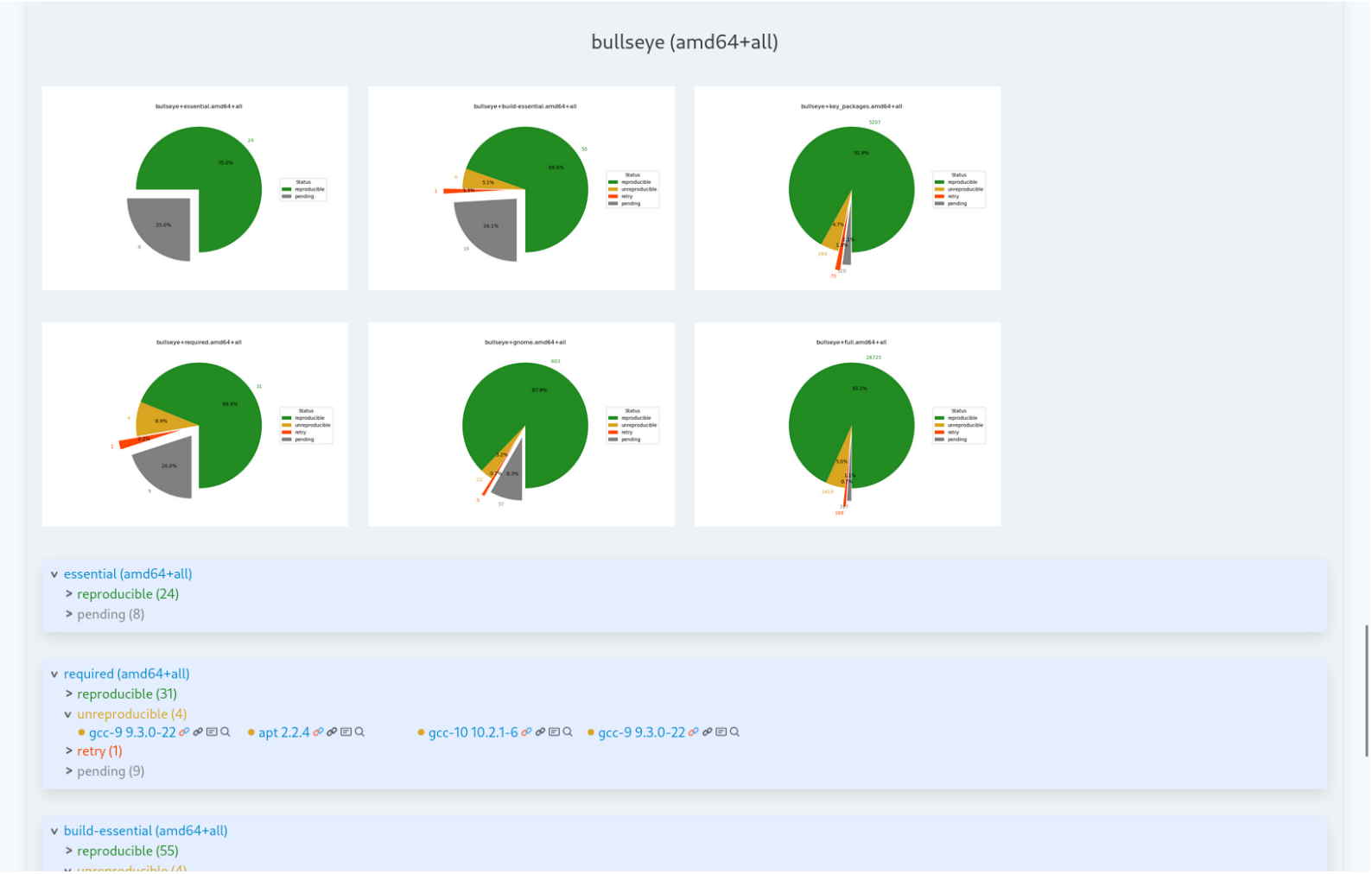
[4] https://github.com/fepitre/debian-snapshot

[5] https://github.com/fepitre/debrebuild

beta.tests.reproducible-builds.org



beta.tests.reproducible-builds.org



in-toto format[6]


```
{
 "signatures": [
  {
   "keyid": "8deb0bef1d99feb8b9a90fb192ef6d6141641e5c",
   "other_headers": "04000108001d1621048deb0bef1d99feb8b9a90fb192ef6d6141641e5c05026157bb29",
   "signature": "1f0b097baa2b82105ba4054966fa84f35eada6088b218e5856ae8bd7fef04eb37ab2e2c0e8f5d9d7b238704d11(
  }
 ],
 "signed": {
  "name": "rebuild",
  "products": {
   "apt_2.2.4_amd64.deb": {
    "sha256": "75f07c4965ff0813f26623a1164e162538f5e94defba6961347527ed71bc4f3d"
   },
(...)
  }
 }
}
```


[6] https://in-toto.io/

beta.tests.reproducible-builds.org

```
root@debian-11:~# apt-get install tzdata
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages will be upgraded:
  tzdata
1 upgraded, 0 newly installed, 0 to remove and 842 not upgraded.
Need to get 284 kB of archives.
After this operation, 4,096 B of additional disk space will be used.
Get:1 intoto://deb.debian.org/debian bullseye/main amd64 tzdata all 2021a-1 [284 kB]
(...)
In-toto verification for '/var/cache/apt/archives/partial/tzdata_2021a-1_all.deb' passed! :)
Fetched 284 kB in 15s (19.3 kB/s)
Reading changelogs... Done
(...)
```

Test

(and test again)

tests.reproducible-builds.org

- Continuous tests,
- Archlinux, coreboot, Debian, FreeBSD, NetBSD, OpenWrt, GNU Guix, NixOS, openSUSE, Qubes OS, Yocto Project.

https://tests.reproducible-builds.org

Purpose:

- Build the package,
- *R*ebuild the package,
- Compare the results,
- This is **not** rebuild.

https://tests.reproducible-builds.org

The second build is different in:

- time,
- file ordering,
- CPU ordering and availability,
- hostname,
- user & group,
- locale,
- kernel,
- etc.

Findings

Identified issues

- Timestamps (recording current time),
- File order,
- (Pseudo-)randomness:
  - Temporary file paths,
  - UUID,
  - Protection against complexity attacks.

Identified issues (cont.)

- CPU and memory related:
  - Code optimizations for current CPU class,
  - Recording of memory addresses,
- Build path,
- Others, e.g. locale settings.

Examples

Timestamps added by build systems

Timestamps in static libraries

data.tar

./usr/lib/libform_g.a

metadata

```
234  14112 Jan  2 06:04 2015 f_trace.o      234  14112 Jan  2 06:22 2015 f_trace.o
234  14336 Jan  2 06:04 2015 fld_arg.o      234  14336 Jan  2 06:22 2015 fld_arg.o
234  16112 Jan  2 06:04 2015 fld_attr.o     234  16112 Jan  2 06:22 2015 fld_attr.o
234  16368 Jan  2 06:04 2015 fld_current.   234  16368 Jan  2 06:22 2015 fld_current.
234  23592 Jan  2 06:04 2015 fld_def.o      234  23592 Jan  2 06:22 2015 fld_def.o
234  14920 Jan  2 06:04 2015 fld_dup.o      234  14920 Jan  2 06:22 2015 fld_dup.o
234  13064 Jan  2 06:04 2015 fld_ftchoice   234  13064 Jan  2 06:22 2015 fld_ftchoice
234  13992 Jan  2 06:04 2015 fld_ftlink.c   234  13992 Jan  2 06:22 2015 fld_ftlink.c
234  13992 Jan  2 06:04 2015 fld_info.o     234  13992 Jan  2 06:22 2015 fld_info.o
234  13616 Jan  2 06:04 2015 fld_just.o     234  13616 Jan  2 06:22 2015 fld_just.o
234  14688 Jan  2 06:04 2015 fld_link.o     234  14688 Jan  2 06:22 2015 fld_link.o
234  13472 Jan  2 06:04 2015 fld_max.o      234  13472 Jan  2 06:22 2015 fld_max.o
234  13208 Jan  2 06:04 2015 fld_move.o     234  13208 Jan  2 06:22 2015 fld_move.o
234  16296 Jan  2 06:04 2015 fld_newftyp.   234  16296 Jan  2 06:22 2015 fld_newftyp.
234  16232 Jan  2 06:04 2015 fld_opts.o     234  16232 Jan  2 06:22 2015 fld_opts.o
234  14312 Jan  2 06:04 2015 fld_pad.o      234  14312 Jan  2 06:22 2015 fld_pad.o
234  13616 Jan  2 06:04 2015 fld_page.o     234  13616 Jan  2 06:22 2015 fld_page.o
234  13504 Jan  2 06:04 2015 fld_stat.o     234  13504 Jan  2 06:22 2015 fld_stat.o
234  14912 Jan  2 06:04 2015 fld_type.o     234  14912 Jan  2 06:22 2015 fld_type.o
234  13488 Jan  2 06:04 2015 fld_user.o     234  13488 Jan  2 06:22 2015 fld_user.o
```

Timestamps by a template engine

data.tar

./usr/lib/python2.7/dist-packages/spectacle/dsc/__init__.py

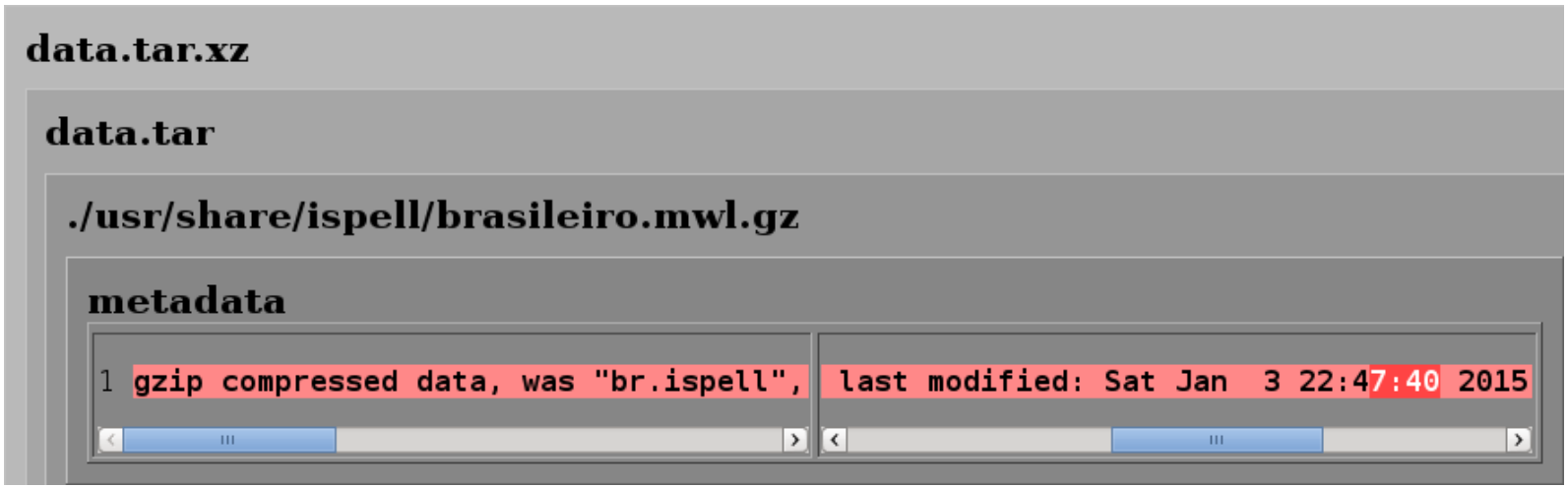
```
 1 +-- 28 lines: #!/usr/bin/env python--
29 VFFSL=valueFromFrameOrSearchList
30 VFSL=valueFromSearchList
31 VFN=valueForName
32 currentTime=time.time
33 __CHEETAH_version__ = '2.4.4'
34 __CHEETAH_versionTuple__ = (2, 4, 4,
35 __CHEETAH_genTime__ = 1421397870.7792
36 __CHEETAH_genTimestamp__ = 'Fri Jan 1
37 __CHEETAH_src__ = 'dsc.tmpl'
38 __CHEETAH_srcLastModified__ = 'Fri Feb
39 __CHEETAH_docstring__ = 'Autogenerated
40
41 if __CHEETAH_versionTuple__ < Required
42     raise AssertionError(
43 +--127 lines: 'This template was comp
```

```
usr/bin/env python-----------------------
ameOrSearchList
rchList

time
__ = '2.4.4'
Tuple__ = (2, 4, 4, 'development', 0)
__ = 1421397902.668939
stamp__ = 'Fri Jan 16 08:45:02 2015'
 'dsc.tmpl'
Modified__ = 'Fri Feb 25 06:40:15 2011'
ng__ = 'Autogenerated by Cheetah: The Pyth

ionTuple__ < RequiredCheetahVersionTuple:
onError(
is template was compiled with Cheetah vers
```

Examples

Archives

Timestamps in gzip headers



Timestamps in ZIP archives

data.tar.xz

data.tar

./usr/share/boa-constructor/Docs/boa.apphelp.htb

metadata

```
+--249 lines: Archive:  boa.apphelp.htb-----------------------------------------
  minimum file system compatibility requ ired:     MS-DOS, OS/2 or NT FAT
  minimum software version required to e xtract:   2.0
  compression method:                              deflated
  compression sub-type (deflation):                normal
  file security status:                            not encrypted
  extended local header:                           no
  file last modified on (DOS date/time):           2015 Jan 16 09:55:52
  file last modified on (UT extra field  modtime): 2015 Jan 16 09:55:52 local
  file last modified on (UT extra field  modtime): 2015 Jan 16 09:55:52 UTC
  32-bit CRC value (hex):                          5bb8662c
  compressed size:                                 1660 bytes
  uncompressed size:                               5682 bytes
  length of filename:                              19 characters
  length of extra field:                           24 bytes
  length of file comment:                          0 characters
```

Timestamps in tarballs

data.tar.xz

data.tar

./usr/share/doc/allegro4-doc/examples/source.tar.gz

source.tar

metadata

54 2015-01-09 22:29:44 CMakeLists.txt	54 2015-01-09 22:32:29 CMakeLists.txt
08 2015-01-09 22:29:44 afinfo.c	08 2015-01-09 22:32:29 afinfo.c
19 2015-01-09 22:29:44 akaitest.c	19 2015-01-09 22:32:29 akaitest.c
32 2015-01-09 22:29:44 allegro.pcx	32 2015-01-09 22:32:29 allegro.pcx
70 2015-01-09 22:29:44 digitest.c	70 2015-01-09 22:32:29 digitest.c
10 2015-01-09 22:29:44 ex12bit.c	10 2015-01-09 22:32:29 ex12bit.c
46 2015-01-09 22:29:44 ex3buf.c	46 2015-01-09 22:32:29 ex3buf.c
57 2015-01-09 22:29:44 ex3d.c	57 2015-01-09 22:32:29 ex3d.c
34 2015-01-09 22:29:44 exaccel.c	34 2015-01-09 22:32:29 exaccel.c
06 2015-01-09 22:29:44 exalpha.c	06 2015-01-09 22:32:29 exalpha.c
62 2015-01-09 22:29:44 example.dat	62 2015-01-09 22:32:29 example.dat
65 2015-01-09 22:29:44 example.h	65 2015-01-09 22:32:29 example.h
30 2015-01-09 22:29:44 examples.txt	30 2015-01-09 22:32:29 examples.txt

Users and groups in tarballs

data.tar.xz

data.tar

./usr/share/doc/raster3d/diffs.tar.gz

diffs.tar

metadata

```
 1 -rw-r--r-- pbuilder1/pbuilder1
 2 -rw-r--r-- pbuilder1/pbuilder1
 3 -rw-r--r-- pbuilder1/pbuilder1
 4 -rw-r--r-- pbuilder1/pbuilder1
 5 -rw-r--r-- pbuilder1/pbuilder1
 6 -rw-r--r-- pbuilder1/pbuilder1
 7 -rw-r--r-- pbuilder1/pbuilder1
 8 -rw-r--r-- pbuilder1/pbuilder1
 9 -rw-r--r-- pbuilder1/pbuilder1
10 -rw-r--r-- pbuilder1/pbuilder1
11 -rw-r--r-- pbuilder1/pbuilder1
12 -rw-r--r-- pbuilder1/pbuilder1
```

```
 1 -rw-r--r-- pbuilder2/pbuilder2
 2 -rw-r--r-- pbuilder2/pbuilder2
 3 -rw-r--r-- pbuilder2/pbuilder2
 4 -rw-r--r-- pbuilder2/pbuilder2
 5 -rw-r--r-- pbuilder2/pbuilder2
 6 -rw-r--r-- pbuilder2/pbuilder2
 7 -rw-r--r-- pbuilder2/pbuilder2
 8 -rw-r--r-- pbuilder2/pbuilder2
 9 -rw-r--r-- pbuilder2/pbuilder2
10 -rw-r--r-- pbuilder2/pbuilder2
11 -rw-r--r-- pbuilder2/pbuilder2
12 -rw-r--r-- pbuilder2/pbuilder2
```

Examples

Timestamps in documentation

Timestamps written by Doxygen

data.tar.xz

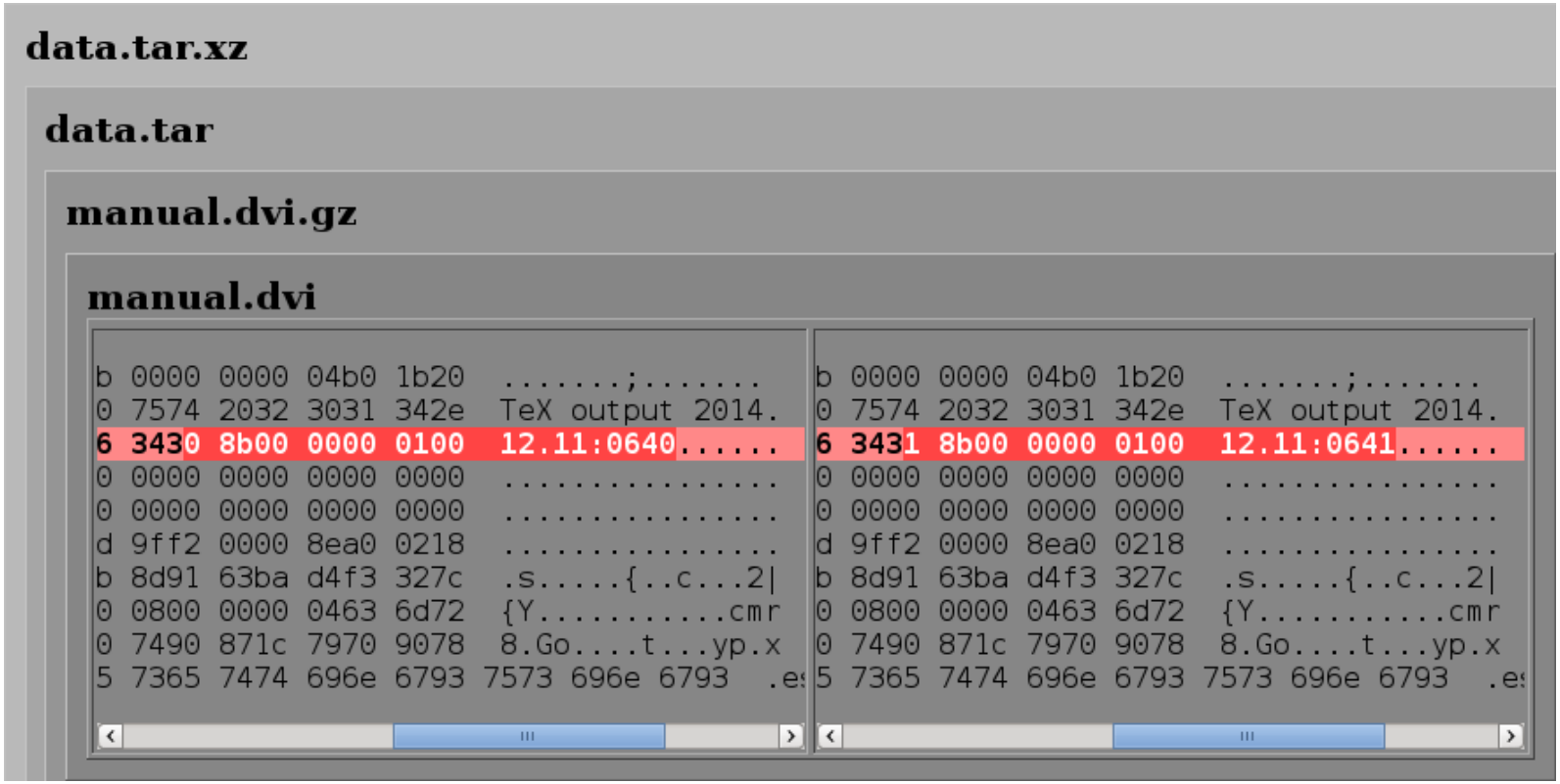
data.tar

./usr/share/doc/libnfo-doc/html/dir_68267d1309a1af8e8297ef4c3efbcdba.html

```
: <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
emitem:nfo_8h"><td class="memItemLeft" ali
eparator:"><td class="memSeparator" colspa

ontents -->
ooter part -->
ooter"/><address class="footer"><small>
 Mon Dec 15 2014 04:03:26 for libnfo by &
footer" src="doxygen.png" alt="doxygen"/>

dress>
```

```
 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
mitem:nfo_8h"><td class="memItemLeft" alig
parator:"><td class="memSeparator" colspar

ntents -->
oter part -->
oter"/><address class="footer"><small>
Mon Dec 15 2014 04:04:03 for libnfo by &#1
ooter" src="doxygen.png" alt="doxygen"/>

ress>
```

Timestamps in TeX output (.dvi)



Examples

"Compiled at/on/by"

Build time via C preprocessor macros

```
80    {"to",1,NULL,'t'},
81     {0,0,0,0}
82  };
83
84  #define MSGSIZE 512
85  #define INPUT_TEXT_SIZE_INIT 32
86
87  char *pgname ="numconv";
88  char compdate[]="Compiled " __DATE__ " " __TIME__ ;
89
90  void ShowNumberSystems(int which) {
```

package info (click to toggle)

libuninum 2.7-1.1

- links: PTS
- area: main
- in suites: jessie, sid, wheezy
- size: 1,972 kB
- SLOC: ansic: 9,968; sh: 8,715; tcl: 553; makefile: 78

Build time recorded via Makefile

```
149 genversion.$(OBJEXT): $(genversion_SOURCES) genversion.h
150
151 genversion.h: $(top_builddir)/config.status
152         -rm -f $@ $@.new
153         echo '#define CC "$(CC)"' > $@.new
154         echo '#define BUILT_DATE "'`date`'"' >> $@.new
155         echo '#define BUILT_MACH "$(target)"' >> $@.new
156         mv $@.new $@
157
158 version.c: genversion$(EXEEXT)
159         -rm -f version.c
160         ./genversion$(EXEEXT) > version.c
161
162 BUILT_SOURCES += genversion.h version.c
```

package info (click to toggle)

amanda 1:3.3.6-4

- links: PTS, VCS
- area: main
- in suites: jessie, sid
- size: 25,248 kB
- ctags: 28,823
- SLOC: ansic: 225,464; perl:

Hostname recorded via ./configure

```
364       VERSION="$VERSION (svn$SVN_REV)"
365     fi
366   fi
367   HOSTNAME=`hostname`
368   DATE=`date +"%d.%m.%Y %H:%M:%S %Z"`
369
370   cat > version.h <<EOF
371   /*
372    * anytun version info
373    *
374    * this file was created automatically
375    * do not edit this file directly
376    * use ./configure instead
377    */
378
379   #ifndef ANYTUN_version_h_INCLUDED
380   #define ANYTUN_version_h_INCLUDED
381
382   #define VERSION_STRING_0 " version $VERSION"
383   #define VERSION STRING 1 "built on $HOSTNAME, $DATE"
```

package info (click to toggle)

anytun 0.3.5-1

- links: PTS, VCS
- area: main
- in suites: jessie, sid
- size: 1,424 kB
- ctags: 1,339
- SLOC: cpp: 9,126; sh: 618; makefile: 367

m4 macros for autoconf (data, build time, username, hostname)

```
75    AC_FLDIGI_SH_DQ([echo $ac_configure_args])
76    AC_DEFINE_UNQUOTED([BUILD_CONFIGURE_ARGS], [$ac_sh_dq], [Configure arguments])
77
78    AC_FLDIGI_SH_DQ([date])
79    AC_DEFINE_UNQUOTED([BUILD_DATE], [$ac_sh_dq], [Build date])
80
81    AC_FLDIGI_SH_DQ([whoami])
82    AC_DEFINE_UNQUOTED([BUILD_USER], [$ac_sh_dq], [Build user])
83
84    AC_FLDIGI_SH_DQ([hostname])
85    AC_DEFINE_UNQUOTED([BUILD_HOST], [$ac_sh_dq], [Build host])
86
87    AC_FLDIGI_SH_DQ([$CXX -v 2>&1 | tail -1])
88    AC_DEFINE_UNQUOTED([BUILD_COMPILER], [$ac_sh_dq], [Compiler])
```

Examples

File ordering

File ordering in python-support files

Examples

Randomness

Random Python hash order

./usr/share/eric/modules/Cooperation/Ui_ChatWidget.py

Offset 201, 10 lines modified		Offset 201, 10 lines modified	
	····self.clearHostsButton.		····self.clearHostsButton.
201	setText(_translate("ChatWidget",· "Clear",·None))	201	setText(_translate("ChatWidget",· "Clear",·None))
	····self.connectionLed.		····self.connectionLed.
202	setToolTip(_translate("ChatWidget",· "Shows·the·connection·status",·None))	202	setToolTip(_translate("ChatWidget",· "Shows·the·connection·status",·None))
	····self.serverGroup.		····self.serverGroup.
203	setTitle(_translate("ChatWidget",· "Server",·None))	203	setTitle(_translate("ChatWidget",· "Server",·None))
	····self.label_4.		····self.label_4.
204	setText(_translate("ChatWidget",· "Port:",·None))	204	setText(_translate("ChatWidget",· "Port:",·None))
	····self.serverPortSpin.		····self.serverPortSpin.
205	setToolTip(_translate("ChatWidget",· "Enter·the·server·port",·None))	205	setToolTip(_translate("ChatWidget",· "Enter·the·server·port",·None))
	····self.serverLed.		····self.serverLed.
206	setToolTip(_translate("ChatWidget",· "Shows·the·status·of·the·server",· None))	206	setToolTip(_translate("ChatWidget",· "Shows·the·status·of·the·server",· None))
207		207	
208	from·E5Gui.E5Led·import·E5Led		
209	from·E5Gui.E5ComboBox·import· E5ClearableComboBox	208	from·E5Gui.E5ComboBox·import· E5ClearableComboBox
210	from·E5Gui.E5LineEdit·import· E5ClearableLineEdit	209	from·E5Gui.E5LineEdit·import· E5ClearableLineEdit
		210	from·E5Gui.E5Led·import·E5Led

Examples

Even more timestamps!

Timestamps in PNG

Even images!

data.tar.xz

data.tar

./usr/share/icons/hicolor/128x128/apps/hedgewars.png

sng

```
  1 +--104 lines: #SNG: from stdin------
105     ( 64,141,230)     # rgb = (0x40,(
106 }
107 bKGD {index: 12}
108 tRNS {
109  0}
110 tIME {
111     # 31 Dec 2014 05:46:02 GMT
112     year:   2014
113     month:  12
114     day:    31
115     hour:   5
116     minute: 46
117     second: 2
118 }
119 tEXt {
120     keyword: "date:create";
121     text: "2014-12-31T05:46:00+00:00
122 }
```

```
  1 +--104 lines: #SNG: from stdin------
105     ( 64,141,230)     # rgb = (0x40,(
106 }
107 bKGD {index: 12}
108 tRNS {
109  0}
110 tIME {
111     # 31 Dec 2014 05:52:27 GMT
112     year:   2014
113     month:  12
114     day:    31
115     hour:   5
116     minute: 52
117     second: 27
118 }
119 tEXt {
120     keyword: "date:create";
121     text: "2014-12-31T05:52:25+00:00
122 }
```

Please help!

Please help!

- Do not record time, username, hostname, kernel version...
  - ... or make it optional.
- Sort file paths.
- Sort dictionary keys.
- If you work for a project where we propose patches, please help into merging them!

Help?

- Inventory issues,
- Make packages build reproducibly,
- Fix known common issues:
  - Get reproducible PE binaries,
  - Random filenames with GCC (e.g. annobin),
  - ...
- Debian archive infrastructure
  - Store and distribute `*.buildinfo` files,
  - ...
- Tools to display local packages reproducibility status (reprotest, diffoscope, etc.).

Summer news

- NSA, CISA, ODNI released `Securing the Software Supply Chain: Recommended Practices Guide for Developers`[7]
- The document expressly recommends having reproducible builds as part of **advanced** recommended mitigations, along with hermetic builds. Page 31 (page 35 in the PDF) says:

  Reproducible builds provide additional protection and validation against attempts to compromise build systems. They ensure the binary products of each build system match: i.e., they are built from the same source, regardless of variable metadata such as the order of input files, timestamps, locales, and paths. (. . .)

[7] https://media.defense.gov/2022/Sep/01/2003068942/-1/-1/0/ESF_SECURING_THE_SOFTWARE_SUPPLY_CHAIN_DEVELOPERS.PDF

Stay in touch

- Website: https://reproducible-builds.org/,
- Mailing lists: `rb-general@lists.reproducible-builds.org`,
- Join `#reproducible-builds` or `#debian-reproducible` (OFTC).
- **https://reproducible-builds.org/events/venice2022/**

Thank you for your attention.

Questions? Comments?

https://reproducible-builds.org/

```
77EE EF6D 0386 962A EA8C F84A 9B82 73F8 0AC2 19E6
9FA6 4B92 F95E 706B F28E 2CA6 4840 10B5 CDC5 76E2
```