securing the software supply chain of infra management tools

Config Management Camp

7th feb. 2023

Alexis Mousset

`whoami`

- sysadmin background
- lead system developer **@rudder**
- secure code working group **@rust-lang**
  - vulnerabilities database for Rust libraries
  - security-related tooling

infra management software

- runs everywhere
  - whether with an agent or remote connections
- high privileges
- often acts as glue
  - cross technologies to adapt to what we configure

infra management software

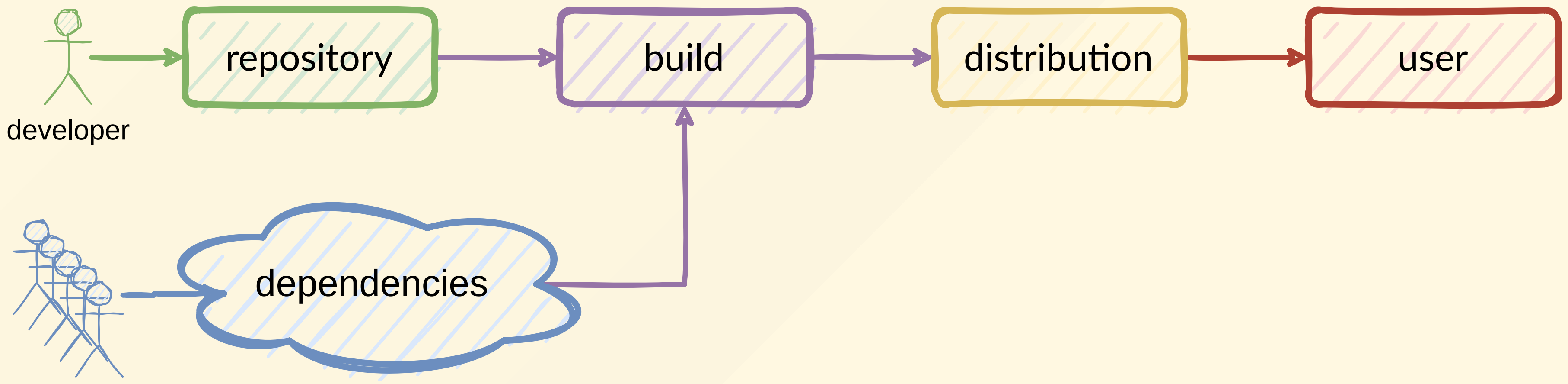
- complex software
  - other remote admin access are simpler (`openssh` , etc.)
  - highly connected to other infra parts
  - big attack surface
  - dependencies

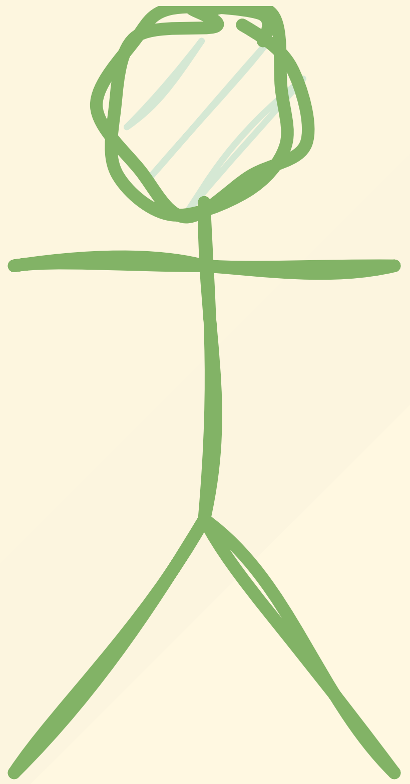
infra management software

- this makes these software targets of attacks
- classic vulnerabilities
    - exploitation of a bug in the program
    - authentication bypass
    - etc.
- we are *not* talking about these

where does infra software come from?

software supply chain

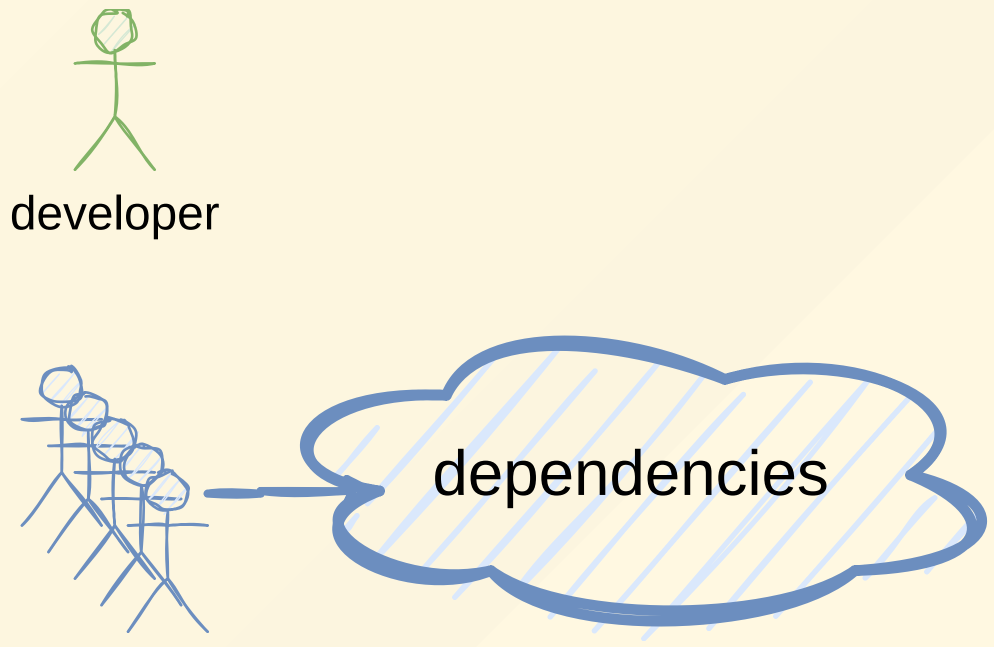




developer

developer

- working on the project/for the company
- a workstation
- various credentials
    - recent **Circle CI** breach
- out of scope here, but needs special attention


developer
dependencies

dependencies

- open-source building blocks are now *everywhere*
- various ecosystems

other developers

(a lot)

who has (indirect) push rights to software?

- every one that has push and release access to all your dependencies
- you can't audit all dependencies
  - can only be a heuristic or a community effort
- more and more package managers and dependencies sources
  - less reliant on system dependencies

estimates on Rudder

- Rust
  - `cargo supply-chain` allows visualizing the dependencies maintainers
  - Our node/server communication daemon lists:
    - 140 individuals
    - 34 Github teams

attacks/vulnerabilities on dependencies

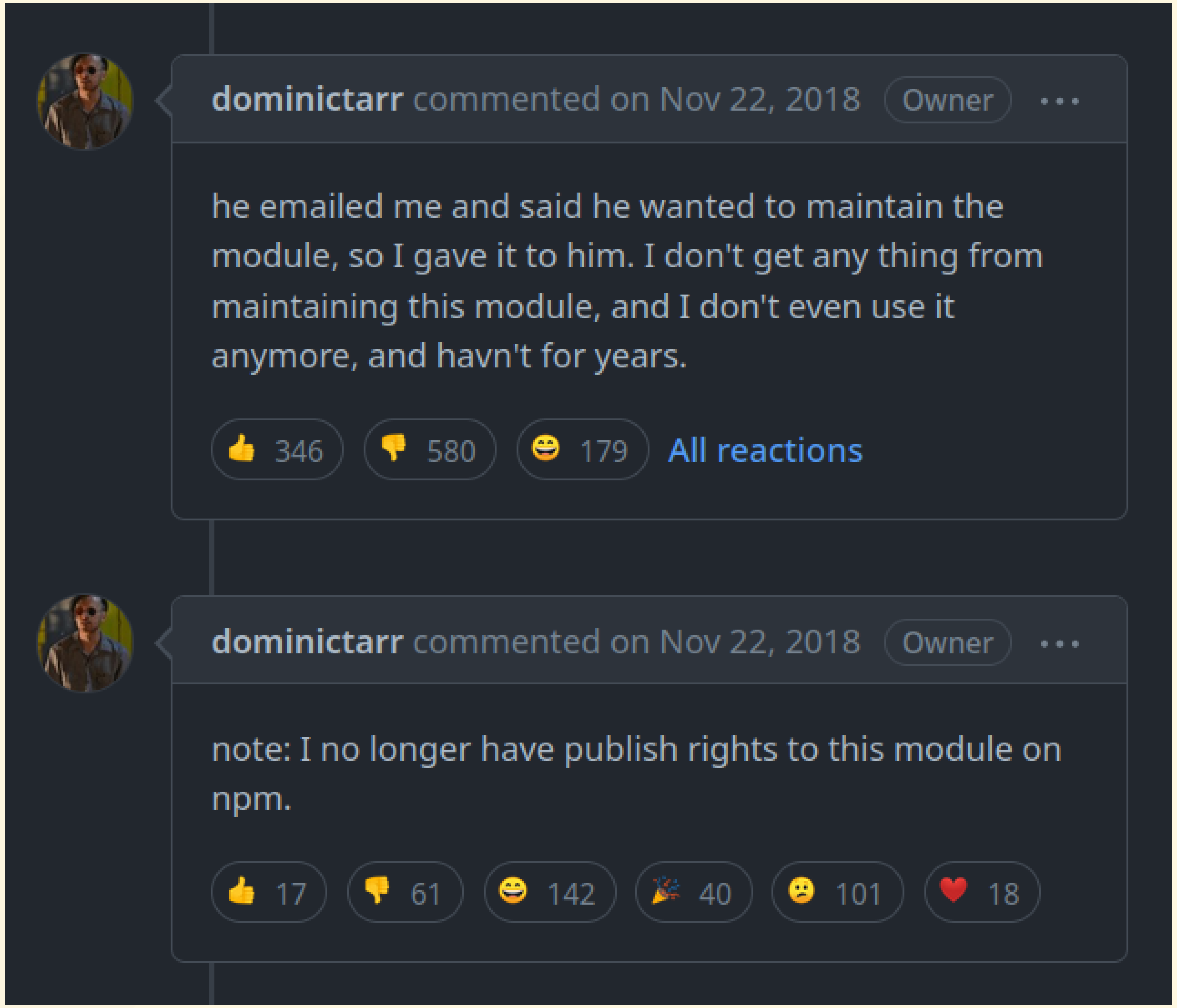
- increasing in the latest years
- huge potential

you may have heard of...

- *log4shell*
  - RCE in log4j, a popular Java logging library
  - revealed that nobody really knows what they are running
- *openssl*

how hard can it be?

- `event-stream`, popular npm package (1.2k stars on github)
- release including code to steal crypto ledgers on dev machines



dominictarr commented on Nov 22, 2018 Owner

he emailed me and said he wanted to maintain the module, so I gave it to him. I don't get any thing from maintaining this module, and I don't even use it anymore, and havn't for years.

👍 346 👎 580 😄 179 All reactions

dominictarr commented on Nov 22, 2018 Owner

note: I no longer have publish rights to this module on npm.

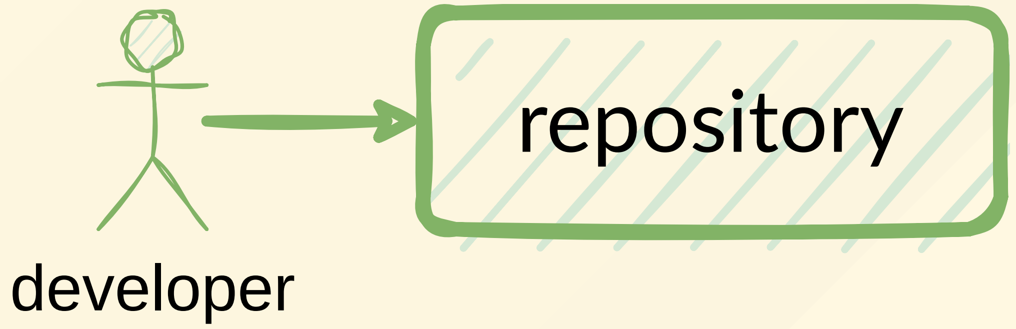
👍 17 👎 61 😄 142 🎉 40 😕 101 ❤️ 18

Rust side

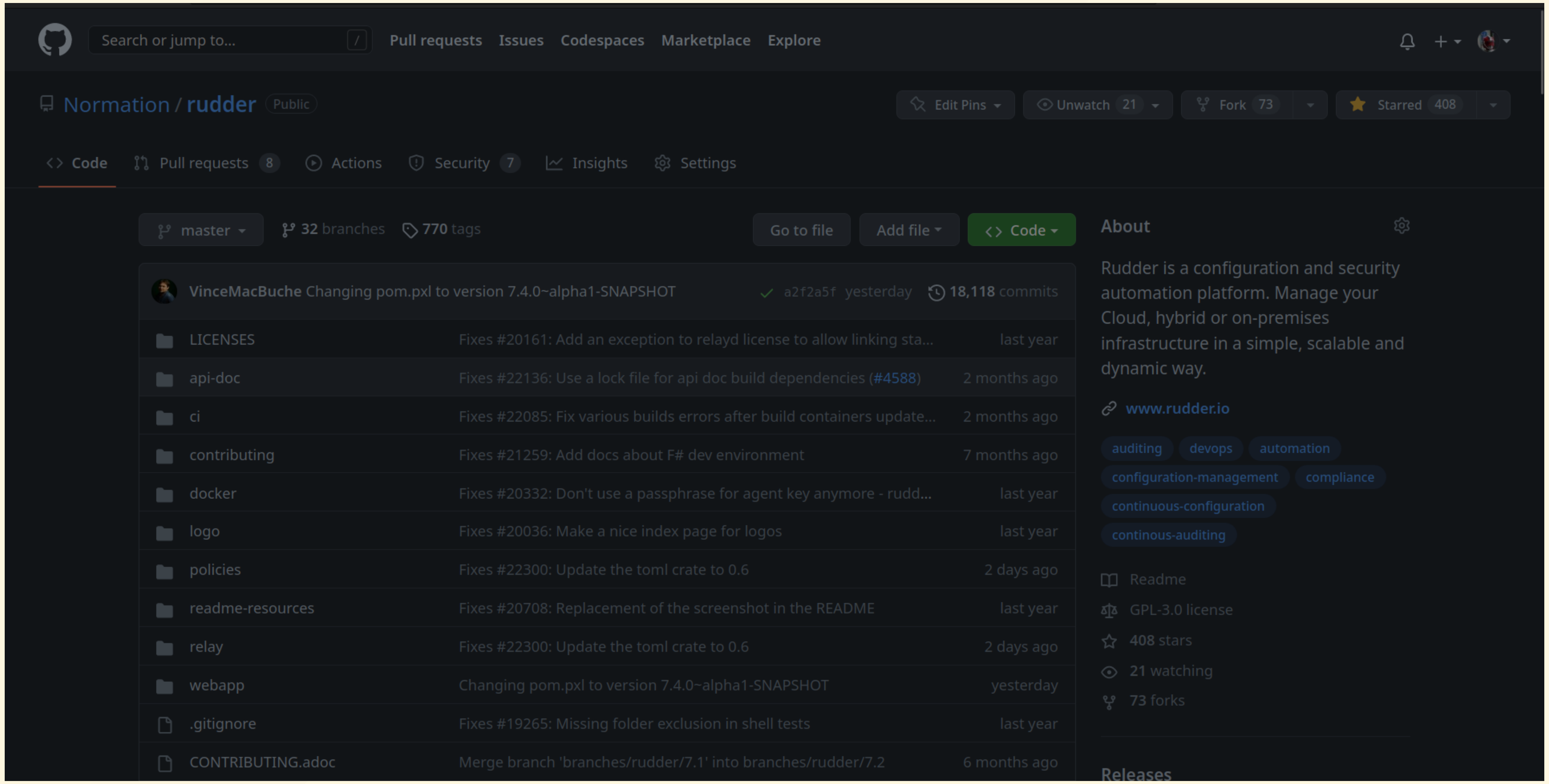
- various attacks on crates.io
  - typosquatting `rustdecimal` instead of `rust_decimal`
  - attack against Gitlab CI

what do we learn from this?

- **good**: people are generally nice to each other!
- **bad**: it is basically our only protection


repository
developer

repository



VinceMacBuche Changing pom.pxl to version 7.4.0~alpha1-SNAPSHOT ✓ a2f2a5f yesterday 18,118 commits

LICENSES	Fixes #20161: Add an exception to relayd license to allow linking sta...	last year
api-doc	Fixes #22136: Use a lock file for api doc build dependencies (#4588)	2 months ago
ci	Fixes #22085: Fix various builds errors after build containers update...	2 months ago
contributing	Fixes #21259: Add docs about F# dev environment	7 months ago
docker	Fixes #20332: Don't use a passphrase for agent key anymore - rudd...	last year
logo	Fixes #20036: Make a nice index page for logos	last year
policies	Fixes #22300: Update the toml crate to 0.6	2 days ago
readme-resources	Fixes #20708: Replacement of the screenshot in the README	last year
relay	Fixes #22300: Update the toml crate to 0.6	2 days ago
webapp	Changing pom.pxl to version 7.4.0~alpha1-SNAPSHOT	yesterday
.gitignore	Fixes #19265: Missing folder exclusion in shell tests	last year
CONTRIBUTING.adoc	Merge branch 'branches/rudder/7.1' into branches/rudder/7.2	6 months ago

About

Rudder is a configuration and security automation platform. Manage your Cloud, hybrid or on-premises infrastructure in a simple, scalable and dynamic way.

www.rudder.io

auditing devops automation configuration-management compliance continuous-configuration continous-auditing

Readme
GPL-3.0 license
408 stars
21 watching
73 forks

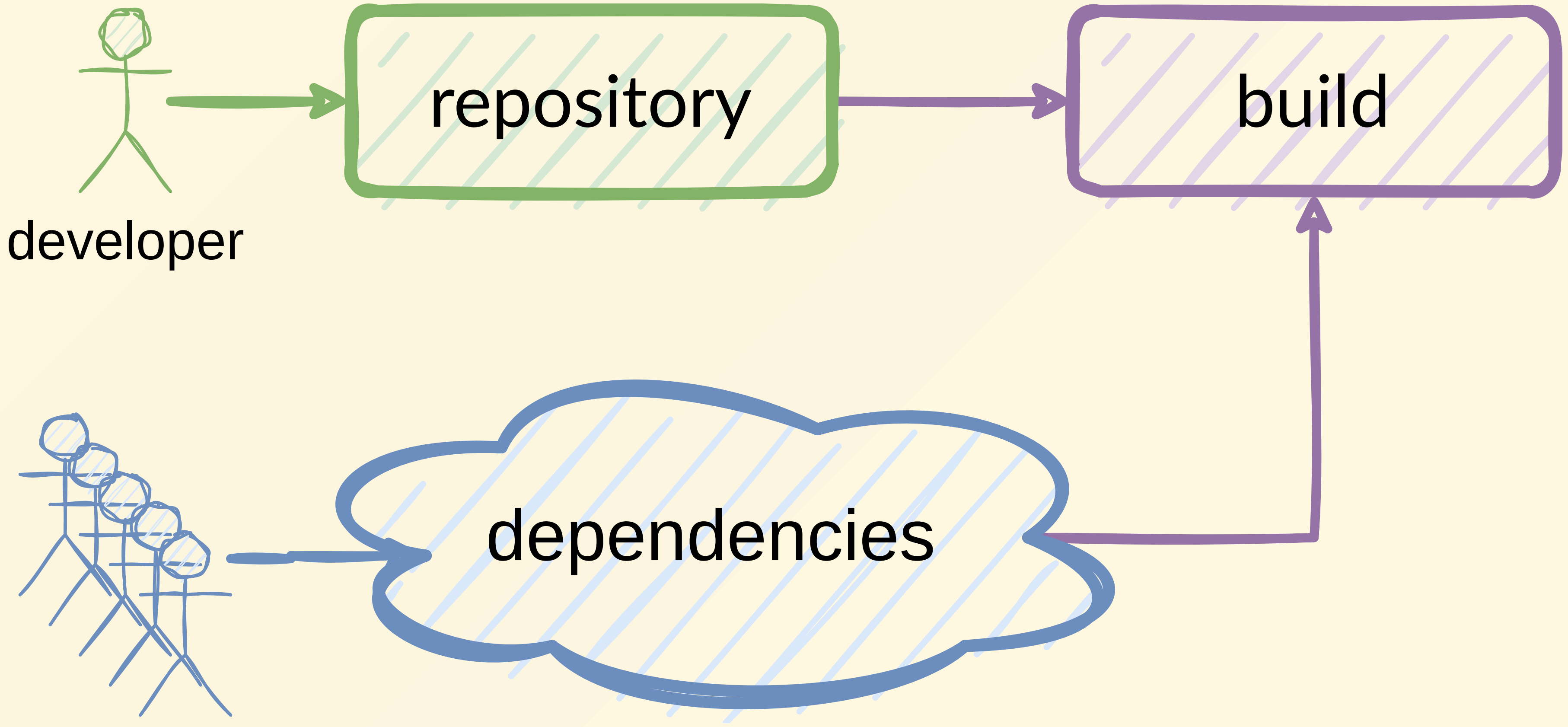
Releases

repository

- not the easiest channel
- still a lot of deploy keys/SSH keys without passwords in the wild

repository

- reviews
- protected branches
    - to force a review and make changes visible


developer
repository
build
dependencies

build process & infra

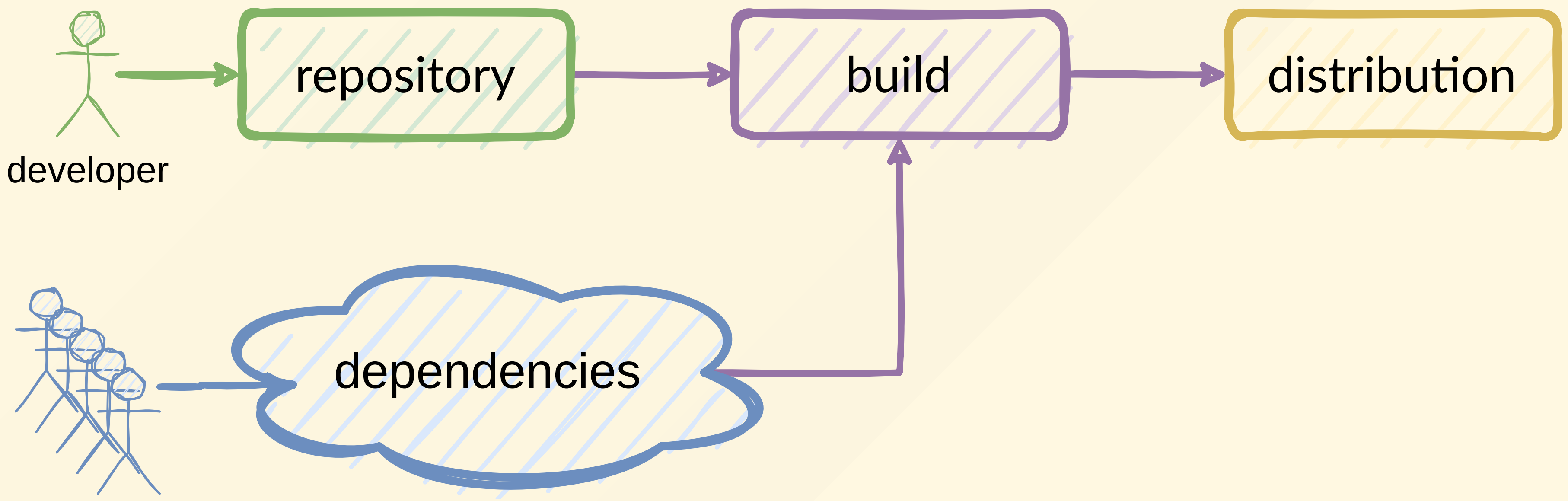
- setup a build environment
  - containers, VM, etc.
  - either SaaS or hosted
- download all sources
  - our code
  - dependencies from various channels
- build
- push artifacts

build process & infra

- SolarWinds
  - Monitoring software *Orion* infected with malware
  - attack through the build platform
  - installed on persistent builder systems
  - modified the sources at build time, hard to detect
- attacks on CI platforms
  - circleCI
  - Gitlab CI

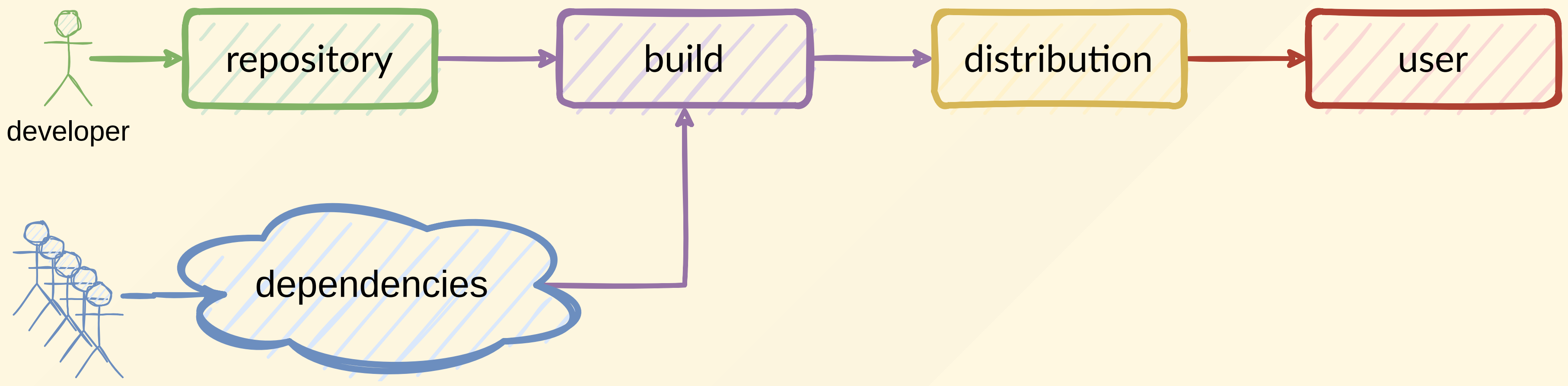
build process & infra

- build environments are critical assets
- security monitoring and update policies
- for sources
  - lock files (i.e. include the dependency' source hash in the repository)
  - signatures check


developer
repository
build
distribution
dependencies

distribution

- generally correctly done!
    - signatures (rpm, dpkg, msi, etc.)


developer
repository
build
distribution
user
dependencies

what do users need?

- visibility
- trust (integrity)

how to reach these goals?

aside: OpenSSF

- *Open Source Security Foundation*
- affiliated with the Linux Foundation
- created in August 2022
- merges several previous efforts

visibility

identifying software

- the first problem with visibility is the ability to identify software.
- we are used to "CPE", used in CVEs

- It is not enough
- SWID and purl

purl

- uniform identifier for software
- good for upstream stuff

```
pkg:deb/debian/curl@7.50.3-1?arch=i386&distro=jessie
pkg:docker/cassandra@sha256:244fd47e07d1004f0aed9c
pkg:gem/ruby-advisory-db-check@0.12.4
pkg:github/package-url/purl-spec@244fd47e07d1004f0aed9c
pkg:golang/google.golang.org/genproto#googleapis/api/annotations
```

SWID

- better for downstream
- NIST/SCAP
- usable in CVEs

```xml
<SoftwareIdentity
  xmlns="http://standards.iso.org/iso/19770/-2/2015/schema.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://standards.iso.org/iso/19770/-2/2015/schema.xsd"
  xml:lang="en-US"
  name="Red Hat Enterprise Linux"
  tagId="com.redhat.RHEL-8-x86_64"
  tagVersion="1"
  version="8"
  versionScheme="multipartnumeric"
  media="(OS:linux)">
```

how to list software?

- *Software Bill of Materials*
- list of ingredients (components and versions)

SPDX

- first open-source oriented SBOM
- started around 2010
- focused on license compliance initially
    - included standardized license identifiers
    - headers

CycloneDX

- from OWASP, in 2017
- security-oriented
- goes beyond SBOM
  - HBOM (hardware), OBOM (operations), etc.
- vulnerability management: VDR, VEX

vulnerability tracking

- CVE historically

OSV

- *Open Source Vulnerability*
- CVE is not enough for everything
  - software badly identified
  - often useless scoring
- a format spec
- a database centralizing information from different ecosystems

vulnerability tracking at ecosystem level

- a database for each language
- Github efforts
  - security tooling
  - dependabot

integrity

- source, build and artifact
- signing distributed binaries is good, and already well deployed
- ...but absolutely not enough!

sigstore

- tooling to sign and check signatures of artifacts
- **Attend next talk for more details!**

what can we do?

- we started hearing about these topics ten years ago
- only starting to actually *exist* now

what can we do?

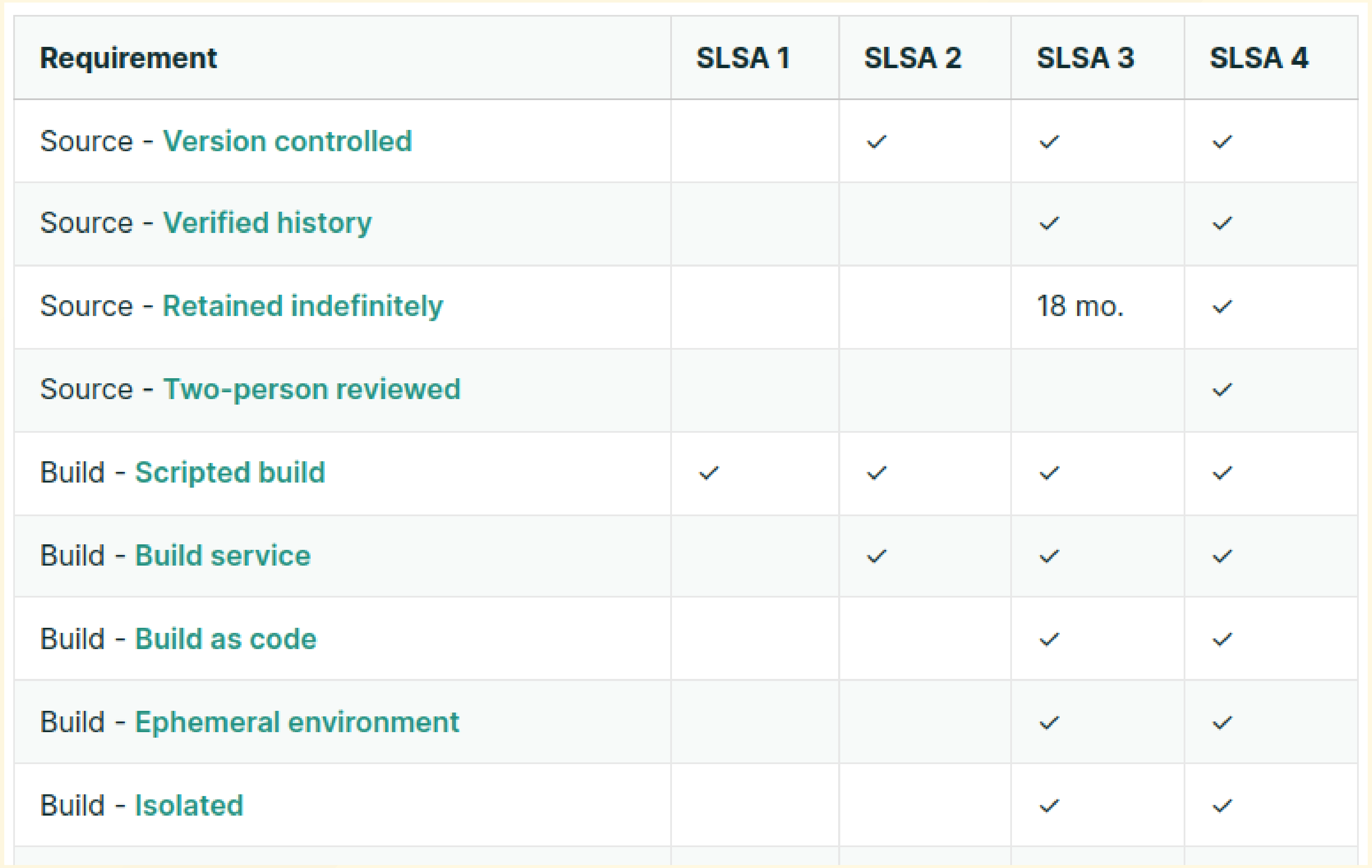
- the problem space is huge
- the cost is potentially huge
- we need to prioritize and focus

SLSA

- pronounced "salsa"
- *Supply chain Levels for Software Artifacts*
- originally from Google, now under the OpenSSF umbrella
- framework providing checklists with levels

SLSA

- the goal is to help list and prioritize
- not transitive



Requirement	SLSA 1	SLSA 2	SLSA 3	SLSA 4
Source - Version controlled		✓	✓	✓
Source - Verified history			✓	✓
Source - Retained indefinitely			18 mo.	✓
Source - Two-person reviewed				✓
Build - Scripted build	✓	✓	✓	✓
Build - Build service		✓	✓	✓
Build - Build as code			✓	✓
Build - Ephemeral environment			✓	✓
Build - Isolated			✓	✓

SLSA level 1

- "The build process must be fully scripted/automated and generate provenance."
- visibility but no integrity
- allow the end user to make risk-based security decisions
- no protection against tampering

SLSA level 2

- "Requires using version control and a hosted build service that generates authenticated provenance."

SLSA level 3

- "The source and build platforms meet specific standards to guarantee the auditability of the source and the integrity of the provenance respectively."
- auditors certify that platforms meet the requirements

SLSA level 4

- "Requires two-person review of all changes and a hermetic, reproducible build process."

where are we at?

rudder

- A **lot** of ecosystems
  - Scala/Java (maven-based)
  - Elm (dedicated tooling)
  - Rust (cargo/crates.io-based)
  - F# (dotnet/nuget-based)
  - JavaScript (npm-based)
  - C
  - Perl (cpan-based)
  - Python (pip-based)

rudder

- visibility
  - dependency management
  - SBOM?
  - vulnerability scanning
- integrity
  - only at distribution level

rudder

- build security and reproducibility improvements
- next step: aggregated SBOM
- continue making the build more deterministic and hermetic

rust

- vulnerability tracking: okayish
- SBOM: early days
- storing SBOM in binaries: `cargo-auditable`
- still a lot to do on crates.io
  - 2FA, sigstore, etc.
- exploring trust: `cargo-crev`, `cargo-vet`

conclusion

- mostly driven by enterprise & government needs
  - might lead to complex solutions
  - **far too many acronyms** (i've spared you a lot of them)
- the supply chain security ecosystem is still quite immature
  - competing norms, technologies, etc.
  - continuous changes

conclusion

- but we can't ignore it, at all levels
  - open source ecosystems
  - software editors
  - end users, especially in critical contexts
- *we are all software editors*

references

- Open Source Security Foundation (OpenSSF, Linux Foundation)
  - SLSA
  - OSV
  - sigstore
- OWASP Foundation
  - CycloneDX
- PBOM.dev
  - OSC&R: Open Software Supply Chain Attack Reference

references

- Chainguard
- Aqua Security
  - open-source tooling: Trivy
- Anchore
  - Grype, Sift
- OmniBOR
  - Artifact Dependency Graph

questions?

- amo@rudder.io
- twitter.com/AlexisMousset
- mastodon.social/@amousset