SBOMs Aren't Enough. Secure Your Software Supply Chain End-To-End

Yongjae Chung
M.S. New York University

Justin Cappos
Professor, New York University

NYU TANDON SCHOOL OF ENGINEERING

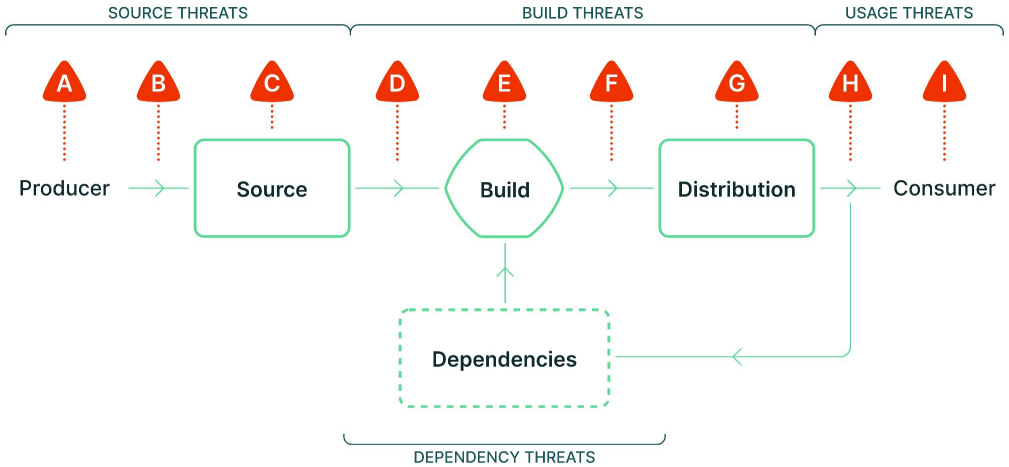
Bio: @yongjae

- Graduated NYU, soon to be Ph.D. Student @ Purdue Univ.
- Contributor of gittuf

Bio: Justin Cappos

- NYU Professor
- A creator of TUF, in-toto, gittuf, SBOMit, Uptane, etc.
- Heavy collaborator in Linux Foundation efforts

Software Supply Chain



A Producer (entity)
B Authoring & reviewing
C Source code management
D External build parameters
E Build process
F Artifact publication
G Distribution channel
H Package selection
I Usage

https://slsa.dev/spec/v1.2/threats-overview

Software

Wait!! But I have a SBOM.

A Producer (entity)	D External build parameters	G Distribution channel
B Authoring & reviewing	E Build process	H Package selection
C Source code management	F Artifact publication	I Usage

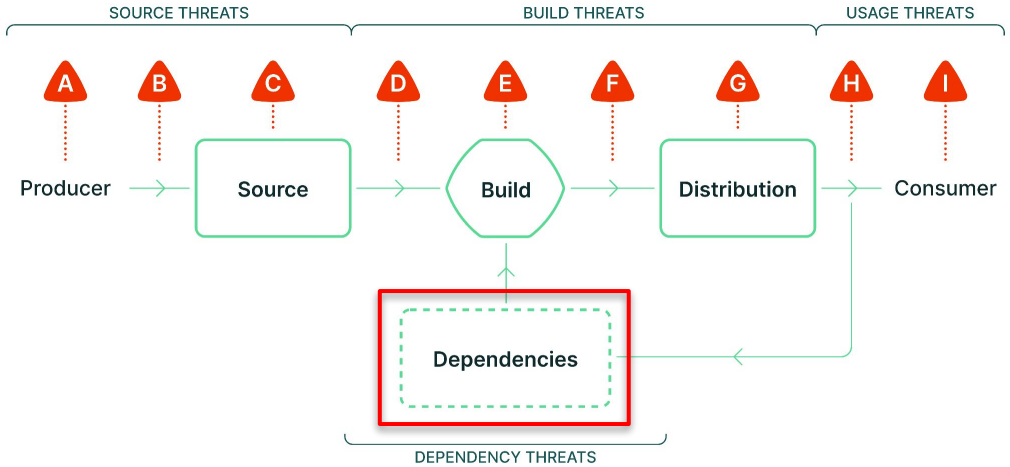
https://slsa.dev/spec/v1.2/threats-overview

Software Bill of Materials (SBOM)

- Often compared to a nutrition label
- An inventory of every component in a piece of software
- Each entry records name, version, supplier, license, and usually a hash or PURL
- Includes transitive dependencies, not just what you declared
- Standard formats: SPDX and CycloneDX

```
{
  "bomFormat": "CycloneDX",
  "specVersion": "1.6",
  "components": [
    {
      "type": "library",
      "name": "log4j-core",
      "version": "2.17.1",
      "purl": "pkg:maven/org.apache.
        logging.log4j/log4j-core@2.17.1",
      "licenses": [{"id": "Apache-2.0"}],
      "hashes": [{"alg": "SHA-256",
        "content": "4d1a3e..."}]
    }
  ]
}
```

What Do SBOMs Do?



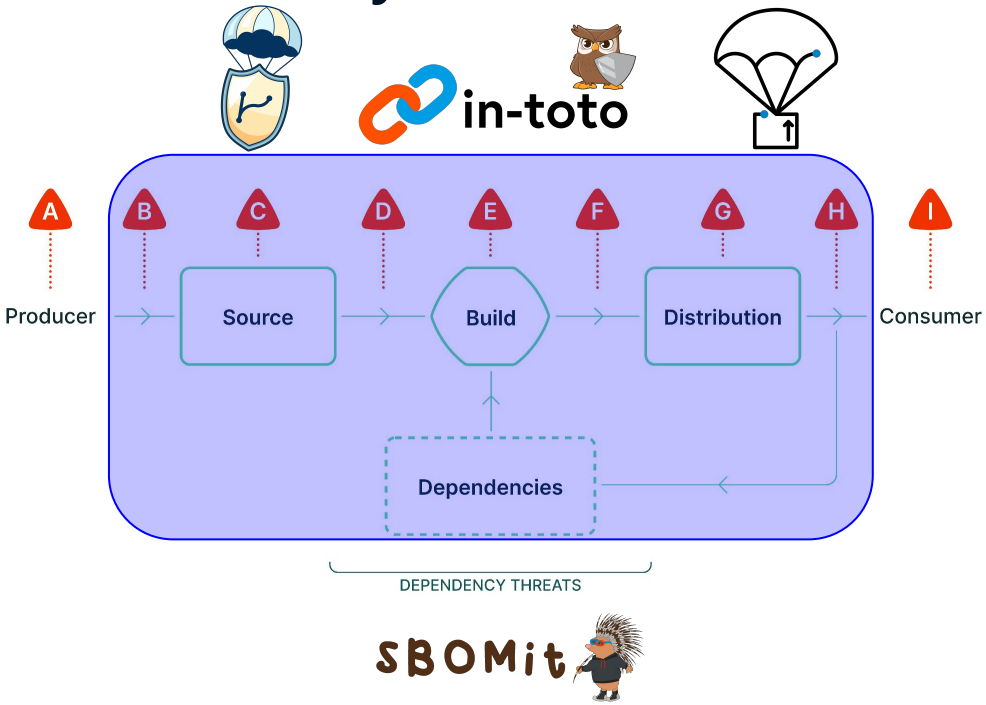
A Producer (entity)
B Authoring & reviewing
C Source code management
D External build parameters
E Build process
F Artifact publication
G Distribution channel
H Package selection
I Usage

What Do

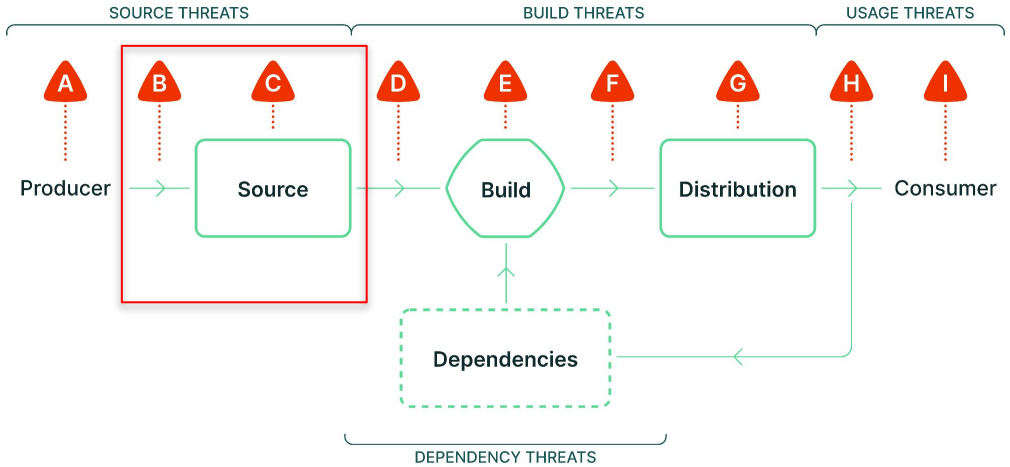
SBOMs are valuable inventory, but they don't protect against anything.

A Producer (entity)
B Authoring & reviewing
C Source code management
D External build parameters
E Build process
F Artifact publication
G Distribution channel
H Package selection
I Usage

What We'll Cover Today



Let's Start from the Top!

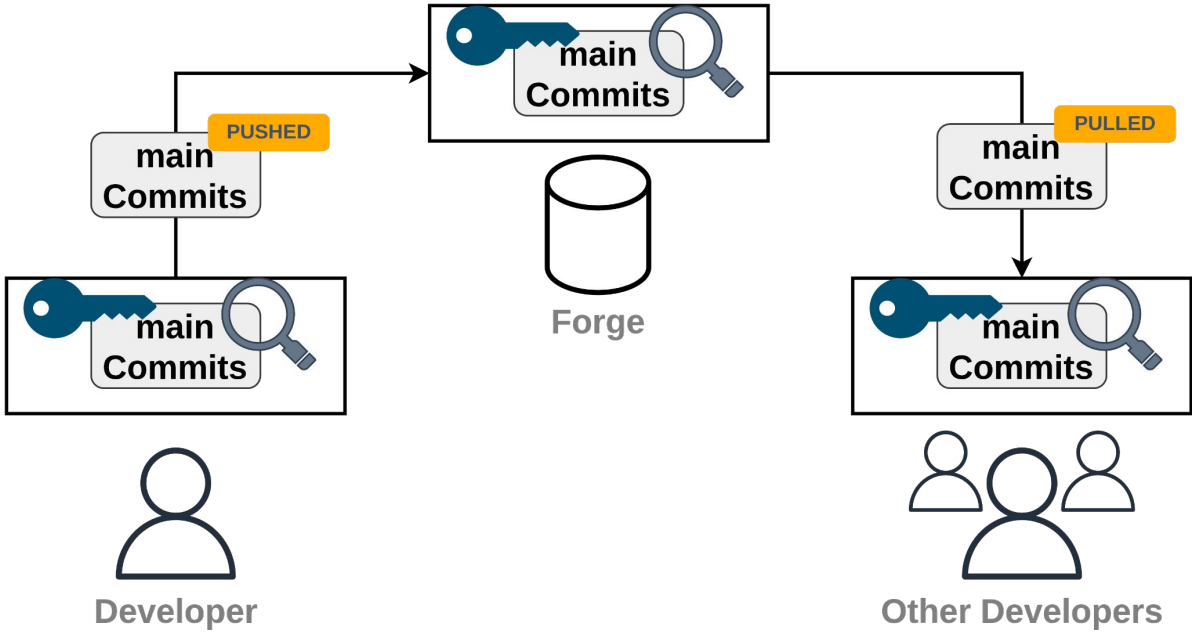


A Producer (entity)
B Authoring & reviewing
C Source code management
D External build parameters
E Build process
F Artifact publication
G Distribution channel
H Package selection
I Usage

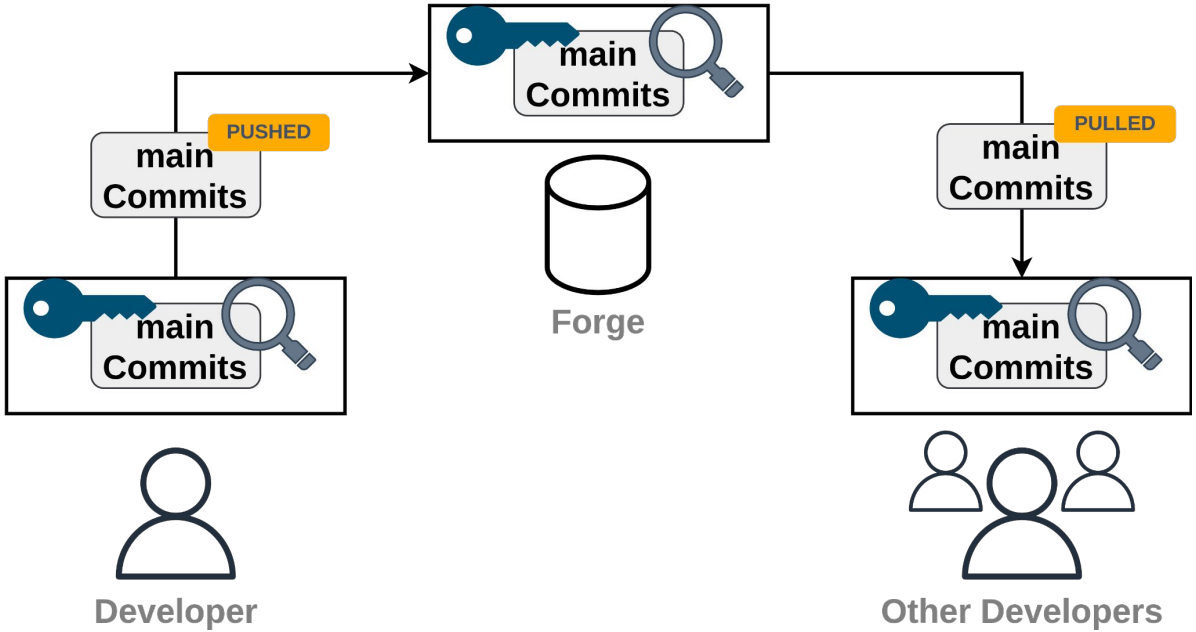
gittuf

A Security Layer for Git Repositories

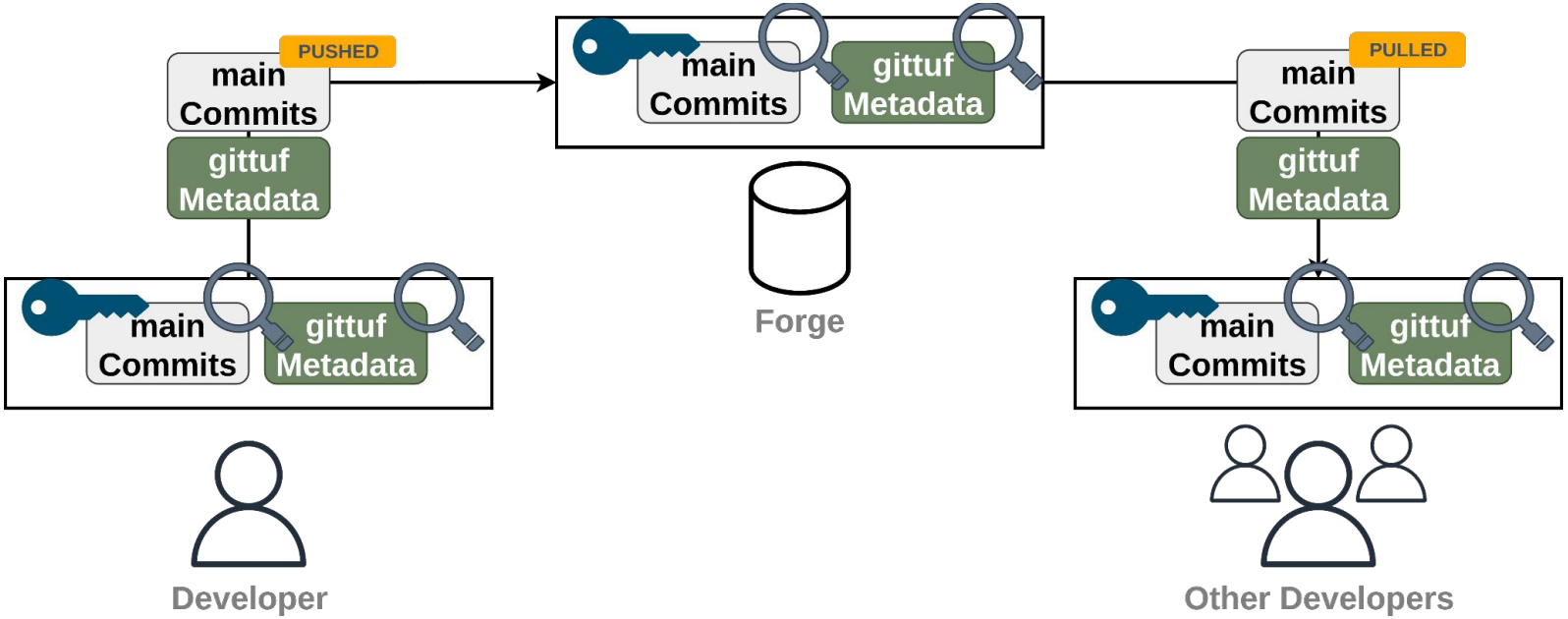
The Problem with Source Code Management



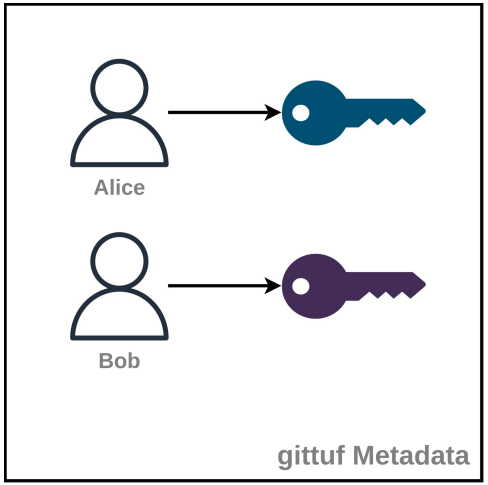
The Problem with Source Code Management



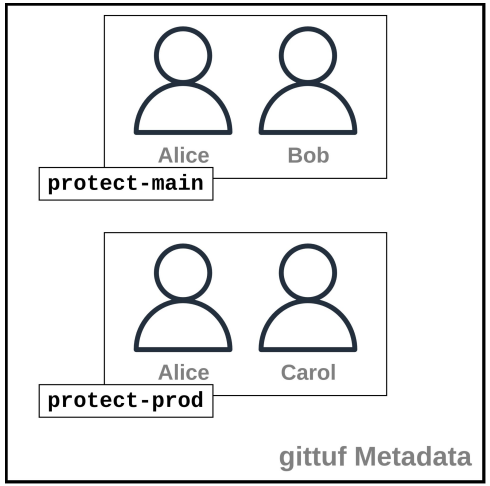
gittuf: In Action



gittuf's Role



Distribute Keys

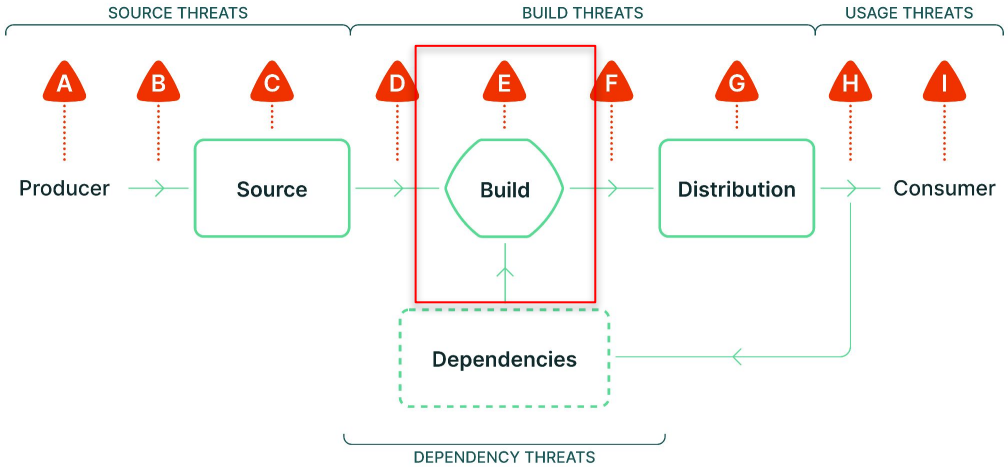


Define Policy

gittuf: Example Policies

- **Branch and Tag Protection:**
  - Only Alice, Bob, and Carol can merge to `main` and create release tags
  - All changes to `main` must be reviewed by at least two of Alice, Bob, and Carol
  - No one can force push to `main`
- **File/folder Protection:**
  - Only the `cryptography` team can make changes to the `signatures` package
  - In our monorepo, each project must be managed by the team responsible for it

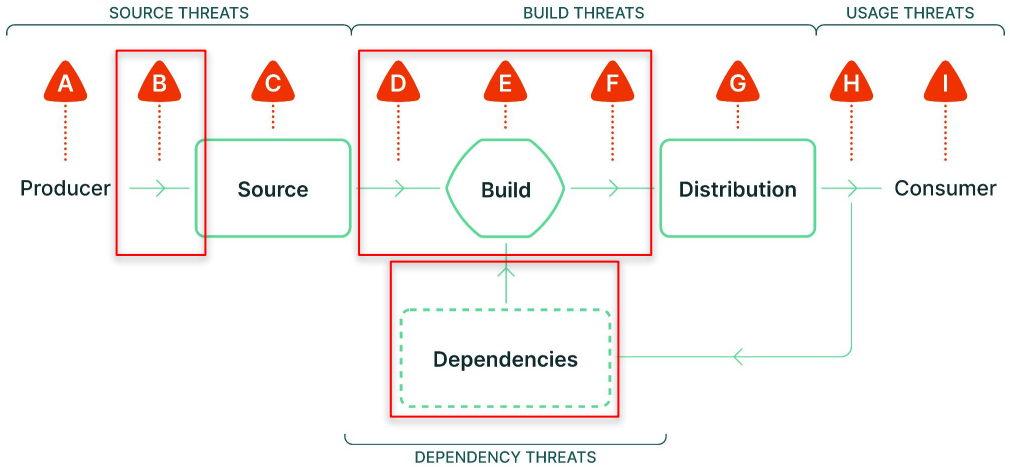
What's next?



A Producer (entity)
B Authoring & reviewing
C Source code management
D External build parameters
E Build process
F Artifact publication
G Distribution channel
H Package selection
I Usage

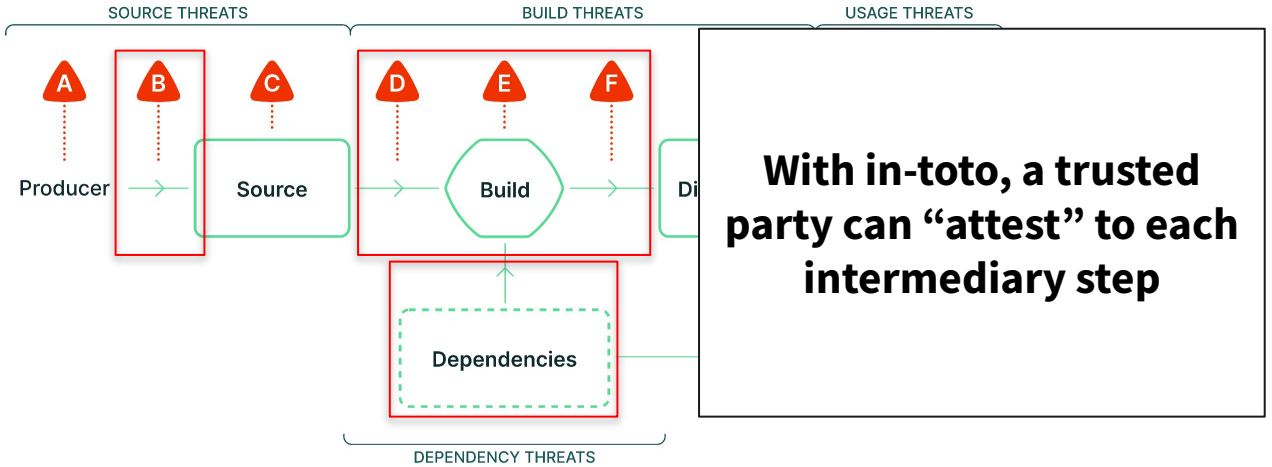
A framework to secure the integrity of software supply chains

What in-toto Can Cover



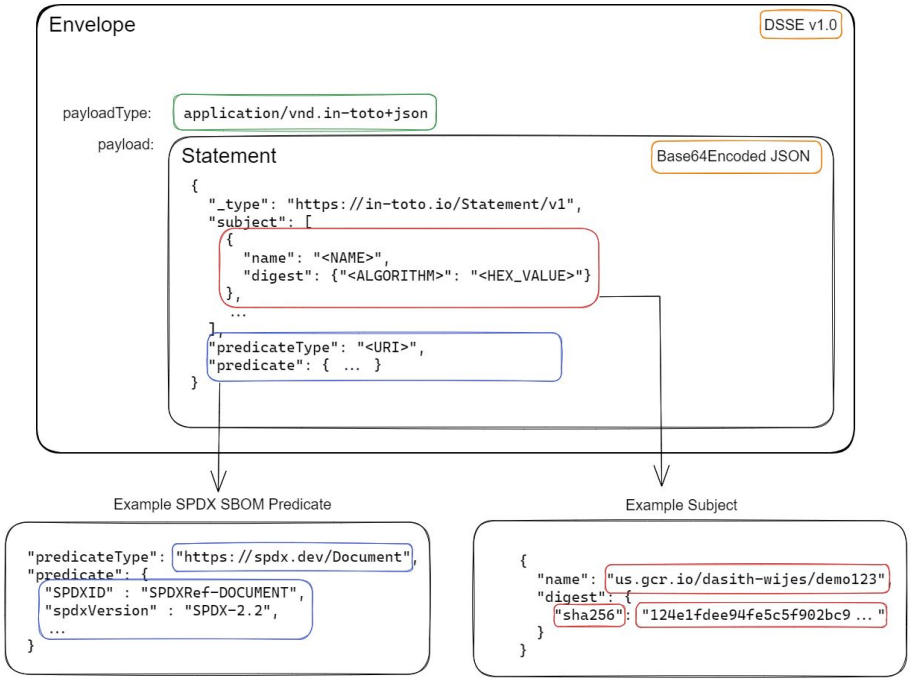
A Producer (entity)
B Authoring & reviewing
C Source code management
D External build parameters
E Build process
F Artifact publication
G Distribution channel
H Package selection
I Usage

What in-toto Can Cover



A Producer (entity)
B Authoring & reviewing
C Source code management
D External build parameters
E Build process
F Artifact publication
G Distribution channel
H Package selection
I Usage

in-toto Attestation Format



Attestation Predicates

- Release: Details an artifact that is part of a given release version.
- Runtime Traces: Captures runtime traces of software supply chain operations.
- SLSA Provenance: Describes how an artifact or set of artifacts was produced.
- SLSA Verification Summary: SLSA verification decision about a software artifact.
- SPDX2 and SPDX3: SPDX-formatted BOM for software artifacts.
- CycloneDX: CycloneDX BOM for software artifacts.
- VULNS: Defines the metadata to share the results of vulnerability scanning on software artifacts.
- … and more!

Attestat

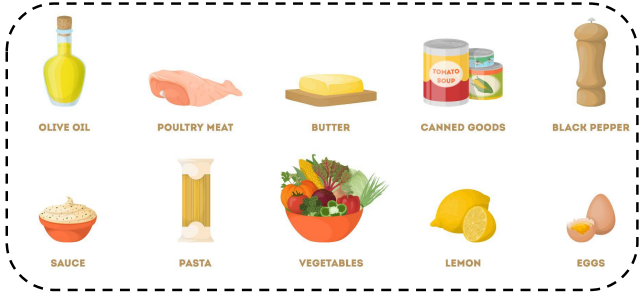
- Release: I
- Runtime T
- SLSA Pro
- SLSA Ver
- SPDX2 ar
- CycloneD
- VULNS: D vare artifacts.
- … and more!

What else can we do with attestations?

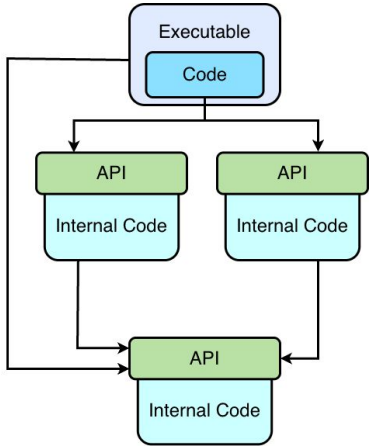
SBOMit

Generate accurate SBOMs directly from in-toto attestations

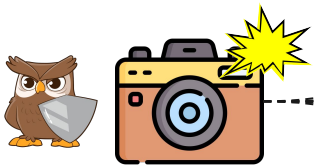
Do You Trust this Pie?

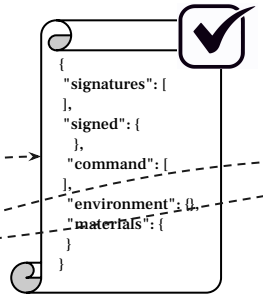


Software

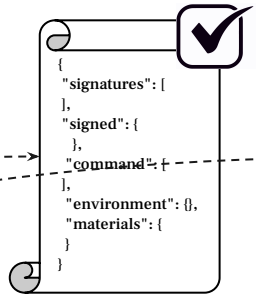


SBOMit = SBOM + in-toto (A kitchen camera 📸 records every step of baking the pie.)

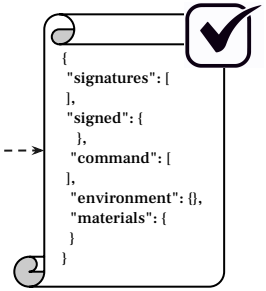




zopfli-0.8.2

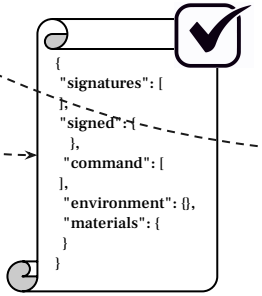


rgb-0.8.50

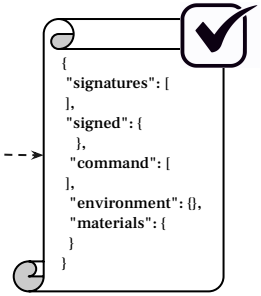


log-0.4.27

👉 “Each attestation is a snapshot of an software component”

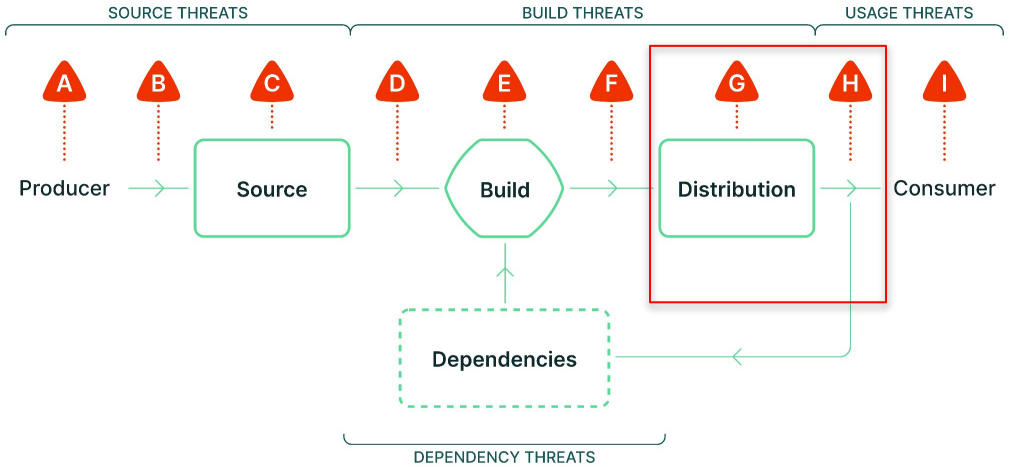


bitvec-1.0.1



indexmap-2.9.0

Secure Package Distribution



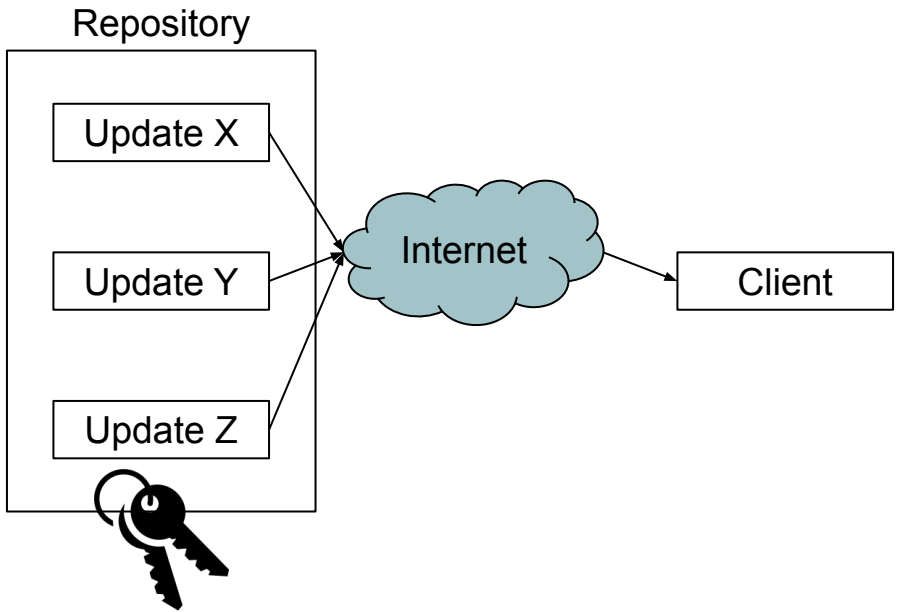
A Producer (entity)
B Authoring & reviewing
C Source code management
D External build parameters
E Build process
F Artifact publication
G Distribution channel
H Package selection
I Usage

TUF (The Update Framework)

A framework for securing software update systems

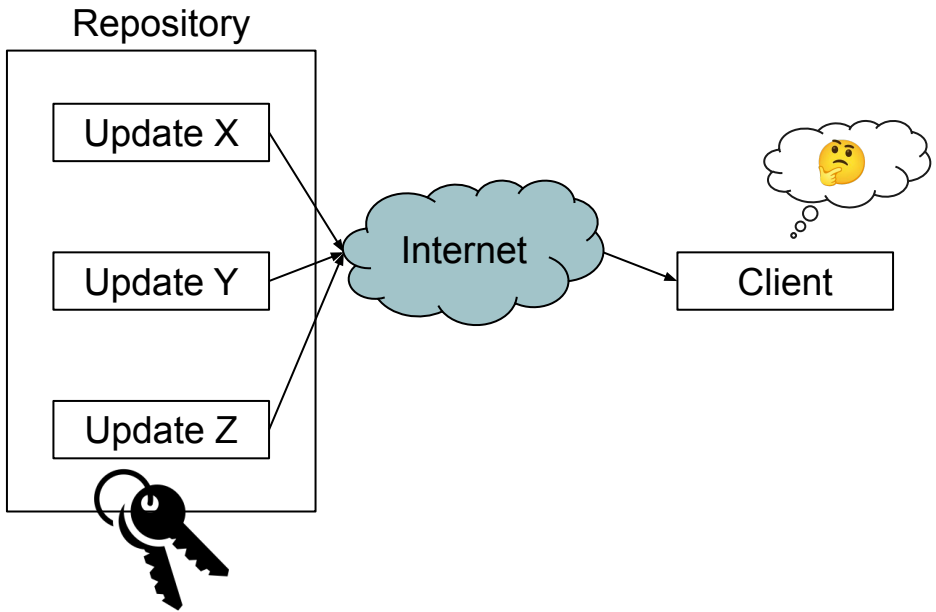
The Build is Secure. What Can Still Go Wrong?

- Package repository or mirror is compromised
- Signing key is stolen
- Client receives an older vulnerable release
- Client downloads a different artifact than intended



What the Client Wants to Know

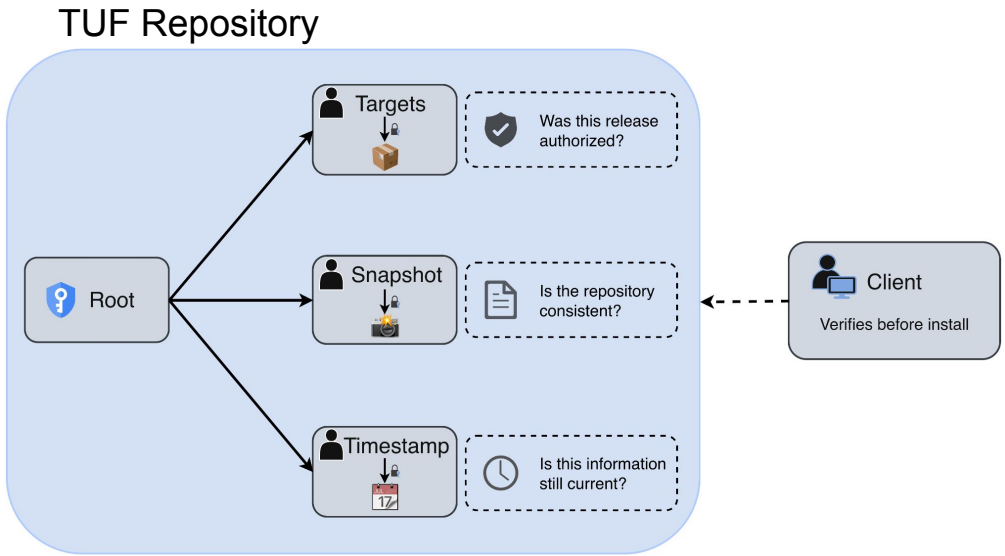
- Was this release authorized?
- Did I receive the authorized bytes?
- Is this information still current?



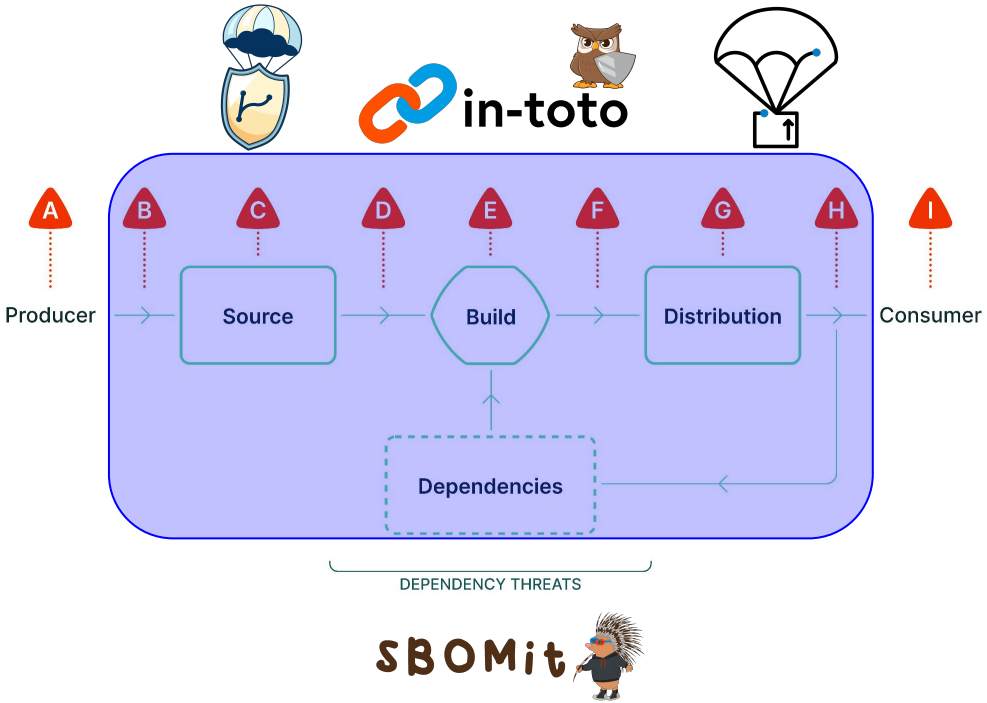
TUF Secures Software Distribution

Responsibility separation to create compromise resilience

Different roles manage different metadata about the released artifact.



Demo



https://github.com/yongjae354/supply-chain-integrity-demo

Getting Started


https://slsa.dev/


https://gittuf.dev/quickstart/


https://in-toto.io/docs/getting-started/


https://witness.dev/docs/


https://sbomit.dev/about/


https://theupdateframework.io/docs/getting-started/

Thank you!