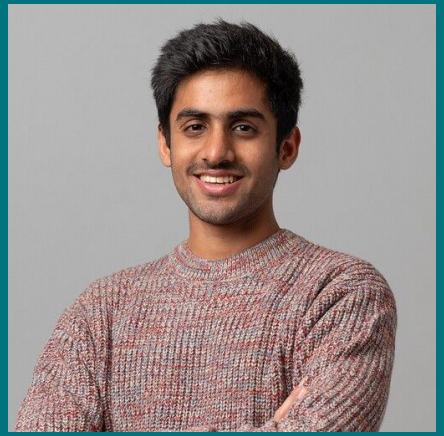
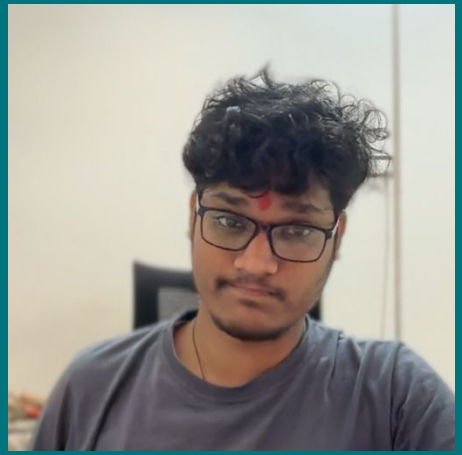
in-toto Attestations for What Really Happens in Your Build Pipeline, with Witness

Vyom Yadav (Canonical)
Rahul Vishwakarma (HighLevel)



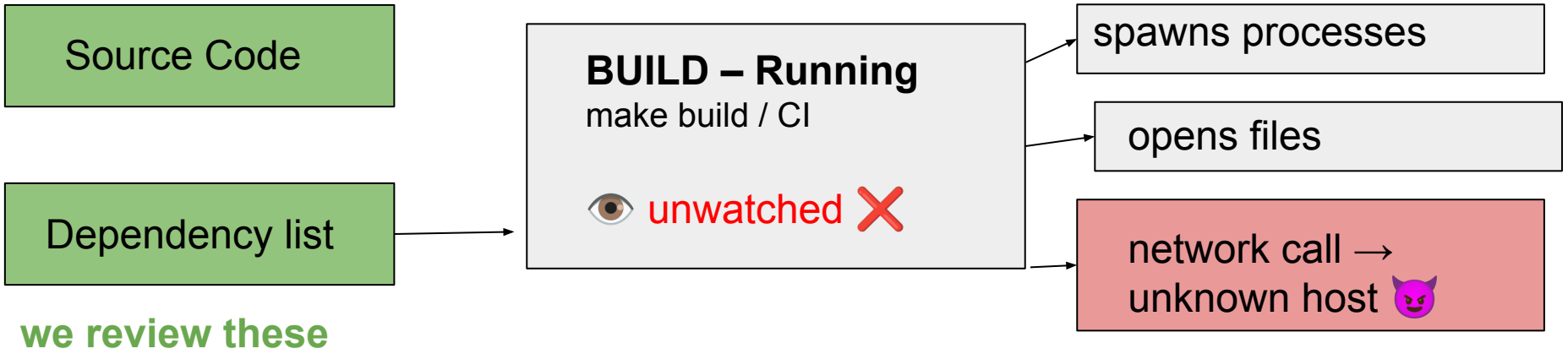
Security Engineer
@Canonical



Engineering Intern
@Highlevel

Do you know what your build actually did?

We inspect the inputs – not the build itself



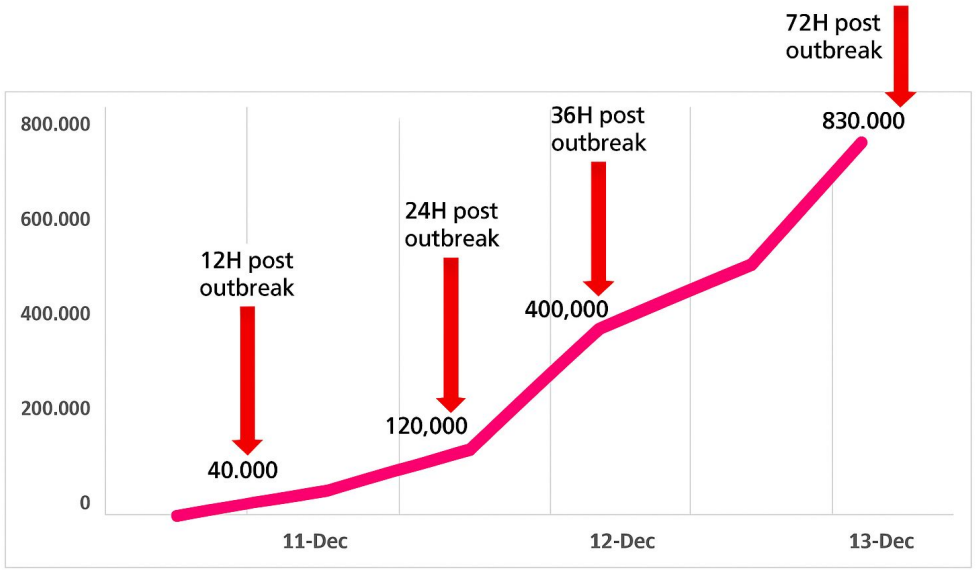
We review the inputs. The build's behaviour, processes, files, network, goes unwatched

When we don't know, we get burned

- Log4Shell (CVE-2021-44228), Dec 2021 : remote code execution
- The real damage: most orgs couldn't tell if they even used log4j
- It hid as a transitive dependency, buried deep
- Teams spent days just finding out if they were exposed

CVE-2021-44228

Why SBOMs matter



🚨 *"Do we use log4j?" nobody knows*
✅ Query → found, version 2.14.1, vulnerable

🚨 *Hard to assess radius across orgs? What services were affected?*
✅ Affected: 3 services, 2 teams

🚨 *Days to assess impact?*
✅ *Minutes, not days*

But can you trust your SBOM?

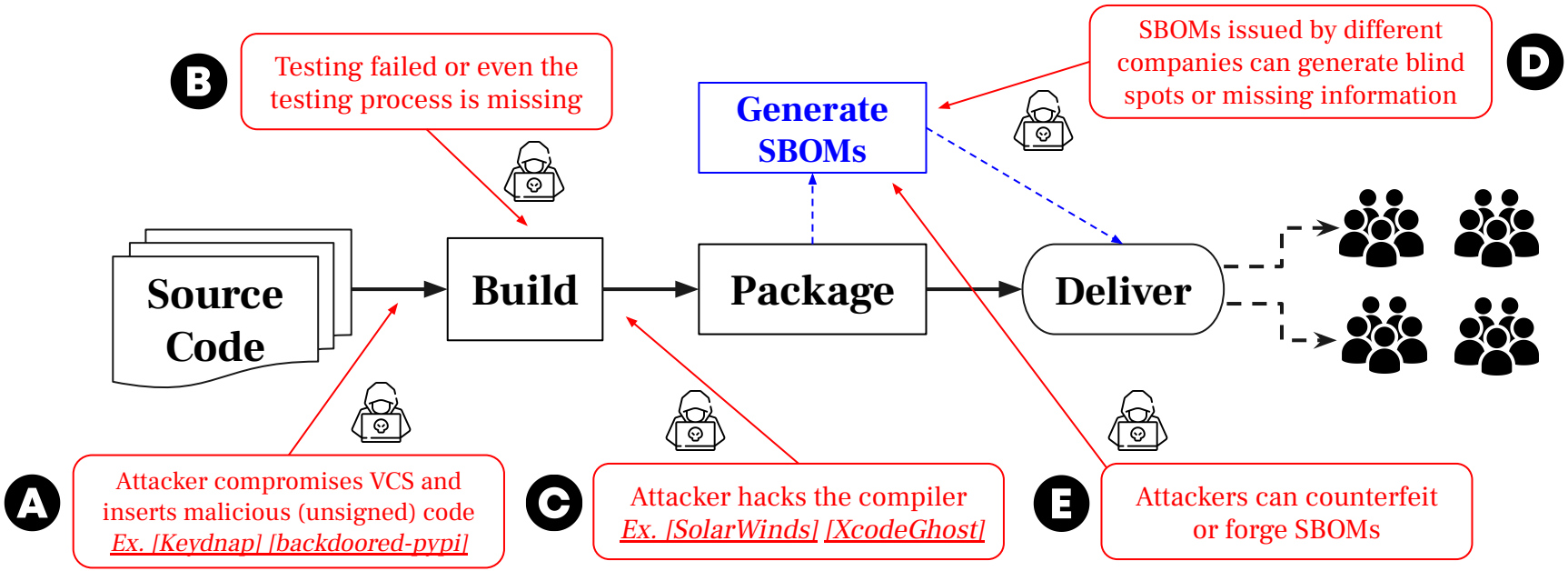
🚨**How was this SBOM generated?**
✅ With Witness : from the real build, not a manifest

🚨 **What information did it capture?**
✅ Processes, files, and the network : not just declared dependencies

🚨 **Is this information accurate?**
✅ Recorded from reality, not the plan (an attestation : records what really happened)

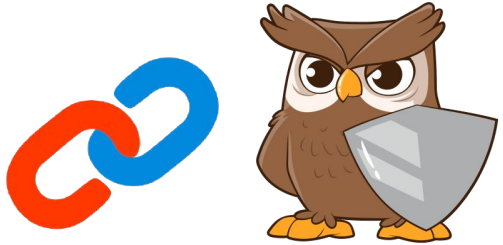
🚨 **Can you verify its authenticity?**
✅ Yes, it's cryptographically signed (DSSE : makes it verifiable)

Recap: Where Traditional SBOMs Break



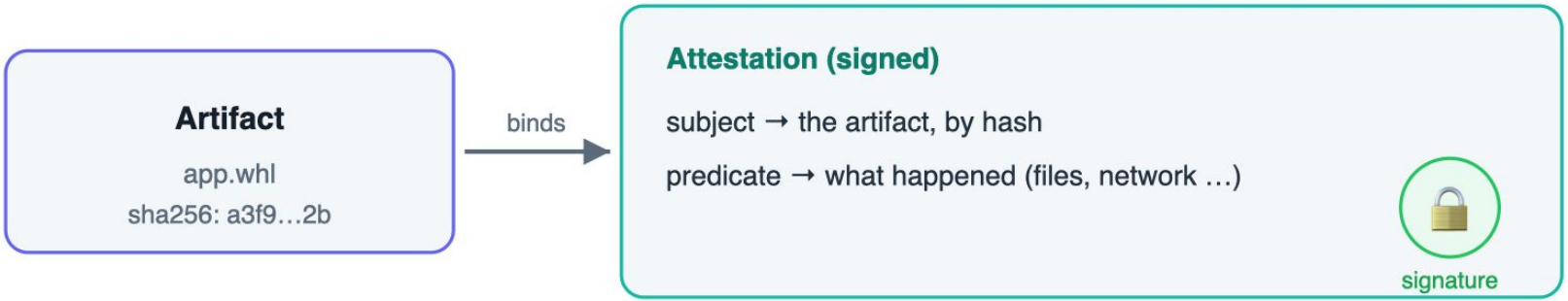
What we actually need is to verify the full story: who did what, when, and with what input. And that's exactly where in-toto comes in."

What is an attestation?



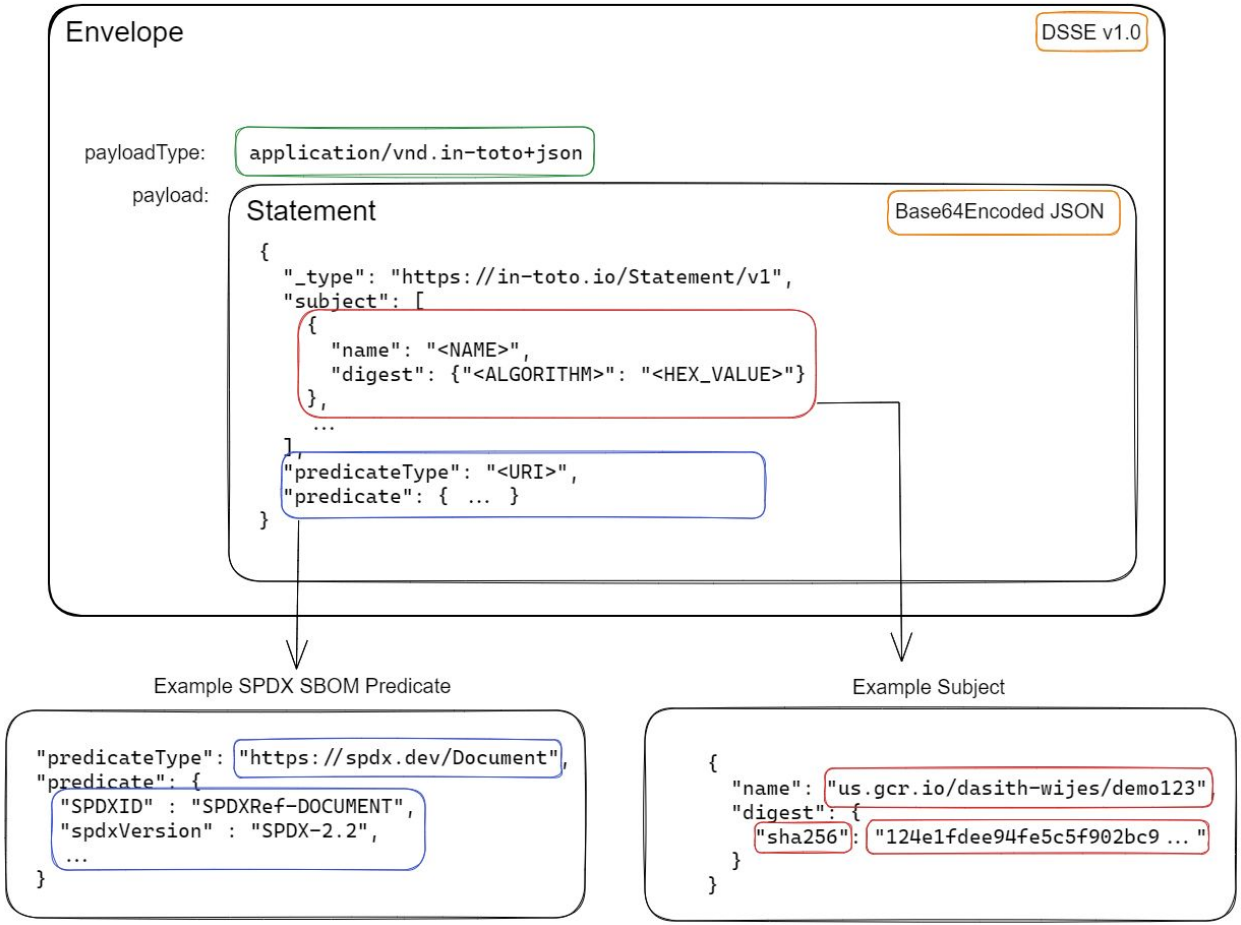
- A signed statement about a specific software artifact
- Structure: subject (which artifact, by hash) + predicate (the data/claim)
- Data inside an attestation is bound to that exact build, typed, and verifiable
- Not a log file you have to trust, evidence anyone can check

An attestation = a signed statement bound to one artifact



Data inside an attestation = evidence you can verify, not a log.

in-toto Attestation Framework

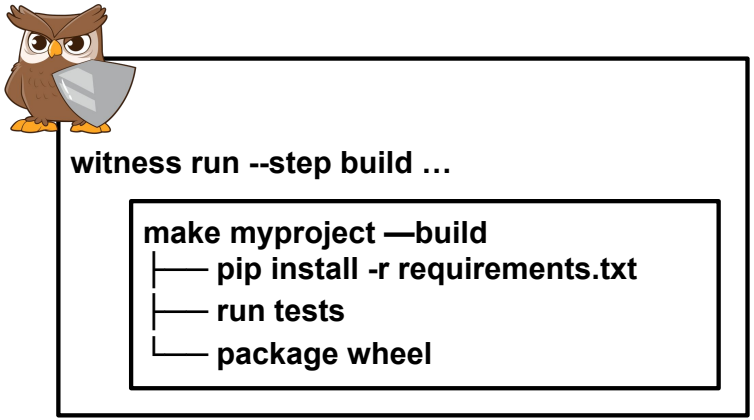


KubeCon CloudNativeCon India 2026

Wait…

Accurate SBOMs ?

Witness (in-toto): Monitor the actual build



Processes Spawned

Files Accessed

Git Commits

Environment

Network Calls

…

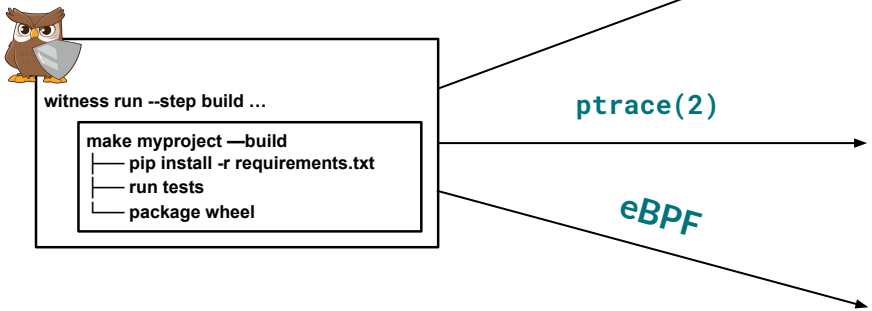
https://slsa.dev/spec/v1.0/about

https://in-toto.io/docs/what-is-in-toto/

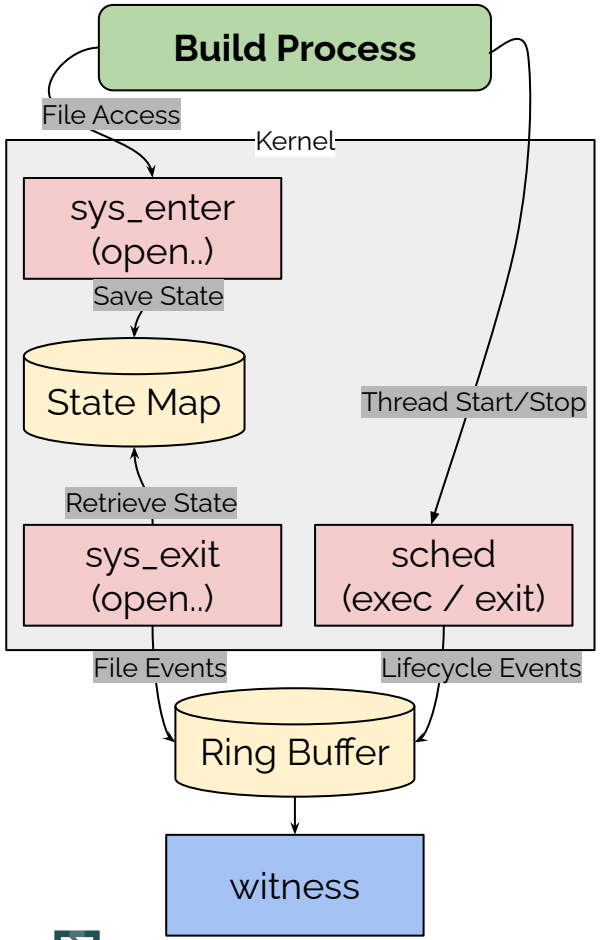
```
$ ./witness attestors list --experimental -c .witness.yaml
INFO    Using config file: .witness.yaml
```

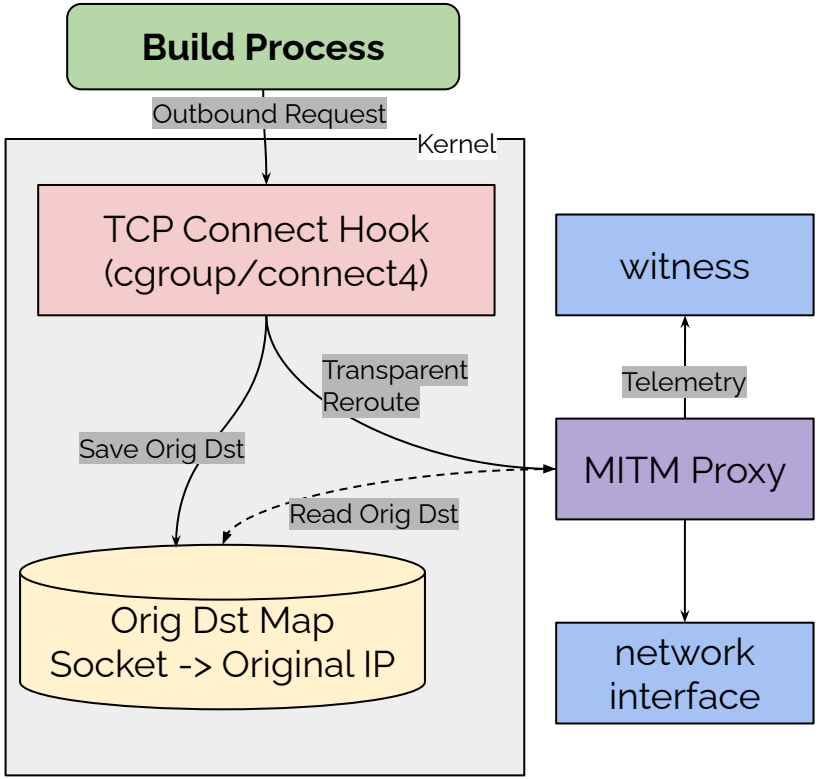
NAME	TYPE	RUN TYPE
maven	https://witness.dev/attestations/maven/v0.1	prematerial
sarif	https://witness.dev/attestations/sarif/v0.1	postproduct
policyverify	https://slsa.dev/verification_summary/v1	verify
aws	https://witness.dev/attestations/aws/v0.1	prematerial
slsa	https://slsa.dev/provenance/v1.0	postproduct
environment (default)	https://witness.dev/attestations/environment/v0.1	prematerial
jenkins	https://witness.dev/attestations/jenkins/v0.1	prematerial
omnitrail	https://witness.dev/attestations/omnitrail/v0.1	prematerial
product (always run)	https://witness.dev/attestations/product/v0.1	product
system-packages	https://witness.dev/attestations/system-packages/v0.1	prematerial
vex	https://openvex.dev/ns	postproduct
oci	https://witness.dev/attestations/oci/v0.1	postproduct
secretscan	https://witness.dev/attestations/secretscan/v0.1	postproduct
material (always run)	https://witness.dev/attestations/material/v0.1	material
link	https://in-toto.io/attestation/link/v0.3	postproduct
sbom	https://witness.dev/attestations/sbom/v0.1	postproduct
git (default)	https://witness.dev/attestations/git/v0.1	prematerial
gcp-iit	https://witness.dev/attestations/gcp-iit/v0.1	prematerial
gitlab	https://witness.dev/attestations/gitlab/v0.1	prematerial
network-trace (experimental)	https://witness.dev/attestations/network-trace/v0.1	execute
aws-codebuild	https://witness.dev/attestations/aws-codebuild/v0.1	prematerial
command-run (always run)	https://witness.dev/attestations/command-run/v0.1	execute
docker	https://witness.dev/attestations/docker/v0.1	postproduct
jwt	https://witness.dev/attestations/jwt/v0.1	prematerial
github	https://witness.dev/attestations/github/v0.1	prematerial
k8smanifest	https://witness.dev/attestations/k8smanifest/v0.2	postproduct
lockfiles	https://witness.dev/attestations/lockfiles/v0.1	prematerial

Witness Attestation Payload




```
"cmd":  ["pip", "install", "-r", "requirements.txt"],

"stdout": "Downloading urllib3-0.61.0-py3-none-any.whl",

"processes": [
   "program": "/usr/bin/pip",
   "processid": 47855,


   "openedfiles": {
        "...urllib3-1.26.5.egg-info/requires.txt":
             {"sha256": "7da776bbb1...."},
        "/lib/x86_64-linux-gnu/libssl.so.3":
             { "sha256": "660a6abea..."}


   "networkcalls": [
       "protocol": "tcp",
       "src_addr": "127.0.0.1:47644",
       "dst_addr": "https://pypi.org:443/simple/urllib3/",
       "data": {
           "request":
           "response":
```


Build Process
File Access
Kernel
sys_enter
(open..)
Save State
State Map
Thread Start/Stop
Retrieve State
sys_exit
(open..)
sched
(exec / exit)
File Events
Lifecycle Events
Ring Buffer
witness


Build Process
Outbound Request
Kernel
TCP Connect Hook
(cgroup/connect4)
witness
Telemetry
Transparent
Reroute
Save Orig Dst
MITM Proxy
Read Orig Dst
Orig Dst Map
Socket -> Original IP
network
interface

KubeCon CloudNativeCon India 2026

Making SBOMs honest with SBOMit

SBOMit = SBOM + in-toto

SBOM + witness/in-toto = SBOMit

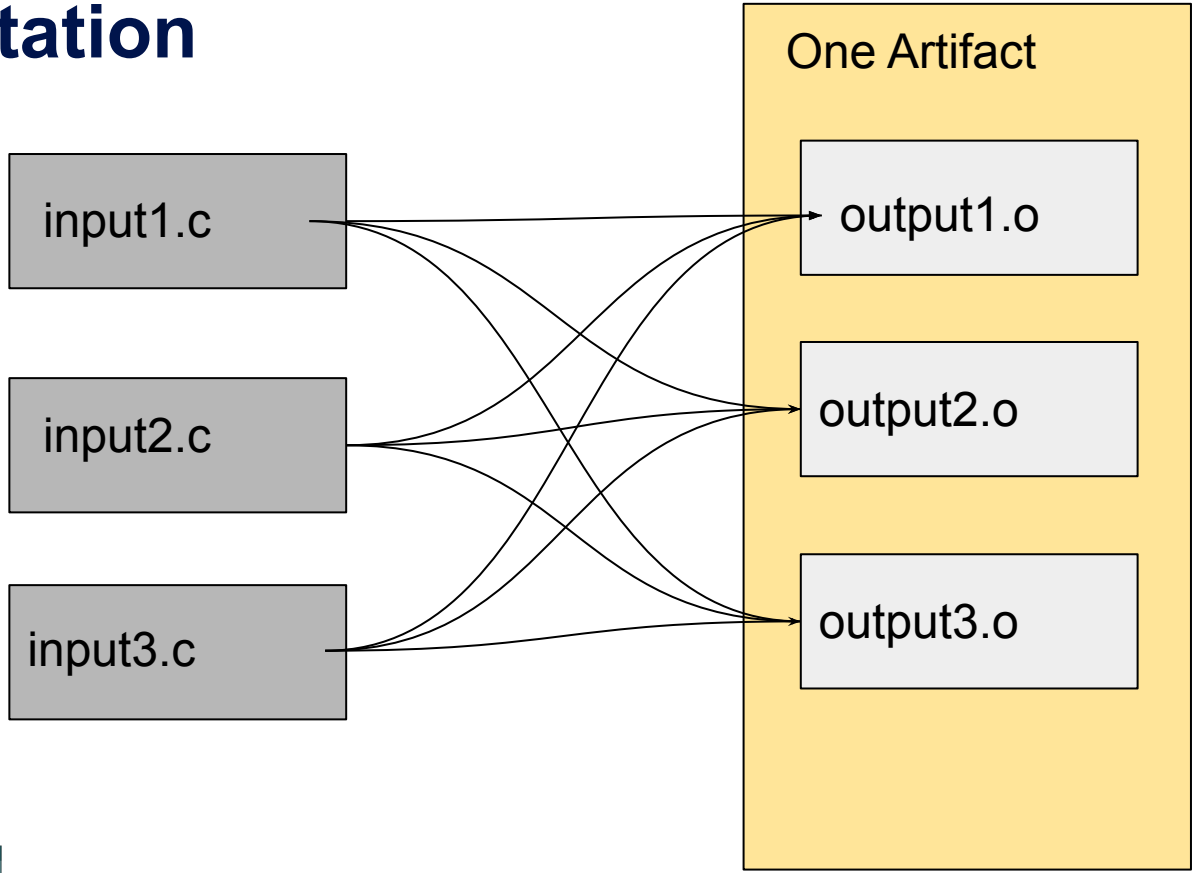
SBOM + 

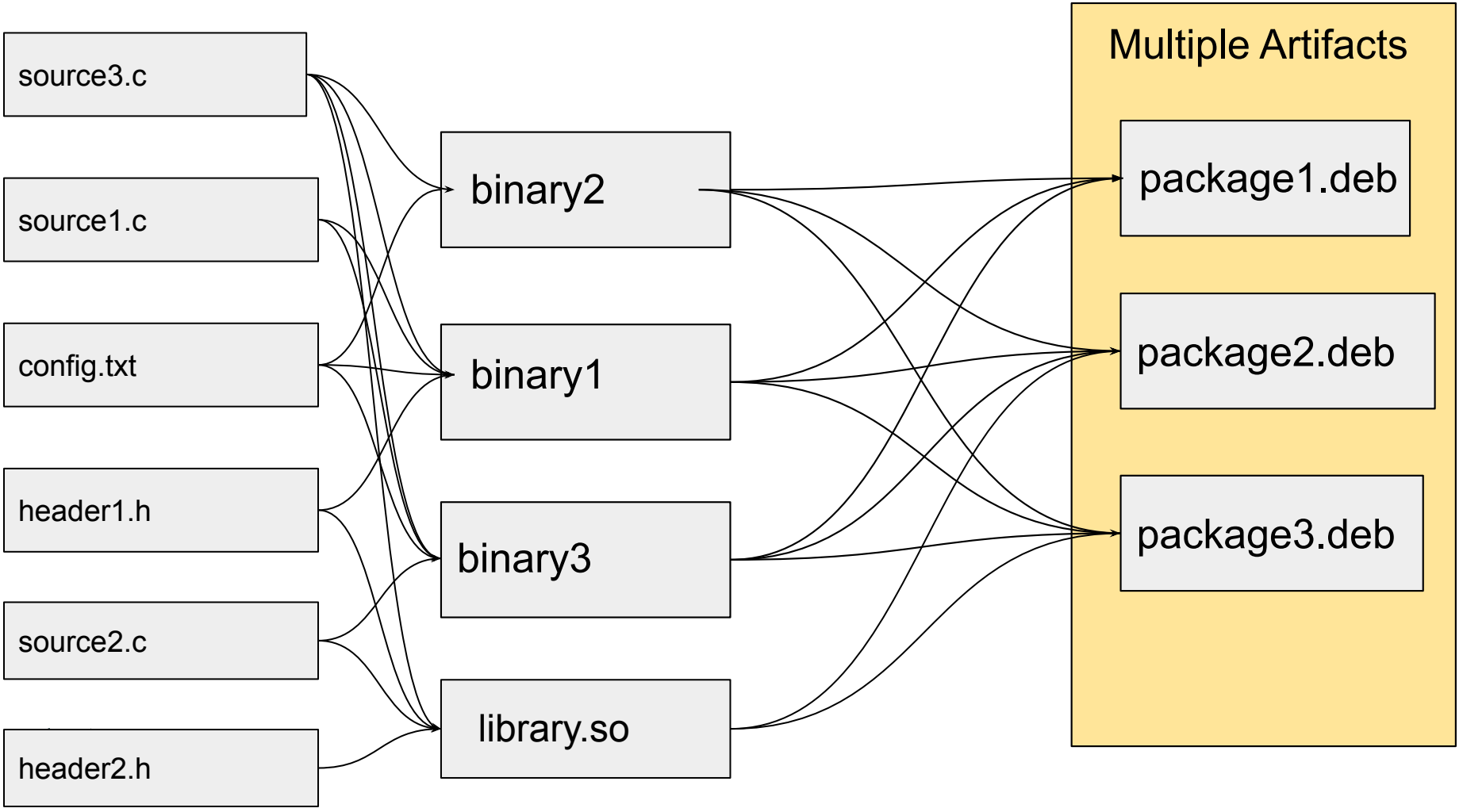
= Enriched SBOM

KubeCon CloudNativeCon India 2026

Demo

!Limitation


One Artifact
input1.c
input2.c
input3.c
output1.o
output2.o
output3.o


source3.c
source1.c
config.txt
header1.h
source2.c
header2.h
binary2
binary1
binary3
library.so
Multiple Artifacts
package1.deb
package2.deb
package3.deb

Join Us!

- First Friday of every month
- #in-toto & #in-toto-witness @ CNCF Slack Workspace
- #sbomit @ OpensSSF Slack Workspace
- https://github.com/in-toto
- https://github.com/in-toto/witness

KubeCon CloudNativeCon India 2026

Thank you