SUPPLYCHAIN SECURITYCON @ OPEN SOURCE SUMMIT JAPAN

THE LINUX FOUNDATION

The Telemetry of Trust, Using Attestations to Secure Your SDLC w/ Open Source Tools

Jesse Sanford - Software Architect, Autodesk
Jagadish Ramidi - Security Software Engineer, Autodesk

Jesse Sanford

Software Architect, Autodesk

Jesse is a lifelong software engineer focused on site reliability and Infosec. Currently architecting the juncture of platform engineering and security/compliance for Autodesk's Developer Enablement team. He regularly contributes to open source and frequently speaks about his work utilizing it at Autodesk and in the community. When not in front of a computer, he is a backpacker, sailor and continuously delivering parent of two young daughters.

Jagadish Ramidi

Software Engineer, Autodesk

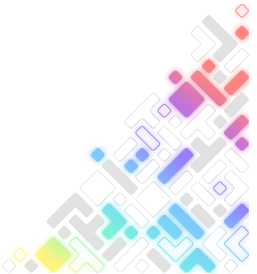
Works as a security software engineer at Autodesk focusing on software composition analysis and supply chain security.

Agenda

- Level set on S3C (Secure Software Supply Chain)
- S3C challenges in practice
- Telling the SDLC “trust story”
- Attestations!
- Demo
- How to get started and where we can go

Let’s level set…

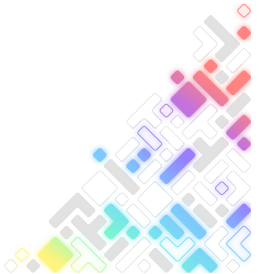
- OSS progress over past few years is good
  - Standards: (SBOM, SLSA 1.X, SSDF 1.X, IETF SCITT)
  - Public Good Services: Sigstore, Alpha-Omega project
  - Major SCM integrations (Github, Gitlab)
  - Sigstore for package managers and major projects (K8s, NPM, Homebrew)
- But internally adoption is slow…
  - Technology inertia (Legacy / brown field)
  - Compliance regime deltas



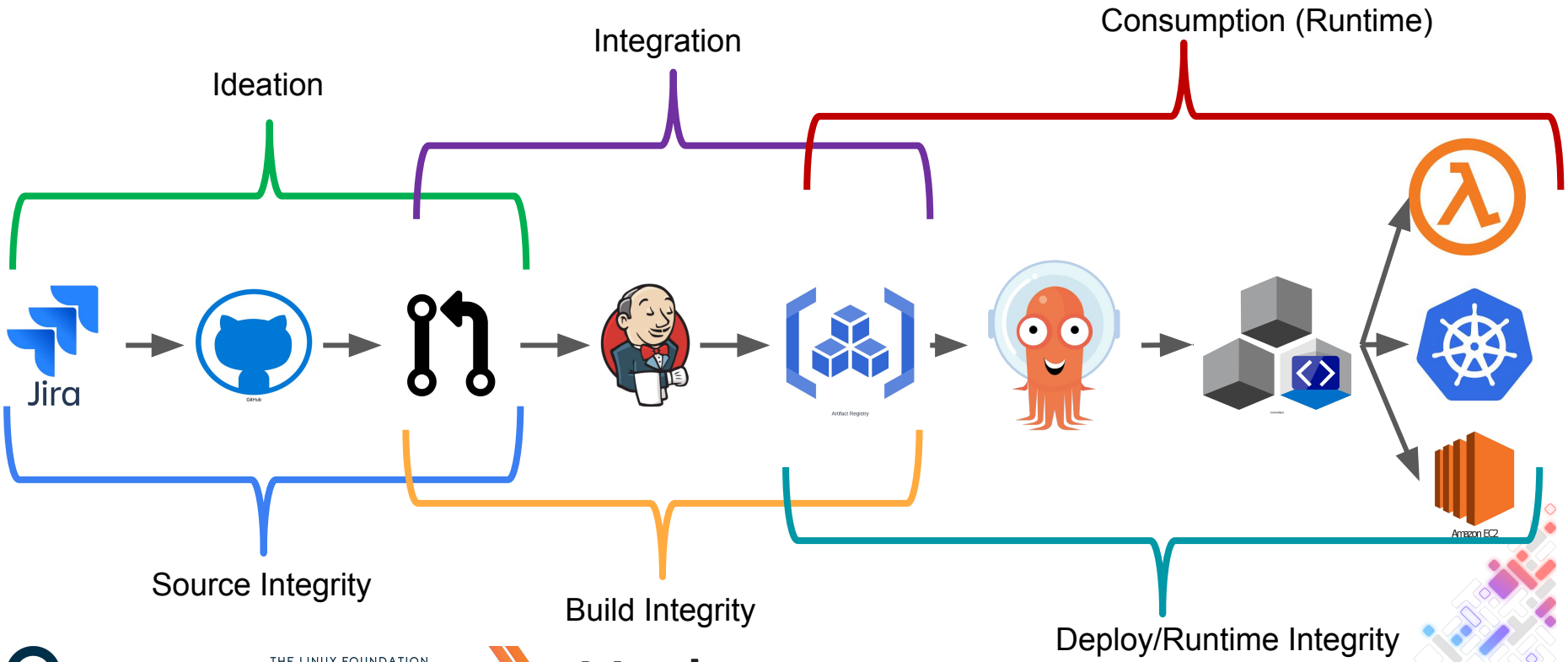
S3C Challenges

- Technology inertia is real!
- Continuous M&A
- How can we wrap all that diversity?
  - Adaptable
  - Encompass the complete SDLC

AUTODESK



Autodesk SDLC overview



THE BUILD…

An ~~un~~EXPECTED journey

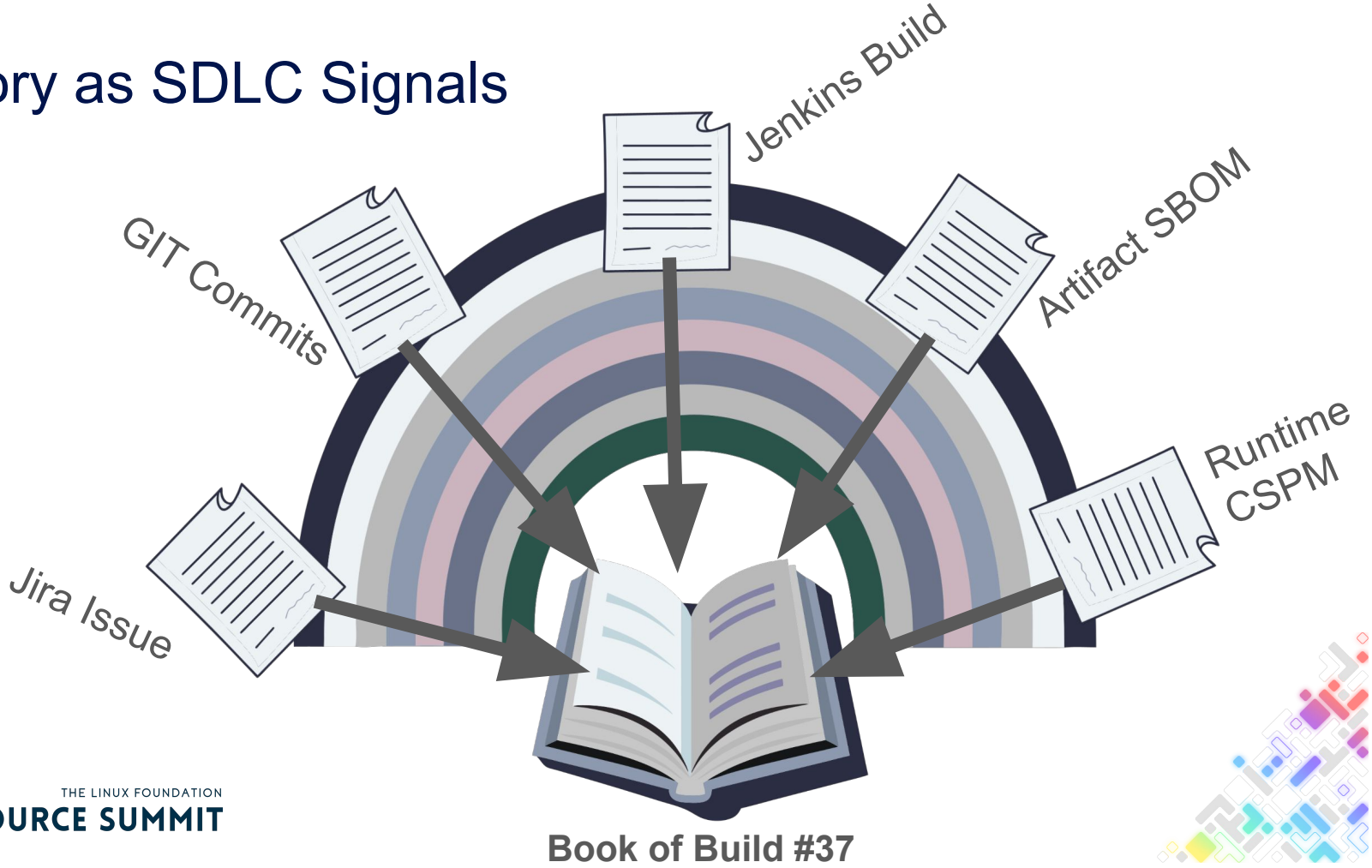
(And trusted!)

Telling the SDLC “story”

For a particular build, let’s tell it’s story:

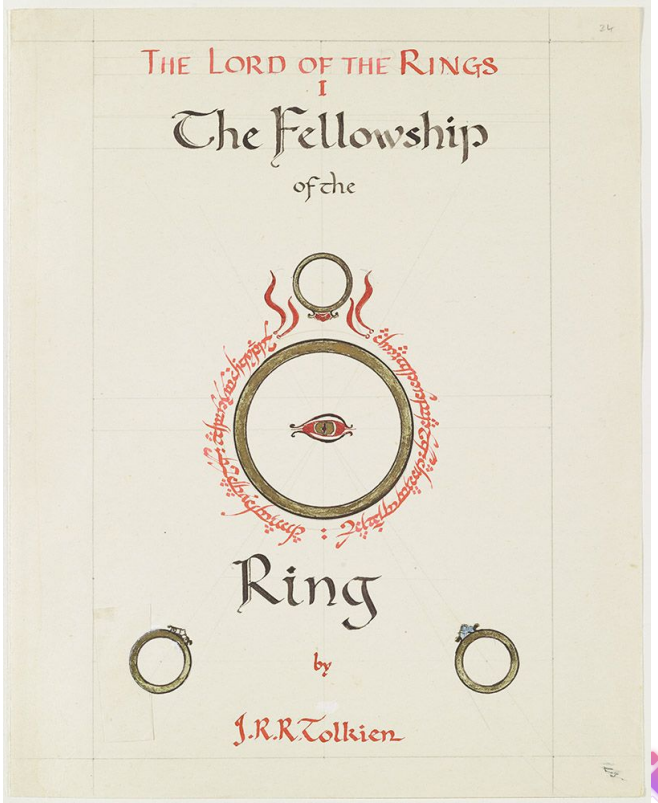
- Every build is it’s own book
- Each phase is another “chapter”
- Each step in a phase (like CI) is another page
- Each page can have multiple “statements”

The Story as SDLC Signals

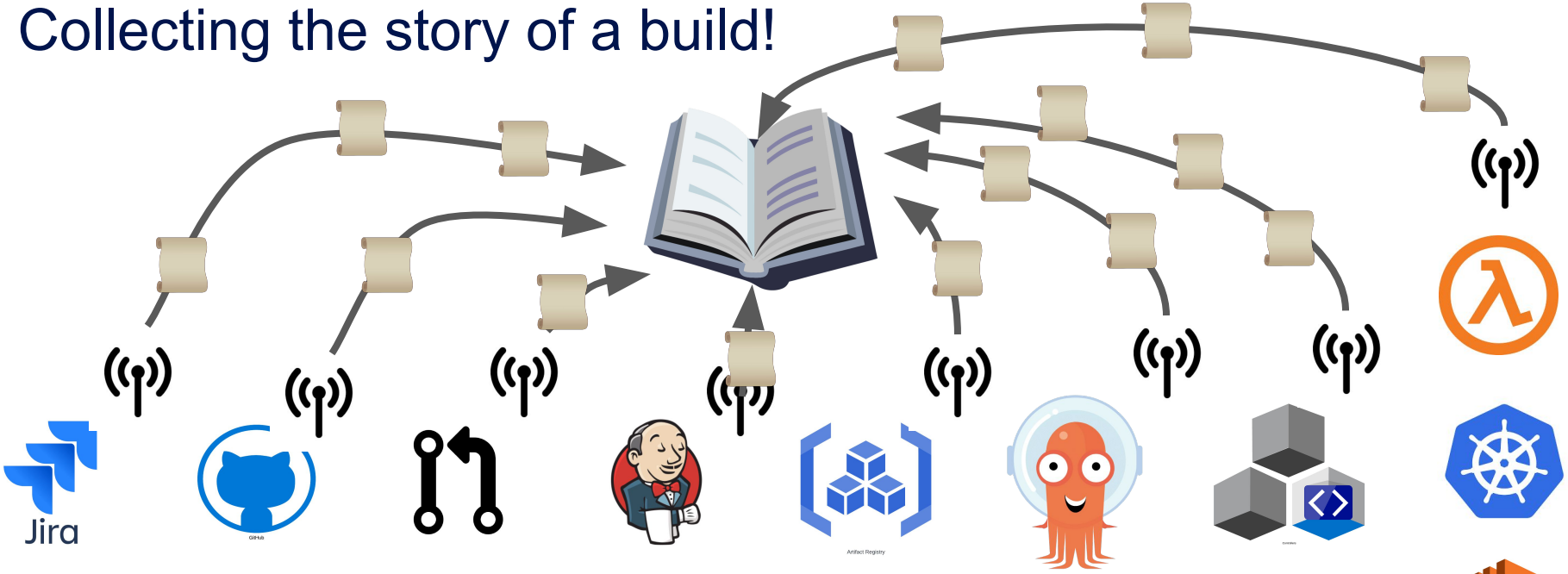


Telling the SDLC “story”

- We can stop the story if we don’t like it
- We can skim the book
  - Focus on pages that build “trust” faster
  - Then go back and embellish



Collecting the story of a build!



Key: *Trust Telemetry*

Attestation Instrumentation

The “Trust Telemetry” of the Story

- We can start to judge builds against expectations
  - Specified through policies
  - Policies for each phase or even each step
  - Only gate when risk requires
  - Manual review when things look strange?

Daily Trust Telemetry

Daily Trust Telemetry

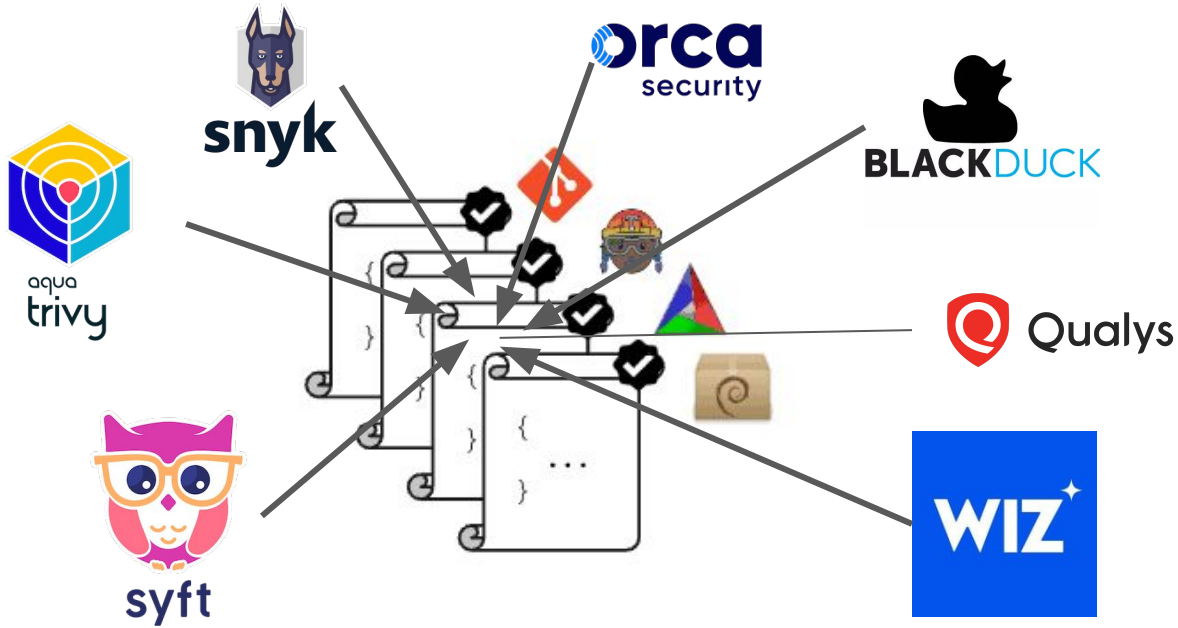
The Signals of Trust

- A snapshot of SDLC
- Each signal is a statement
  - Something we expect to happen
  - But it must be instrumented!
- The statements are “attested” to
  - Just like testifying to a statement in a court

Attestations provide the context

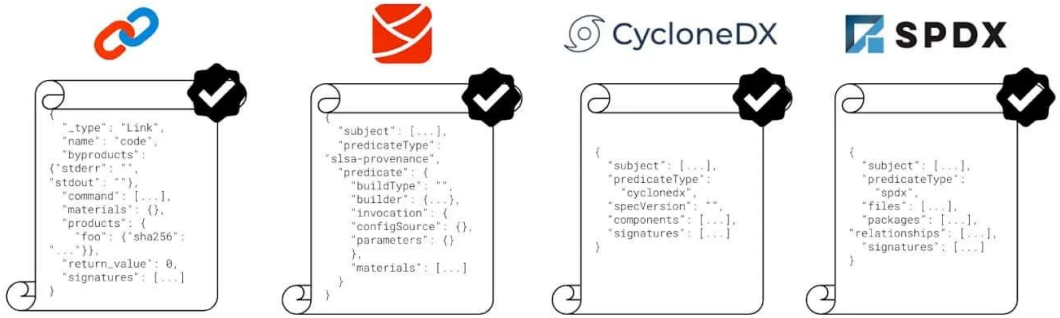
- Individually, they provide little trust
- In aggregate, stronger than the sum of their parts
- Brings picture into focus
- More context = Smarter Decisions

Attestations bridge the diversity



The API of DevSecOps?

Towards Contextual in-toto Attestations

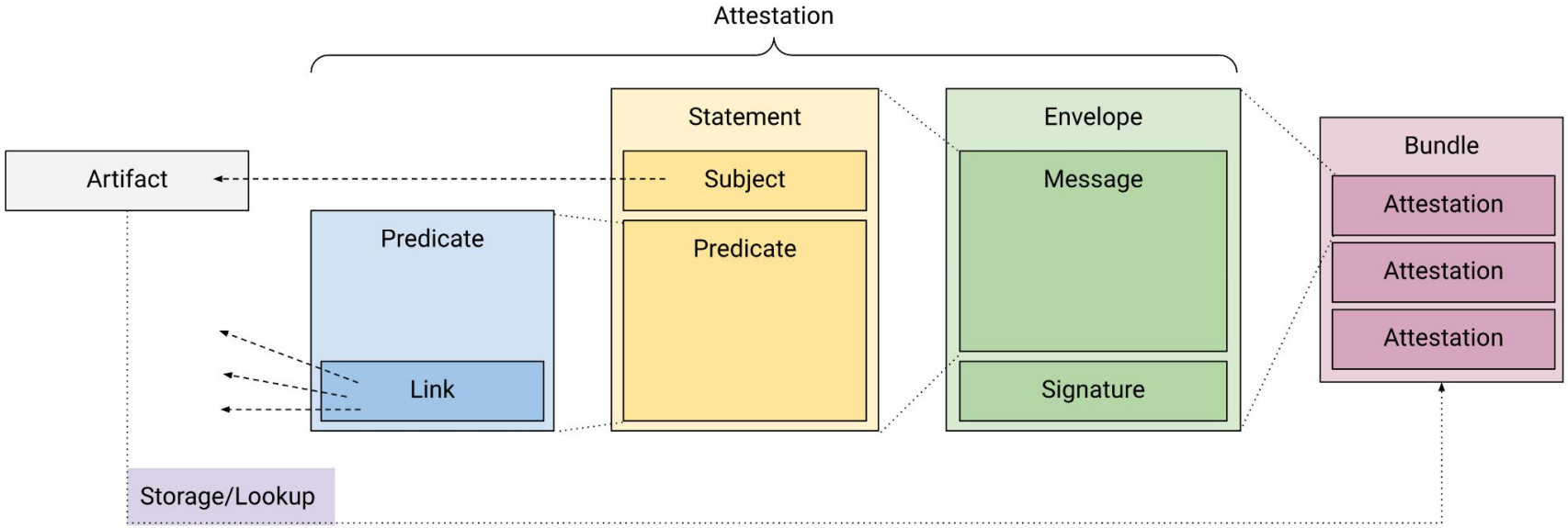


- ITE-10: https://github.com/in-toto/ITE/blob/master/ITE/10/README.adoc
- Blog post: https://www.cncf.io/blog/2023/08/17/unleashing-in-toto-the-api-of-devsecops/
- Kubecon EU 2023 Preso: https://www.youtube.com/watch?v=ezV_oWBPqKw

Focus on Attestations

- “Claims” about a process
- Metadata
- Time stamped
- Persistent
- Verifiable
- Examples:
  - in-toto attestations
  - SCITT statements

Focus on attestations ctd

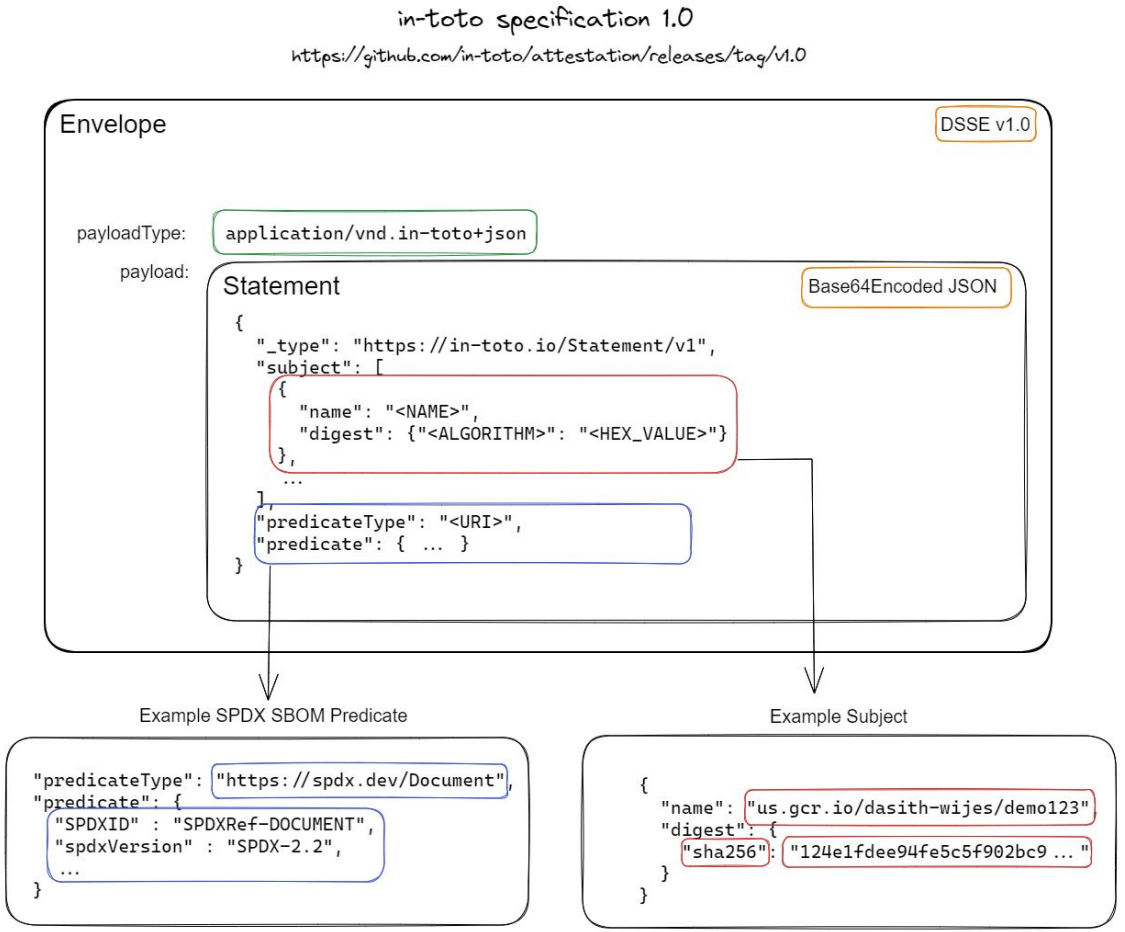


The in-toto Attestation

Predicate: Defines an action taken

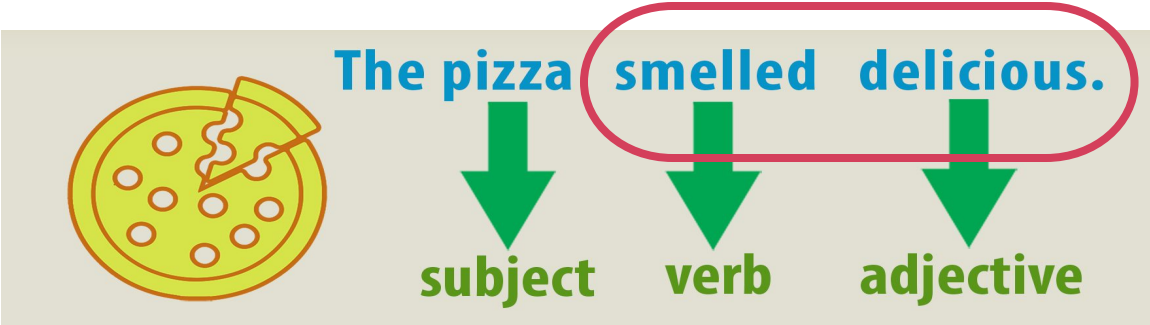
Statement: Binds the action to the subject of the action.

Envelope: Enables verification and ultimately trust.



Predicates = the “types” of actions

- “The software was scanned”
- Many common predicates or “types”
  - SBOM
  - SLSA Provenance
  - Test results
  - Runtime Traces
- Custom predicates
  - Escape hatch
  - Semantics are hard



Let's look at example attestations

SLSA Provenance:

```
{"_type": "https://in-toto.io/Statement/v1",
  "subject": [...],

  "predicateType": "https://slsa.dev/provenance/v1",

  "predicate": {
   "buildDefinition": {
    "buildType": string,
    "externalParameters": object,
    "internalParameters": object,
    "resolvedDependencies": [ ...#ResourceDescriptor ],
   },
   "runDetails": {
    "builder": {
      "id": string,
      "builderDependencies": [ ...#ResourceDescriptor ],
      "version": { ...string },
    },
    "metadata": {
      "invocationId": string,
      "startedOn": #Timestamp,
      "finishedOn": #Timestamp,
```

```
#ResourceDescriptor: {
  "uri": string,
  "digest": {
    "sha256": string,
    "sha512": string,
    "gitCommit": string,
    [string]: string,
  },
  "name": string,
  "downloadLocation": string,
  "mediaType": string,
  "content": bytes, // base64
  "annotations": object,
}
```

Other interesting ones…

SBOM:

```
{  "_type": "https://in-toto.io/Statement/v0.1",
  "subject": [{ ... }],

"predicateType":"https://cyclonedx.org/bom/v1.4",

"predicate": {
    "bomFormat": "CycloneDX",
    "specVersion": "1.4",
    "serialNumber":
"urn:uuid:3e671687-395b-41f5-a30f…",
    "version": 1,
    "components": [
        {
        "type": "library",
        "name": "acme-library",
        "version": "1.0.0"
        }
    ]
    ...
  }
```

Vuln Scan:

```
{ "_type": "https://in-toto.io/Statement/v0.1",
  "subject": [{...}],

  "predicateType":
"https://cosign.sigstore.dev/attestation/vuln/v1"
,

  "predicate": {
    "invocation": {
      "parameters": [],
      // [ "--format=json", "--skip-db-update" ]
      "uri": "",
      "event_id": "",
      "builder.id": "" …
```

Generating Attestations

A few tools can create attestations:

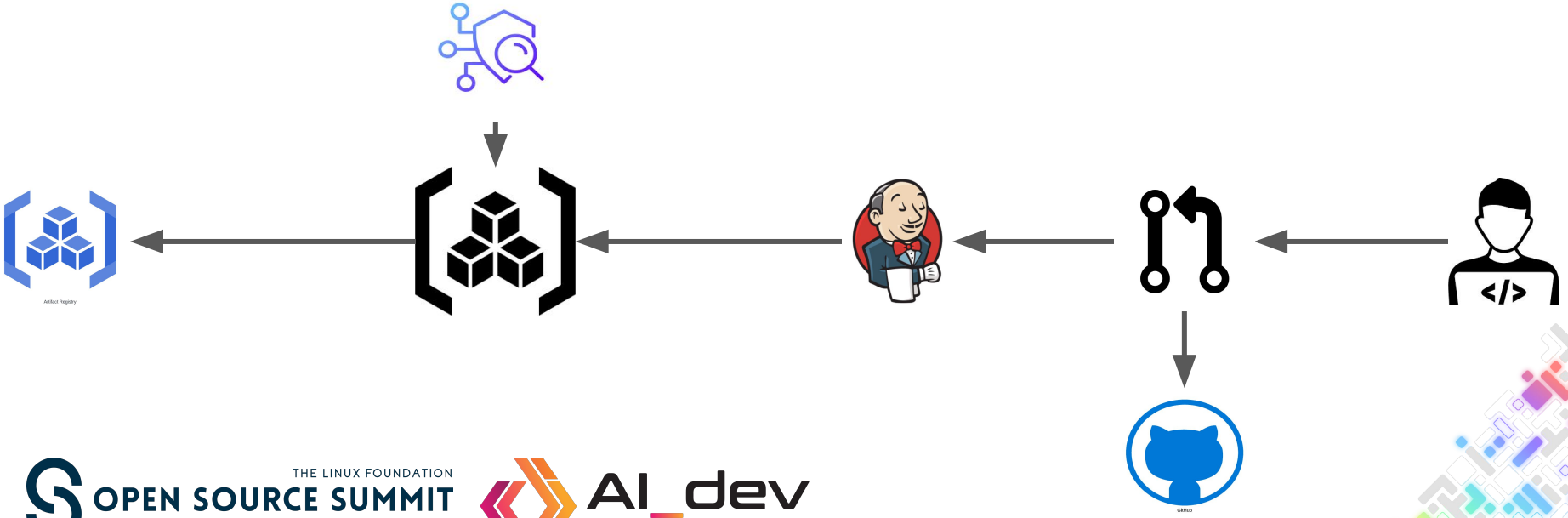
- in-toto suite (in-toto-run, in-toto-record, in-toto-sign)
  - Requires Layouts
  - Creates Link files
- witness
  - Does not need layouts
  - Automated data collection
  - Leverages in-toto attestations
- cosign
  - Part of the sigstore suite
  - Can take in attestation as json
  - Store with container image in OCI registry
- slsa-attestor
  - Specifically for slsa provenance

Spotlight on Witness

- Allows us to declare policies independently of build pipeline design
- Automated Data collection to reduce developer friction.
- Allows separation of concerns from software developers and policy owners
    - Important when security and compliance teams need ownership over policy

But where are they stored?

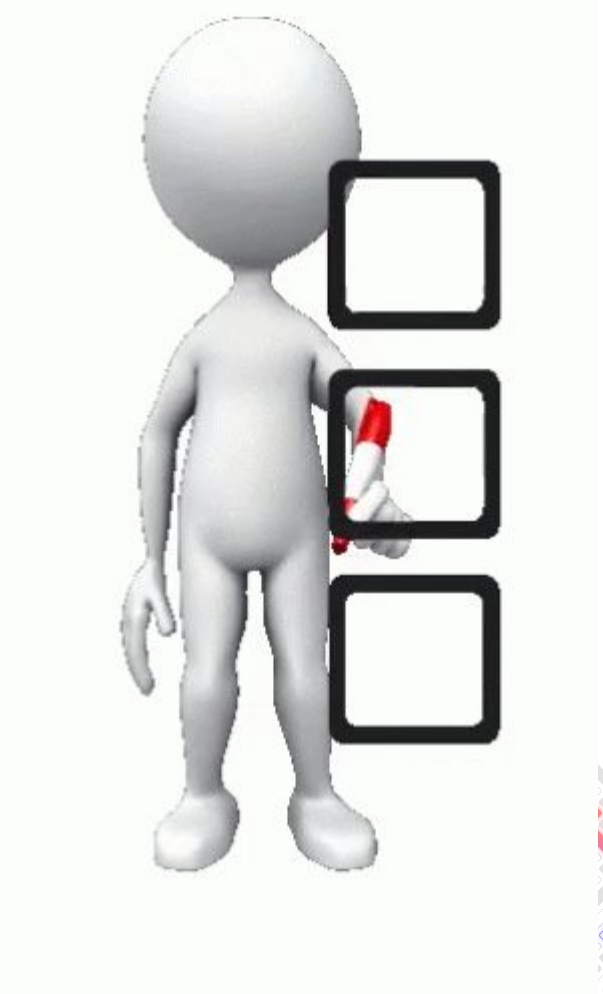
- Archivista is a graph and storage service for in-toto attestations.
- Enables the discovery and retrieval of attestations for software artifacts.



Verifying Attestations

And their verifying pairs:

- in-toto-verify
  - Verifies layouts and link files
- witness verify command
  - Verifies against witness (rego) policies
- cosign verify
  - Validates attestation signatures
  - Can validate with cue or rego policies
- slsa-verifier
  - Verifies slsa provenance attestations
  - Limited to provenance specifics
    - Builder id
    - Src code repo info



Now let’s see an attestation get generated

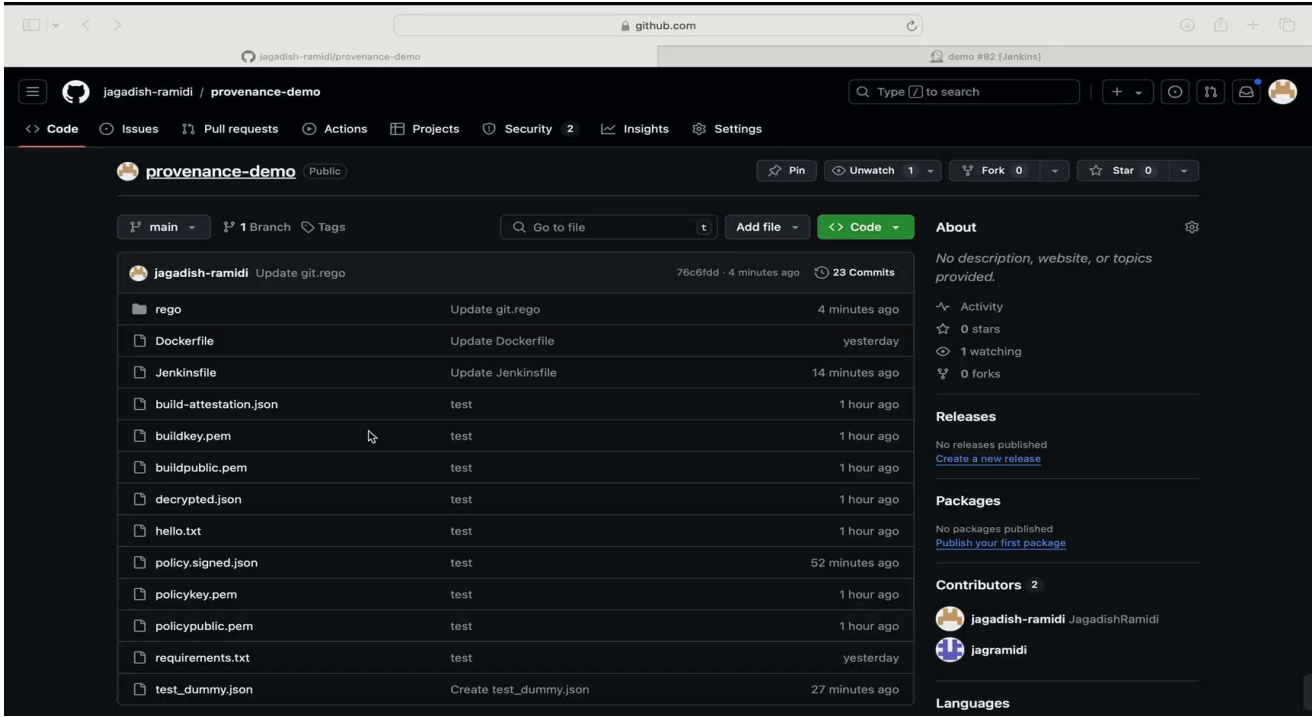
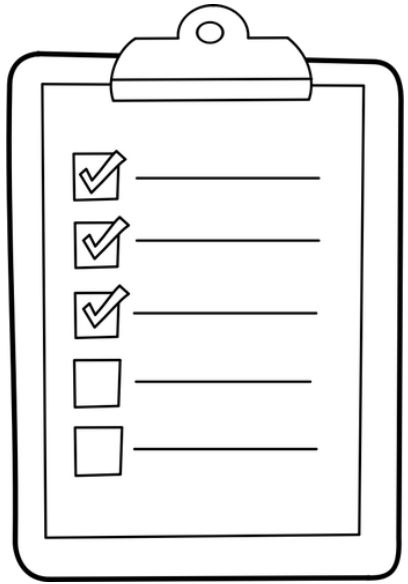


Diagram of the build SDLC as attestations in graphql viz

How start telling your SDLC stories?

- Start with accounting your processes
  - Threat modeling helps!
- Find places that you can instrument
  - The common “predicate types” are obvious
- Sort by those that instill greatest “trust”
- Instrument and write policy to use them
- Repeat!

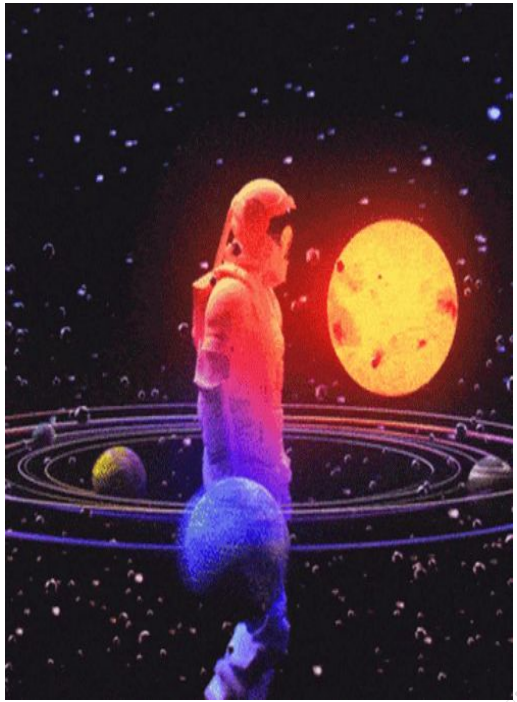


Where can we take this?

- Centralized policy evaluation
  - More on this next…
- AI/ML
  - Trained on data lake
  - Augment with sw quality and incident data
  - Enable earlier “trust” decisions

Centralized Policy Decisions

- Central data lake w/ graph API is foundational
- Policy evaluation API
- Policy becomes pluggable
- Diverse environment becomes easier
  - Avoid policy distribution issues!
- Strong separation of concerns
- OOB policy execution
  - Backtest policies (w/out re-running pipelines)



Thank you!

KCNA 24 Talk Coming SOON!
Secure by Design CI/CD: Practical Insights from Adobe and Autodesk - Vikram Sethi, Adobe Inc. & Jesse Sanford, Autodesk

https://sched.co/1i7lS

Slides/demo and links:

QUESTIONS?

https://github.com/jessesanford/scscon-japan-2024

Follow us on linkedin!

/in/jessesanford

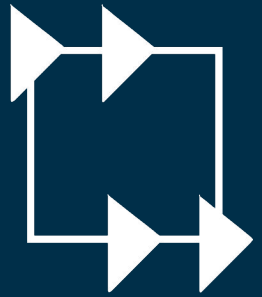
/in/jagadishramidi/

THE LINUX FOUNDATION OPEN SOURCE SUMMIT JAPAN

AI_dev Open Source GenAI & ML Summit

THE LINUX FOUNDATION

OPEN SOURCE SUMMIT JAPAN



SUPPLYCHAIN SECURITYCON