Build Reproducibility for Software Integrity

Vivi Andersson, vivia@kth.se

KTH EECS, CHAINS Research Group

2024-05-23

Introduction

- **KTH 2019-2024**
  - MSc. Computer Science, Software Engineering
  - **Interests:** (SC) Security, Blockchains

- **CHAINS Research Group**
  - **Lead:** Musard Balliu, Benoit Baudry, Mathias Ekstedt, Martin Monperrus
  - **Topics:** Software Supply Chains > Dependencies, Builds, SBOMs…
  - **My project:** Go Ethereum SSC Security

Agenda: Reproducible Builds

- Motivation: Software Supply Chains
- Reproducible builds: Why and What?
- Causes for **un**reproducible builds
- In practice: Experiences from Go Ethereum
- Future Outlook

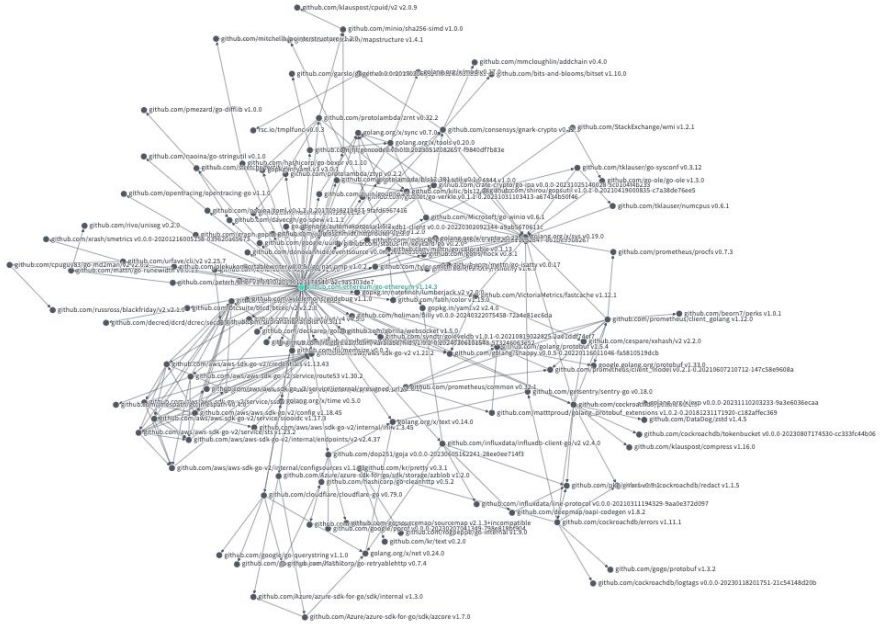
Reproducible Builds for Software Integrity

Software Supply Chains: Motivation

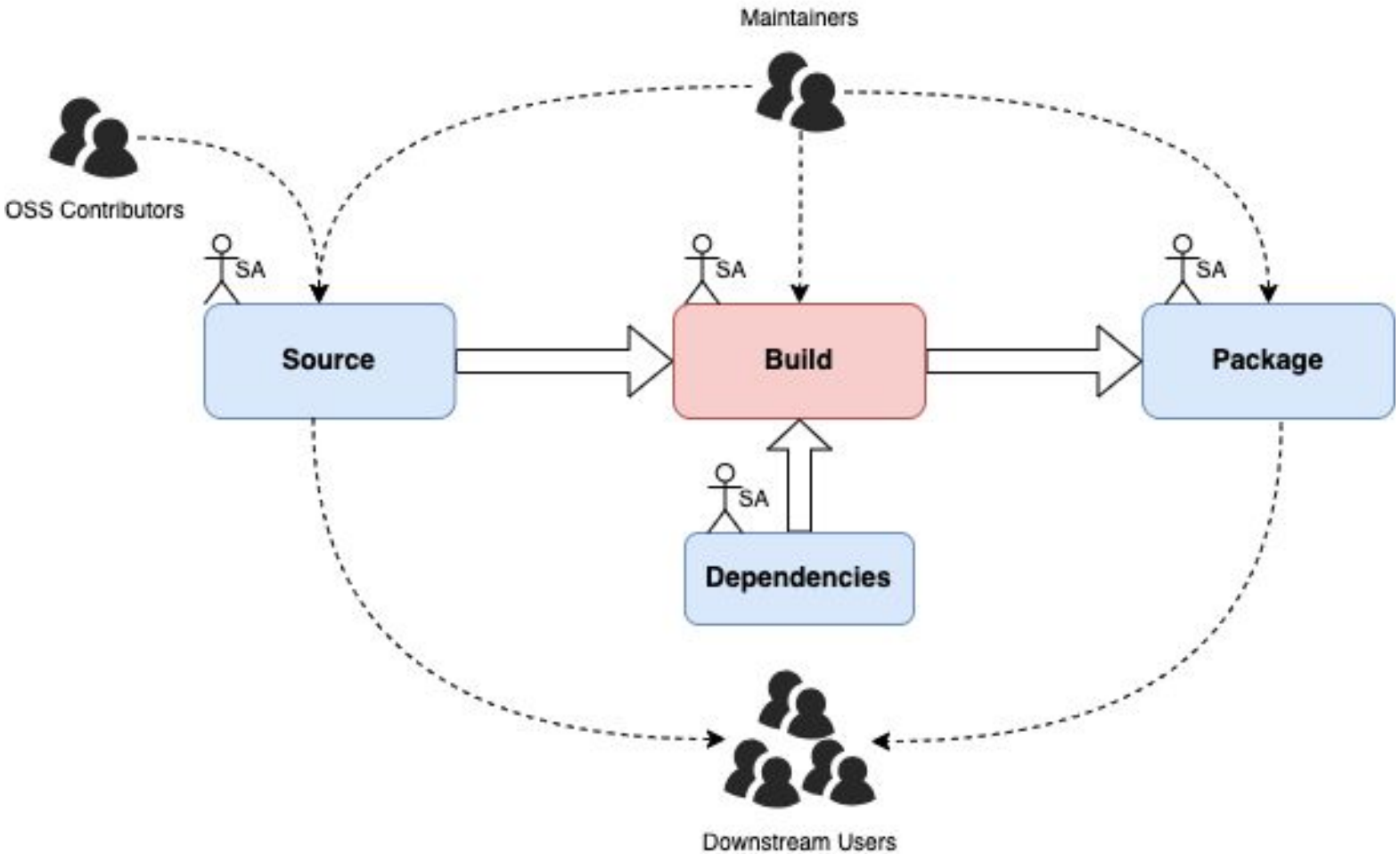
Definition

A software supply chain is defined as all components involved in producing a final software artifact [1].

- **Threat landscape**
  - 2023 saw twice as many software supply chain attacks as 2019-2022 combined [2].
  - **Canonical attacks:** SUNBURST, XZ-Utils
    - Sophisticated approaches
  - **“Simpler” attacks:** npm event-stream
    - Maintainer change



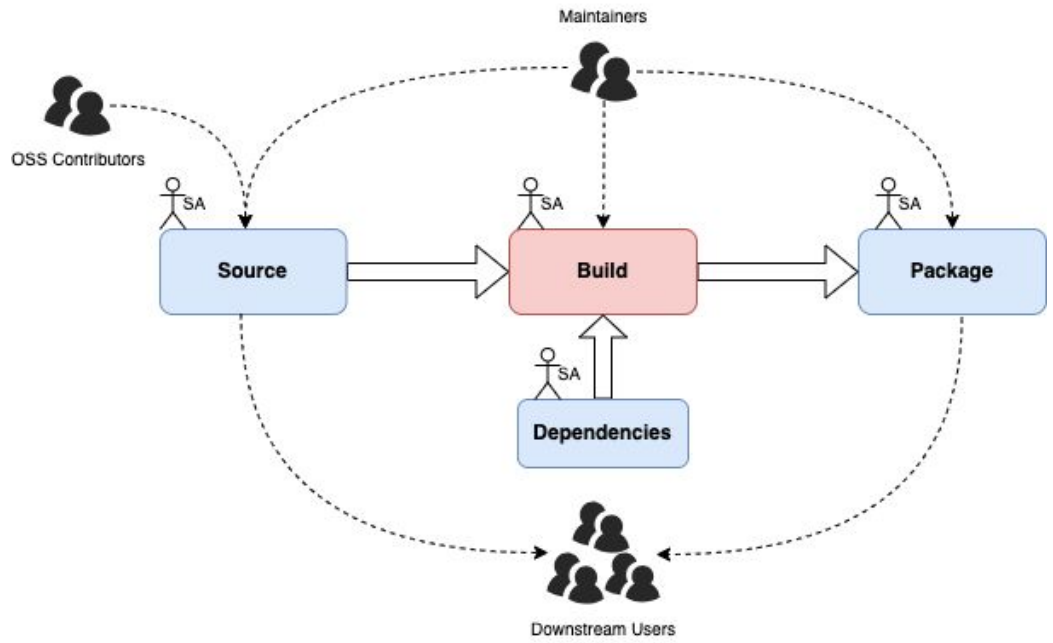
[1] ENISA Threat Landscape for Supply Chain Attacks. Technical Report. July 2021.
[2] Sonatype 9th Annual State of the Software Supply Chain. October 2023.


Maintainers
OSS Contributors
SA
SA
SA
Source
Build
Package
SA
Dependencies
Downstream Users

Problem Space

- Complex software supply chain: **Opaque build process**
  - Binaries whose origins are difficult to determine
  - Compile same source twice

    → **get two different results!**



- **Are we executing the correct code?**
  - Are the source-to–binary semantics are correct?
  - How can we detect vulnerabilities introduced
    - in the build?
    - during distribution?

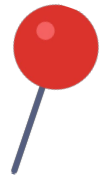
- **If the same sources always compile to different binaries, how can we know if it is correct?**

Reproducible Builds (R-Bs)

Definition

*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 [3].*
>
> *f(x) = y*

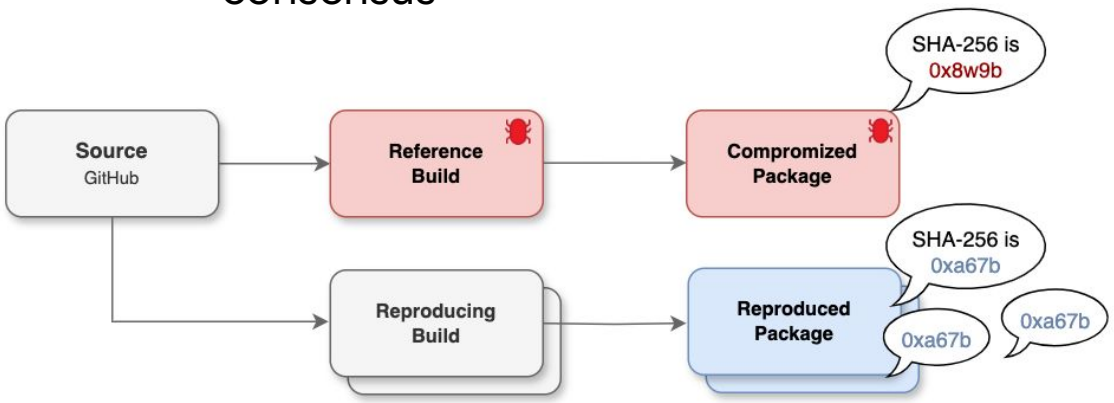


- **Motivation:** Strengthens confidence in source-to-binary correctness
- **Core assumption:** All verifiers' toolchains cannot be subverted
- **Limitation:** Does not claim absence of vulnerabilities in code

[3] https://reproducible-builds.org/docs/definition/

Leveraging R-Bs

- “Strengthens confidence in source-to-binary correctness”
- **How?**
  - Build & publish hash digest
  - Independent verifiers → distributed consensus



State of R-Bs

- Most software today is not reproducible [4]
- Research focuses on Linux OS/packages
- Driver: Reproducible builds project [3]

[4] Fourné, M., et al. (2023, May). It's like flossing your teeth: On the Importance and Challenges of Reproducible Builds for Software Supply Chain Security.

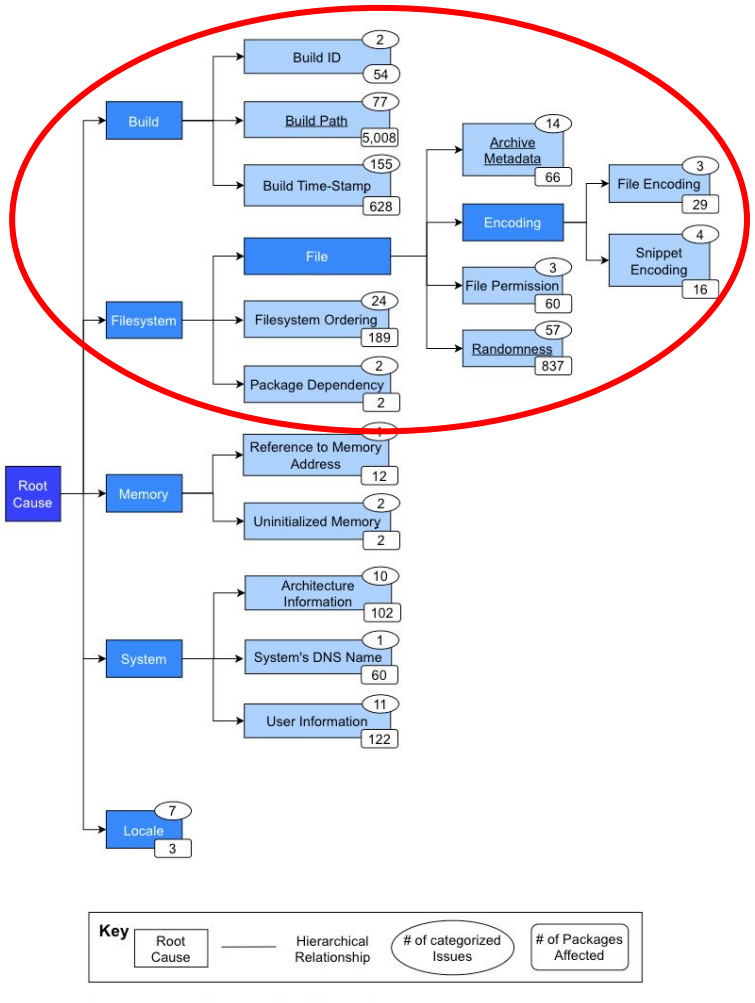
Why is software not reproducible?

Causes of Unreproducibility [5] [6]

- Even when specifying all inputs, the build may be unreproducible

- **Build**
  - Build ID
    - Embedding host unique data
  - Build path
    - Absolute paths to executables
  - Time-Stamp
    - C preprocessor macros e.g. `__DATE__`

- **Filesystem**
  - Archive metadata
    - During build - embeds metadata
  - Filesystem Ordering
    - Auto generation of makefiles
  - Randomness
    - Parallel builds



[5] Lamb, Zacchiroli. "Reproducible Builds: Increasing the Integrity of Software Supply Chains" (Apr. 2021).
[6] Bajaj, et al.. "Unreproducible builds: time to fix, causes, and correlation with external ecosystem factors". Empirical Softw. Engg. 29.1 (Nov. 2023).

For reproducible builds, we need to remove all cases of unreproducibility

R-Bs in Practice: Go Ethereum

- **Motivation**
  - Majority execution client for Ethereum → exploits can have large effect
  - Supply chain security of blockchain software remains relatively unexplored [6]

- **Identifying & rectifying cases of unreproducibility**
  - Compile in container
  - Diff two binaries
    - Tool: `diffoscope`
  - Identify root cause
  - Fix

[6] Soto-Valero, César, Monperrus, Martin, and Baudry, Benoit. "The Multibillion Dollar Software Supply Chain of Ethereum" IEEE C. (2022)

R-Bs in Practice:
Identified Cases of Unreproducibility

Unreproducibility: Build IDs

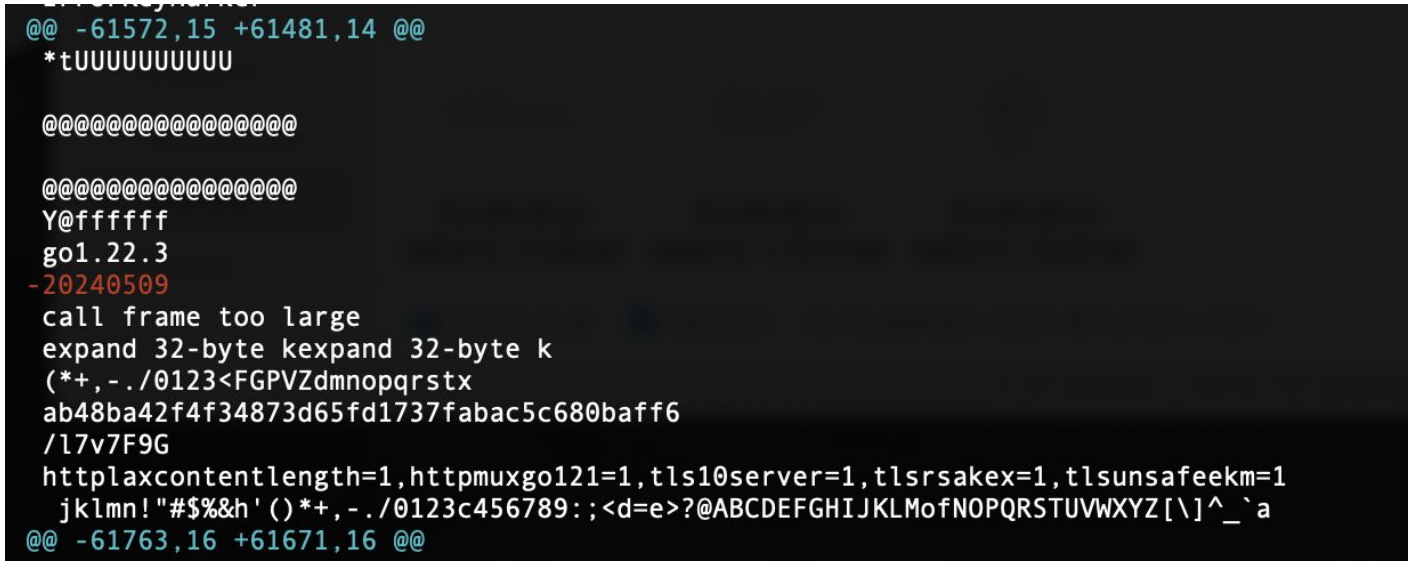
- **Manifestation**
  - GNU Build ID
    - C in Go (CGO)
    - .note.gnu.build-id
  - Go build ID
    - .note.go.buildid

- **Why?**
  - Both IDs claim reproducibility, yet breaks when combined
- **Fix**
  - Remove from analysis

```
--- geth-reference
+++ geth-reproduce
|┄ File has been modified after NT_GNU_BUILD_ID has been applied.
└── readelf --wide --notes {}
 @@ -1,16 +1,16 @@

   Displaying notes found in: .note.gnu.property
     Owner                Data size     Description
     GNU                  0x00000010    NT_GNU_PROPERTY_TYPE_0       Properties: stack size: 0x800000

   Displaying notes found in: .note.gnu.build-id
     Owner                Data size     Description
   -  GNU                 0x00000014    NT_GNU_BUILD_ID (unique build ID bitstring)         Build ID: f40579b453df78b49b822b66e7ac6a8585ca150e
   +  GNU                 0x00000014    NT_GNU_BUILD_ID (unique build ID bitstring)         Build ID: b70d6be12c1b4e795738600a41bd8544cc6bb784

   Displaying notes found in: .note.ABI-tag
     Owner                Data size     Description
     GNU                  0x00000010    NT_GNU_ABI_TAG (ABI version tag)            OS: Linux, ABI: 3.2.0

   Displaying notes found in: .note.go.buildid
     Owner                Data size     Description
   -  Go                  0x00000053    GO BUILDID         description data: 33 6e 35 70 64 72 4c 53 2d 43 69 4c 33 4f 6e 54 56 53 38 5a 2f 2d
70 73 51 37 50 68 58 61 51 71 52 37 46 33 61 6a 2d 6a 31 2f 55 4e 30 74 42 32 78 47 30 37 69 76 48 32 77 5f 39 71 4b 6b 2f 6e 70 67 77 42 67 74
 45 75 51 30 54 65 51 59 48 70 57 51 42
   +  Go                  0x00000053    GO BUILDID         description data: 54 43 33 34 7a 6b 42 72 6c 48 37 44 4d 55 39 66 63 68 35 6e 2f 2d
70 73 51 37 50 68 58 61 51 71 52 37 46 33 61 6a 2d 6a 31 2f 55 4e 30 74 42 32 78 47 30 37 69 76 48 32 77 5f 39 71 4b 6b 2f 6e 66 50 33 44 6b 57
 63 78 33 32 42 71 54 62 31 63 31 39 59
```

Unreproducibility: Date

- **Manifestation**
  - Date embedding
    - Release date! (essentially good)
- **Why?**
  - Source code logic
- **Fix**
  - Check out build in branch…



Unreproducibility: Full Paths & GCC

- **Manifestation**
  - Absolute paths for C libraries
- **Why?**
  - Bug in cmd/go (for ubuntu:bionic)
- **Fix**
  - Fix bug or…
  - Go around problem

- **Manifestation**
  - Differing GCC versions
- **Why?**
  - Bug in Travis CI

geth-reference: go1.22.2

Running `readelf -p .rodata geth-reference | grep /home/travis` …

```
[9e8b70] /home/travis/gopath/pkg/mod/github.com/karalabe/hid@v1.0.1-0.20240306101548-573246063e52/libusb/libusb/libu
[9e8be8] /home/travis/gopath/pkg/mod/github.com/karalabe/hid@v1.0.1-0.20240306101548-573246063e52/libusb/libusb/os/e
[9e8c68] /home/travis/gopath/pkg/mod/github.com/karalabe/hid@v1.0.1-0.20240306101548-573246063e52/libusb/libusb/os/l
[9e8ce8] /home/travis/gopath/pkg/mod/github.com/karalabe/hid@v1.0.1-0.20240306101548-573246063e52/libusb/libusb/os/l
[9e8d60] /home/travis/gopath/pkg/mod/github.com/karalabe/hid@v1.0.1-0.20240306101548-573246063e52/libusb/libusb/core
[9e8e00] /home/travis/gopath/pkg/mod/github.com/karalabe/hid@v1.0.1-0.20240306101548-573246063e52/libusb/libusb/hotp
[9e8e78] /home/travis/gopath/pkg/mod/github.com/karalabe/hid@v1.0.1-0.20240306101548-573246063e52/libusb/libusb/io.c
[9e9db8] /home/travis/gopath/pkg/mod/github.com/ethereum/c-kzg-4844@v1.0.0/bindings/go/../../src/c_kzg_4844.c
```

go/cmd: trimpath does not clean some CGO C filepaths #67011

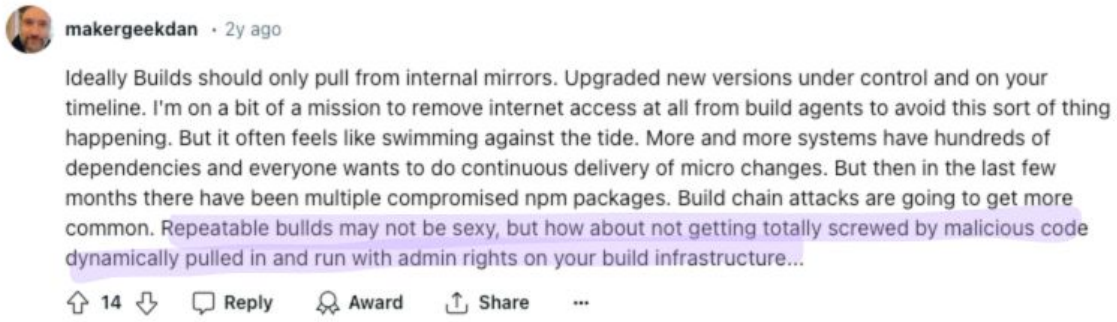
Open **vivi365** opened this issue last month · 5 comments

R-Bs in Practice: Summary

- Manual work (At least initially)
- Your favourite language is now assembly
- Compiling all day is slow
- Reproducible builds **make sense**
- Great bug hunter

Current Challenges [4]

- Unclear definitions of reproducibility
- Manual process, somewhat ad-hoc
- Reproducibility for upstream dependencies
- Some toolchains are extra problematic



Future

- "Make R-Bs the new standard, so new releases are **reproducible by default.**" [4]
- "Treat unreproducible builds as a **security threat**." [4]

- Related work
  - SLSA
  - SBOMs/Attestations
  - CI/CD efforts
  - (...)

[4] Fourné, M., et al. (2023, May). It's like flossing your teeth: On the Importance and Challenges of Reproducible Builds for Software Supply Chain Security.

Thank You!

reproducible-builds.org/

chains.proj.kth.se/

linkedin.com/in/viviandersson/