Reproducible builds

From Wikipedia, the free encyclopedia

[Jump to navigation](https://en.wikipedia.org/wiki/Reproducible_builds#mw-head)[Jump to search](https://en.wikipedia.org/wiki/Reproducible_builds#searchInput)

[](https://en.wikipedia.org/wiki/File:Reproducible_Builds_logo.svg)

**Reproducible builds**, also known as deterministic compilation, is a process of [compiling](https://en.wikipedia.org/wiki/Compiling) software which ensures the resulting [binary code](https://en.wikipedia.org/wiki/Binary_code) can be [reproduced](https://en.wikipedia.org/wiki/Reproducibility). [Source code](https://en.wikipedia.org/wiki/Source_code) compiled using deterministic compilation will always output the same binary.[[1]](https://en.wikipedia.org/wiki/Reproducible_builds#cite_note-reproducible-builds-homepage-1)[[2]](https://en.wikipedia.org/wiki/Reproducible_builds#cite_note-2)

Reproducible builds can act as part of a [chain of trust](https://en.wikipedia.org/wiki/Chain_of_trust);[[1]](https://en.wikipedia.org/wiki/Reproducible_builds#cite_note-reproducible-builds-homepage-1) the source code can be signed, and deterministic compilation can prove that the binary was compiled from trusted source code.



**Contents**

* [1Methods](https://en.wikipedia.org/wiki/Reproducible_builds#Methods)
* [2History](https://en.wikipedia.org/wiki/Reproducible_builds#History)
* [3References](https://en.wikipedia.org/wiki/Reproducible_builds#References)
* [4External links](https://en.wikipedia.org/wiki/Reproducible_builds#External_links)

Methods[[edit](https://en.wikipedia.org/w/index.php?title=Reproducible_builds&action=edit&section=1)]

For the compilation process to be deterministic, the input to the compiler must be the same, regardless of the build environment used. This typically involves normalizing [variables](https://en.wikipedia.org/wiki/Environment_variable) that may change, such as order of input files, [timestamps](https://en.wikipedia.org/wiki/Timestamps), [locales](https://en.wikipedia.org/wiki/Locale_(computer_software)), and [paths](https://en.wikipedia.org/wiki/Path_(computing)).

Additionally, the compilers must not introduce non-determinism themselves. This sometimes happens when using hash tables with a random hash seed value. It can also happen when using the address of variables because that varies from [address space layout randomization](https://en.wikipedia.org/wiki/Address_space_layout_randomization) (ASLR).

[Build systems](https://en.wikipedia.org/wiki/Build_system), such as [Bazel](https://en.wikipedia.org/wiki/Bazel_(software)) and Gitian,[[3]](https://en.wikipedia.org/wiki/Reproducible_builds#cite_note-3) can be used to automate deterministic build processes.

History[[edit](https://en.wikipedia.org/w/index.php?title=Reproducible_builds&action=edit&section=2)]

The GNU project used reproducible builds in the early 1990s. Changelogs from 1992 indicate the ongoing effort. [[4]](https://en.wikipedia.org/wiki/Reproducible_builds#cite_note-4)

One of the older[[5]](https://en.wikipedia.org/wiki/Reproducible_builds#cite_note-5) projects to promote reproducible builds is the [Bitcoin](https://en.wikipedia.org/wiki/Bitcoin) project with Gitian. Later, in 2013, the [Tor (anonymity network)](https://en.wikipedia.org/wiki/Tor_(anonymity_network)) project started using Gitian for their reproducible builds.[[6]](https://en.wikipedia.org/wiki/Reproducible_builds#cite_note-6)

In July 2013 on the [Debian](https://en.wikipedia.org/wiki/Debian) project started implementing reproducible builds across its entire package archive.[[7]](https://en.wikipedia.org/wiki/Reproducible_builds#cite_note-7)[[8]](https://en.wikipedia.org/wiki/Reproducible_builds#cite_note-8)

By July 2017 more than 90% of the packages in the repository have been proven to build reproducibly.[[9]](https://en.wikipedia.org/wiki/Reproducible_builds#cite_note-9)

In November 2018, the Reproducible Builds project joined the [Software Freedom Conservancy](https://en.wikipedia.org/wiki/Software_Freedom_Conservancy).[[10]](https://en.wikipedia.org/wiki/Reproducible_builds#cite_note-10)

[F-droid](https://en.wikipedia.org/wiki/F-Droid) uses reproducible builds to provide a guarantee that the distributed APKs use the claimed [free source code](https://en.wikipedia.org/wiki/Free_software).[[11]](https://en.wikipedia.org/wiki/Reproducible_builds#cite_note-11)

References[[edit](https://en.wikipedia.org/w/index.php?title=Reproducible_builds&action=edit&section=3)]

* 1. ^ [Jump up to:***a***](https://en.wikipedia.org/wiki/Reproducible_builds#cite_ref-reproducible-builds-homepage_1-0) [***b***](https://en.wikipedia.org/wiki/Reproducible_builds#cite_ref-reproducible-builds-homepage_1-1) [*"reproducible-builds.org"*](https://reproducible-builds.org/)*. reproducible-builds.org.*[*Archived*](https://web.archive.org/web/20160520123008/https:/reproducible-builds.org/)*from the original on 20 May 2016. Retrieved 22 August 2016. 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....build system needs to be made entirely deterministic: transforming a given source must always create the same result.*
  2. [**^**](https://en.wikipedia.org/wiki/Reproducible_builds#cite_ref-2) *Ratliff, Emily (4 April 2016).*[*"Establishing Correspondence Between an Application and its Source Code | SecurityWeek.com"*](http://www.securityweek.com/establishing-correspondence-between-application-and-its-source-code)*. www.securityweek.com. SecurityWeek.*[*Archived*](https://web.archive.org/web/20160920014341/http:/www.securityweek.com/establishing-correspondence-between-application-and-its-source-code)*from the original on 20 September 2016. Retrieved 22 August 2016.*
  3. [**^**](https://en.wikipedia.org/wiki/Reproducible_builds#cite_ref-3) [*"Gitian: a secure software distribution method"*](https://gitian.org/)*. gitian.org. Retrieved 2018-01-10.*
  4. [**^**](https://en.wikipedia.org/wiki/Reproducible_builds#cite_ref-4) [*"Email describing GNU reproducible builds"*](https://lists.reproducible-builds.org/pipermail/rb-general/2017-January/000309.html)*.*
  5. [**^**](https://en.wikipedia.org/wiki/Reproducible_builds#cite_ref-5) [*"LICENSE-file of the Gitian-Project"*](https://github.com/devrandom/gitian-builder/blob/master/LICENSE)*. Retrieved 2019-12-03.*
  6. [**^**](https://en.wikipedia.org/wiki/Reproducible_builds#cite_ref-6) [Deterministic Builds Part Two: Technical Details.](https://blog.torproject.org/blog/deterministic-builds-part-two-technical-details) October 04, 2013
  7. [**^**](https://en.wikipedia.org/wiki/Reproducible_builds#cite_ref-7) [*"Reproducible Builds talk in Debian"*](http://penta.debconf.org/dc13_schedule/events/1063.en.html)*.*
  8. [**^**](https://en.wikipedia.org/wiki/Reproducible_builds#cite_ref-8) [*"Reproducible Builds history"*](https://wiki.debian.org/ReproducibleBuilds/History)*.*
  9. [**^**](https://en.wikipedia.org/wiki/Reproducible_builds#cite_ref-9) [*"Linux-Distributionen: Mehr als 90 Prozent der Debian-Pakete reproduzierbar - Golem.de"*](https://www.golem.de/news/linux-distributionen-mehr-als-90-prozent-der-debian-pakete-reproduzierbar-1707-129094.html)*(in German). 2017-07-24. Retrieved 2018-10-30.*
  10. [**^**](https://en.wikipedia.org/wiki/Reproducible_builds#cite_ref-10) [*"Reproducible Builds joins the Software Freedom Conservancy"*](https://reproducible-builds.org/news/2018/11/08/reproducible-builds-joins-software-freedom-concervancy/)*. Retrieved 2018-12-15.*
  11. [**^**](https://en.wikipedia.org/wiki/Reproducible_builds#cite_ref-11) [*"Reproducible Builds"*](https://f-droid.org/docs/Reproducible_Builds/)*. F-Droid.*

External links[[edit](https://en.wikipedia.org/w/index.php?title=Reproducible_builds&action=edit&section=4)]

* [reproducible-builds.org](https://reproducible-builds.org/)
* [Debian Reproducible Builds](https://wiki.debian.org/ReproducibleBuilds)

|  |  |
| --- | --- |
| [Stub icon](https://en.wikipedia.org/wiki/File:Monitor_padlock.svg) | *This*[*computer security*](https://en.wikipedia.org/wiki/Computer_security)*article is a*[*stub*](https://en.wikipedia.org/wiki/Wikipedia:Stub)*. You can help Wikipedia by*[*expanding it*](https://en.wikipedia.org/w/index.php?title=Reproducible_builds&action=edit)*.* |

|  |  |
| --- | --- |
| [Stub icon](https://en.wikipedia.org/wiki/File:Software_spanner.png) | *This*[*computer-programming*](https://en.wikipedia.org/wiki/Computer_programming)*-related article is a*[*stub*](https://en.wikipedia.org/wiki/Wikipedia:Stub)*. You can help Wikipedia by*[*expanding it*](https://en.wikipedia.org/w/index.php?title=Reproducible_builds&action=edit)*.* |

**Definitions**

When is a build reproducible?

A build is **reproducible** if given the same source code, build environment and build instructions, any party can recreate bit-by-bit identical copies of all specified artifacts.

The relevant attributes of the build environment, the build instructions and the source code as well as the expected reproducible artifacts are defined by the authors or distributors. The artifacts of a build are the parts of the build results that are the desired primary output.

Explanations

**Source code** is usually a checkout from version control at a specific revision or a source code archive.

**Relevant attributes of the build environment** would usually include dependencies and their versions, build configuration flags and environment variables as far as they are used by the build system (eg. the locale). It is preferable to reduce this set of attributes.

**Artifacts** would include executables, distribution packages or filesystem images. They would not usually include build logs or similar ancillary outputs.

The reproducibility of artifacts is **verified** by bit-by-bit comparison. This is usually performed using cryptographically secure hash functions.

**Authors or distributors** means parties that claim reproducibility of a set of artifacts. These may be upstream authors, distribution maintainers or any other distributor.