

Kubecon EU 2023

**Silly gooses,
let's make sense of the security supply chain,
*together***

Grace Nguyen

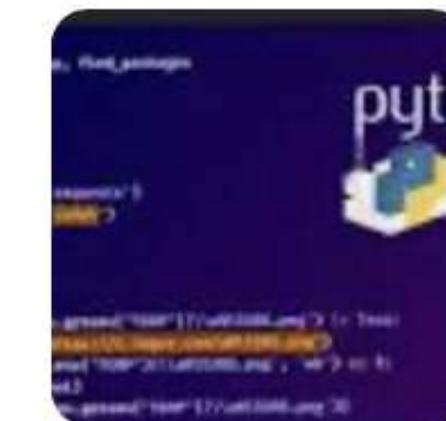


- Security issues introduced by the **third-party** components and technologies
- **700% increase** in attacks over 3 years
- Executive Order 14028

The Hacker News

W4SP Stealer Constantly Targeting Python Developers in Ongoing Supply Chain Attack

Nov 18, 2022



Reuters

SolarWinds hack was 'largest and most sophisticated attack' ever: Microsoft president



Feb 14, 2021

Dark Reading

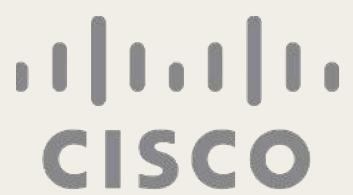
Twilio Hackers Scarf 10K Okta Credentials in Sprawling Supply Chain Attack



Aug 25, 2022



Canada Revenue
Agency



Meraki



- Kubernetes release team since v1.22
- Release Manager Associate v1.27
- v1.28 Release Lead

Riskfuel

^Expo

**not an expert*



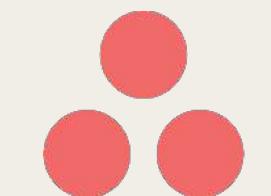
Canada Revenue
Agency

ERICSSON ≡

Riskfuel



Meraki

 **asana**

^Expo

- Kubernetes release team since v1.22
- Release Manager Associate v1.27
- v1.28 Release Lead

Agenda

1. Intro ✓
2. Frameworks
3. Landscape and tools
4. Misconceptions and discussion
5. What's next?

Agenda

1. Intro 
2. Frameworks
3. Landscape and tools
4. Misconceptions and discussion
5. What's next?

*goal: to establish foundation and familiarity
to software supply chain security*

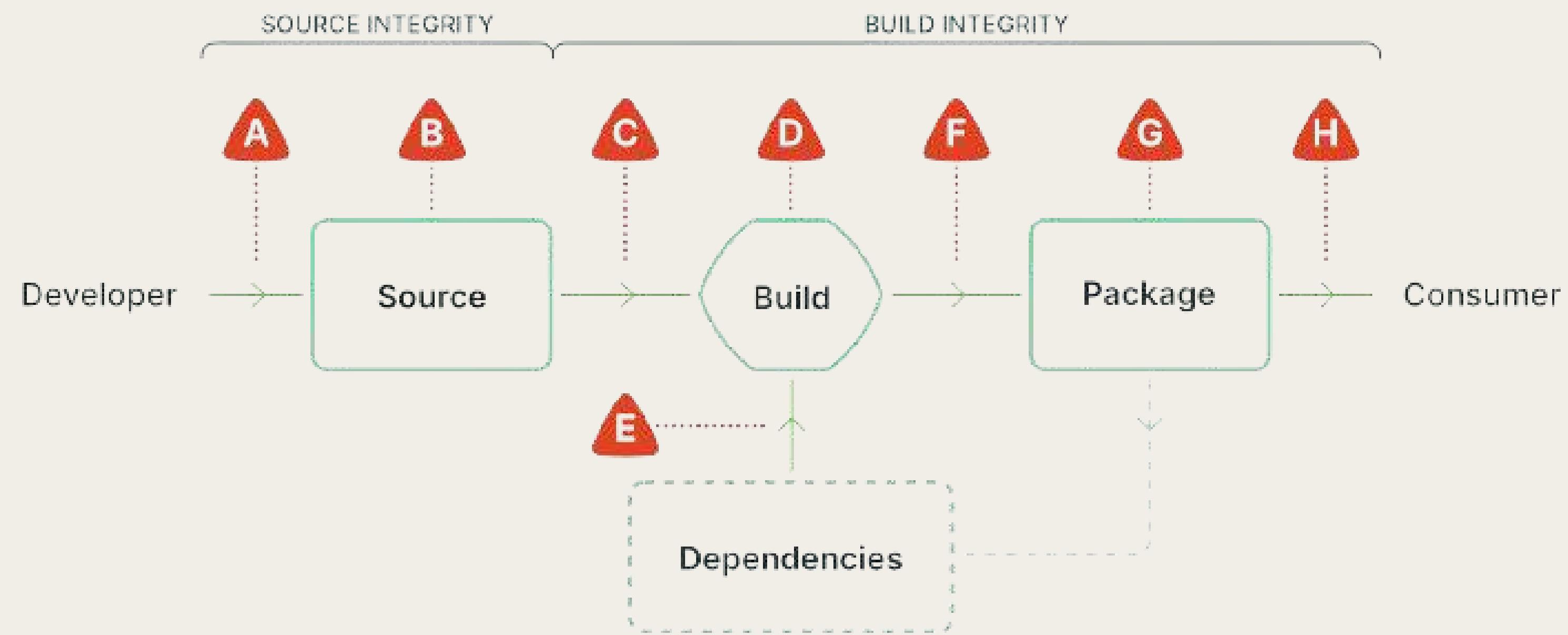
I excel at new, hard things

2. Framework

- **Attestation**
- **Provenance**
- **SBOM**

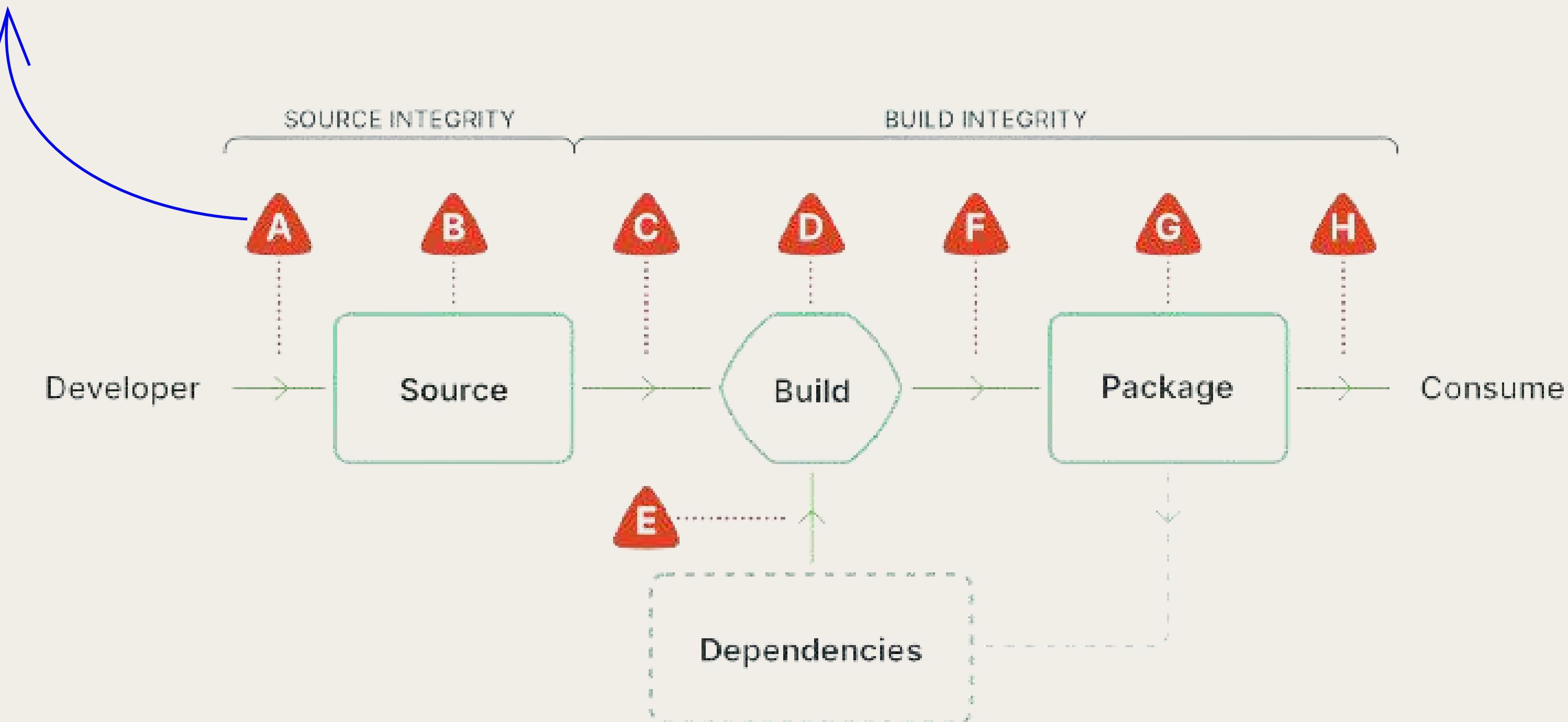
Level	Description	Example
1	Documentation of the build process	Unsigned provenance
2	Tamper resistance of the build service	Hosted source/build, signed provenance
3	Extra resistance to specific threats	Security controls on host, non-falsifiable provenance
4	Highest levels of confidence and trust	Two-party review + hermetic builds

from slsa.dev

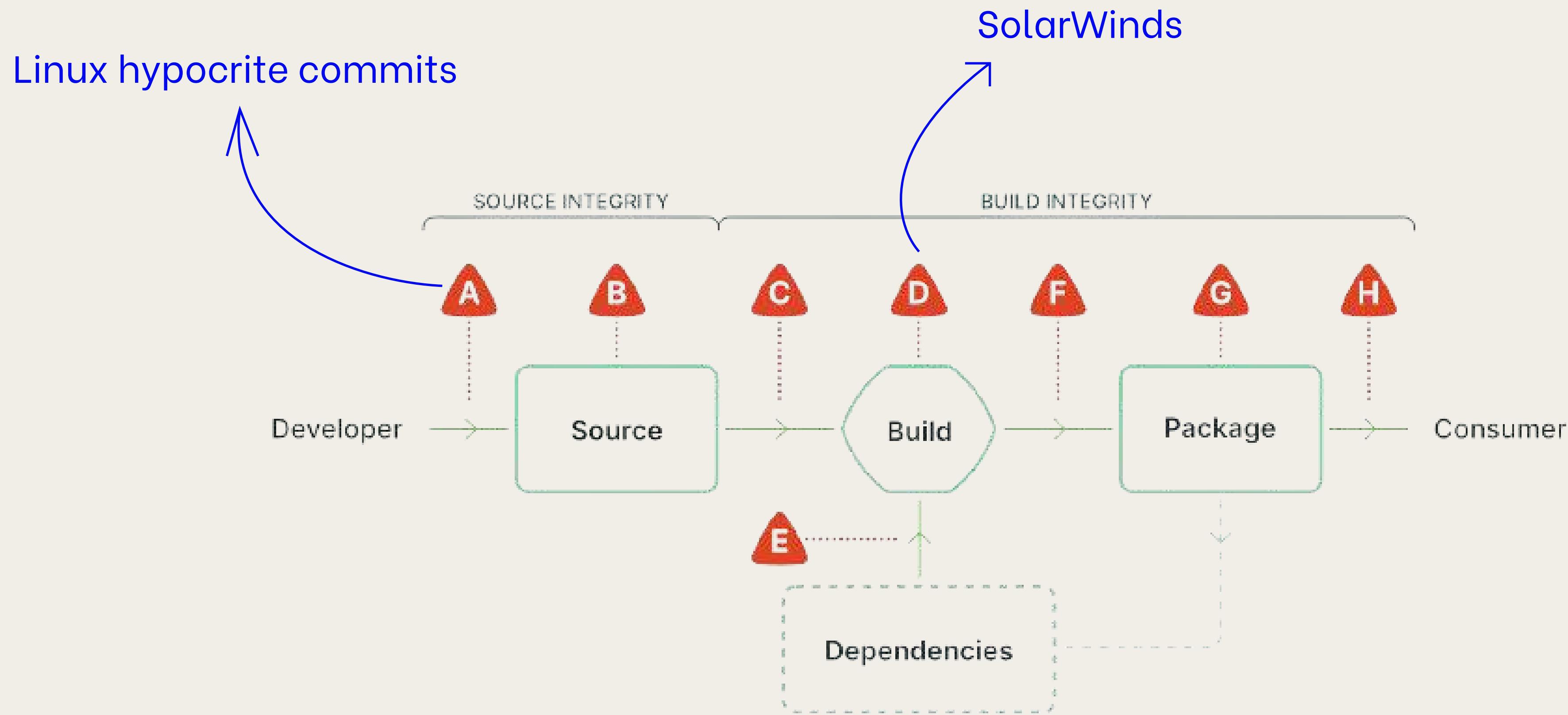


from slsa.dev

Linux hypocrite commits



from slsa.dev



from slsa.dev

3. Landscape and tools

We can't talk about securing the supply chain,
without understanding encryption

Assuming an understanding of encryption are leaving folks
out of the conversation



<https://a-very-brief-attempt-at-teaching-encryption.pain>

Encryption

Symmetric Encryption

Asymmetric Encryption

Signer

Verifier

Binaries A

Binaries A

encrypt w/
private key

decrypt w/
public key

Signature

Signature





Certificate

Company

Public key

Signature

Binaries A

Signer

Verifier



Binaries A

encrypt w/
private key

Binaries A

decrypt w/
public key

Signature

Signature

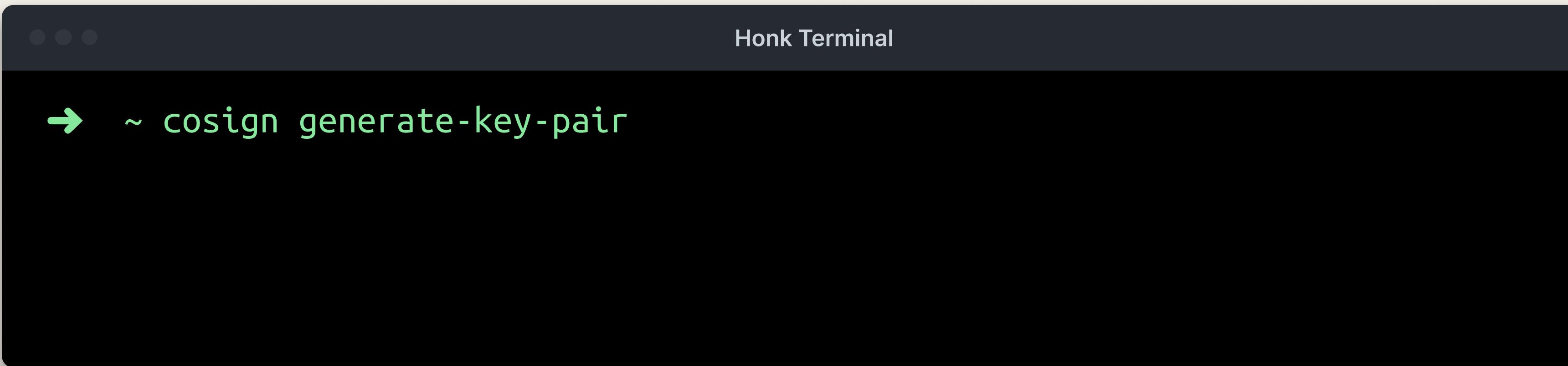


Sigstore

- cosign: sign and verify software artifacts
- fulcio: root certificate authority
- rekor: transparency logs

Cosign

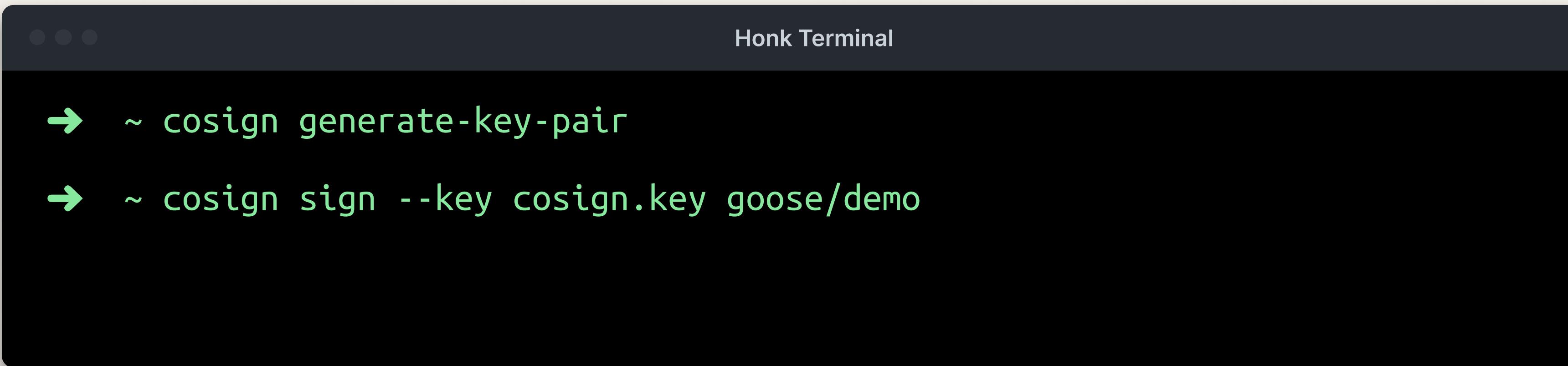
- Sign and verify containers, standard files, blobs



```
→ ~ cosign generate-key-pair
```

Cosign

- Sign and verify containers, standard files, blobs



A screenshot of a terminal window titled "Honk Terminal". The window has a dark background and contains two green terminal commands:

```
→ ~ cosign generate-key-pair
→ ~ cosign sign --key cosign.key goose/demo
```

Cosign

- Sign and verify containers, standard files, blobs

```
Honk Terminal

→ ~ cosign generate-key-pair
→ ~ cosign sign --key cosign.key goose/demo
→ ~ cosign verify --key cosign.pub goose/demo
```

public key!

Fulcio

- Certificate Authority (CA)
- Short-lived and based on email address

Certificate Request

OIDC ID Token

Public Key

Challenge

Rekor

- Tamper-proof log of metadata from the software supply chain
- Blockchain?

Honk Terminal

```
→ ~ cosign verify \  
  --certificate-identity=krel-trust@k8s-releng-  
prod.iam.gserviceaccount.com \  
  --certificate-oidc-issuer=https://accounts.google.com \  
  registry.k8s.io/kube-apiserver:v1.27.0 | jq .
```

Verification for registry.k8s.io/kube-apiserver:v1.27.0 --

The following checks were performed on each of these signatures:

- The cosign claims were validated
- Existence of the claims in the transparency log was verified offline
- The code-signing certificate was verified using trusted certificate authority certificates

[

{

 "critical": {

 "identity": {

 "docker-reference": "us-south1-docker.pkg.dev/k8s-artifacts-prod/
images/kube-apiserver"

 },

Honk Terminal

```
→ ~ crane digest registry.k8s.io/kube-apiserver:v1.27.0  
sha256:89b8d9dbef2b905b7d028ca8b7f79d35ebd9baa66b0a3ee2ddd4f3e0e2804b45
```

tool for interacting with remote
images and registries

hash of contents

Honk Terminal

```
→ ~ crane digest registry.k8s.io/kube-apiserver:v1.27.0  
sha256:89b8d9dbef2b905b7d028ca8b7f79d35ebd9baa66b0a3ee2ddd4f3e0e2804b45  
→ ~ crane manifest registry.k8s.io/kube-  
apiserver:sha256-89b8d9dbef2b905b7d028ca8b7f79d35ebd9baa66b0a3ee2ddd4f3e0e28  
04b45.sig | jq .
```

Honk Terminal

```
→ ~ crane digest registry.k8s.io/kube-apiserver:v1.27.0
sha256:89b8d9dbef2b905b7d028ca8b7f79d35ebd9baa66b0a3ee2ddd4f3e0e2804b45
→ ~ crane manifest registry.k8s.io/kube-
apiserver:sha256-89b8d9dbef2b905b7d028ca8b7f79d35ebd9baa66b0a3ee2ddd4f3e0e28
04b45.sig | jq .
{
  "schemaVersion": 2,
  "mediaType": "application/vnd.oci.image.manifest.v1+json",
  "config": {
    "mediaType": "application/vnd.oci.image.config.v1+json",
    "size": 352,
    "digest":
"sha256:594433956c7de81ac1ffc991f5b940e3ae6894c57ece37f88fe78df4165edde8"
  },
  "layers": [
    {
      "mediaType": "application/vnd.dev.cosign.simplesigning.v1+json",
      "size": 260,
      "digest":
"sha256:82df0875a2190ccfca2933954a1ecbddfe042aa4c3aebf6781d7359cbf414854",
```

(side notes uh oops [docs](#) are outdated for cosign 2.0, we're getting to it lol thanks)

The screenshot shows a web browser window with the Cosign documentation. The title bar has a blue icon and three horizontal lines. A note in a blue box says: "Note: To learn more about keyless signing, please refer to [Keyless Signatures](#)". The main content area has a large heading "Verifying image signatures". Below it, a note says: "For a complete list of images that are signed please refer to [Releases](#)". Another note says: "Let's pick one image from this list and verify its signature using the `cosign verify` command:". A code block shows the command: `COSIGN_EXPERIMENTAL=1 cosign verify registry.k8s.io/kube-apiserver-a`. A blue box at the bottom contains a note: "Note: `COSIGN_EXPERIMENTAL=1` is used to allow verification of images signed in `KEYLESS` mode. To learn more about keyless signing, please refer to [Keyless Signatures](#) ."

Note: To learn more about keyless signing, please refer to [Keyless Signatures](#).

Verifying image signatures

For a complete list of images that are signed please refer to [Releases](#).

Let's pick one image from this list and verify its signature using the `cosign verify` command:

```
COSIGN_EXPERIMENTAL=1 cosign verify registry.k8s.io/kube-apiserver-a
```

Note: `COSIGN_EXPERIMENTAL=1` is used to allow verification of images signed in `KEYLESS` mode. To learn more about keyless signing, please refer to [Keyless Signatures](#) .

The screenshot shows a web browser window with the URL `search.sigstore.dev` in the address bar. The page title is "Rekor Search". On the left, there is a logo for "sigstore rekord" featuring a purple circular icon with a white camera-like shape and the word "rekord" in bold purple letters. To the right of the title are two icons: a gear and a GitHub logo.

Below the title, there are two input fields: "Attribute" set to "Log Index" and "Log Index" set to "17710987". A large blue "SEARCH" button is centered between them.

Underneath the search form, the text "Showing 1 of 1" is displayed. Below this, an entry is shown with the following details:

Entry UUID: 24296fb24b8ad77a943e8e9b1390053f3c96b5b0ecfda4bf3d4c9f5ec8ff2cd37603e98e0ccc6e97		
TYPE	LOG INDEX	INTEGRATED TIME
hashedrekord	17710987	16 hours ago (2023-04-11T11:41:36-07:00)

Below the table, under the heading "Hash", is the value `sha256:b6efabb74392a717fc...79`. Under the heading "Signature", is the value `MEQCIDtwoim5tF/eXWwgEF0qwGF9j/7x0EsD4Edtb1eU0cfJAiBjtPI5/KeYN/L+96zCM4t0GLdWhsWvhx60X5yPy`.

Sigstore Adoption

The screenshot shows the README.md file for the `sigstore-python` repository. It includes a CI status badge (CI passing), a PyPI package version badge (1.1.1), an OpenSSF Scorecard badge (7.8), and an SLSA level 3 badge. Below these badges, there are links for Conformance Tests (passing) and Documentation (passing). The main text describes `sigstore` as a Python tool for generating and verifying Sigstore signatures, mentioning its use for signing and verifying Python package distributions.

Index

- Features
- Installation
 - GitHub Actions
- Usage
 - Signing
 - Verifying
 - Generic identities
 - Signatures from GitHub Actions
- Example uses
 - Signing with ambient credentials

The screenshot shows a GitHub page for an RFC titled "Link npm packages to the originating source code repository and build". The page has a summary section explaining the goal of linking public npm packages to their source code repositories to prevent package hijacking. It also includes a motivation section and a link to a glossary. The page is authored by @feelepxyz, @kommendorkapten, and @trevrosen.

Link npm packages to the originating source code repository and build

Authors: @feelepxyz @kommendorkapten @trevrosen

Summary

Link public npm packages to the source code repository and build it originated from, making it harder to execute package hijacking attacks where a malicious version of an existing open source package gets uploaded to the registry.

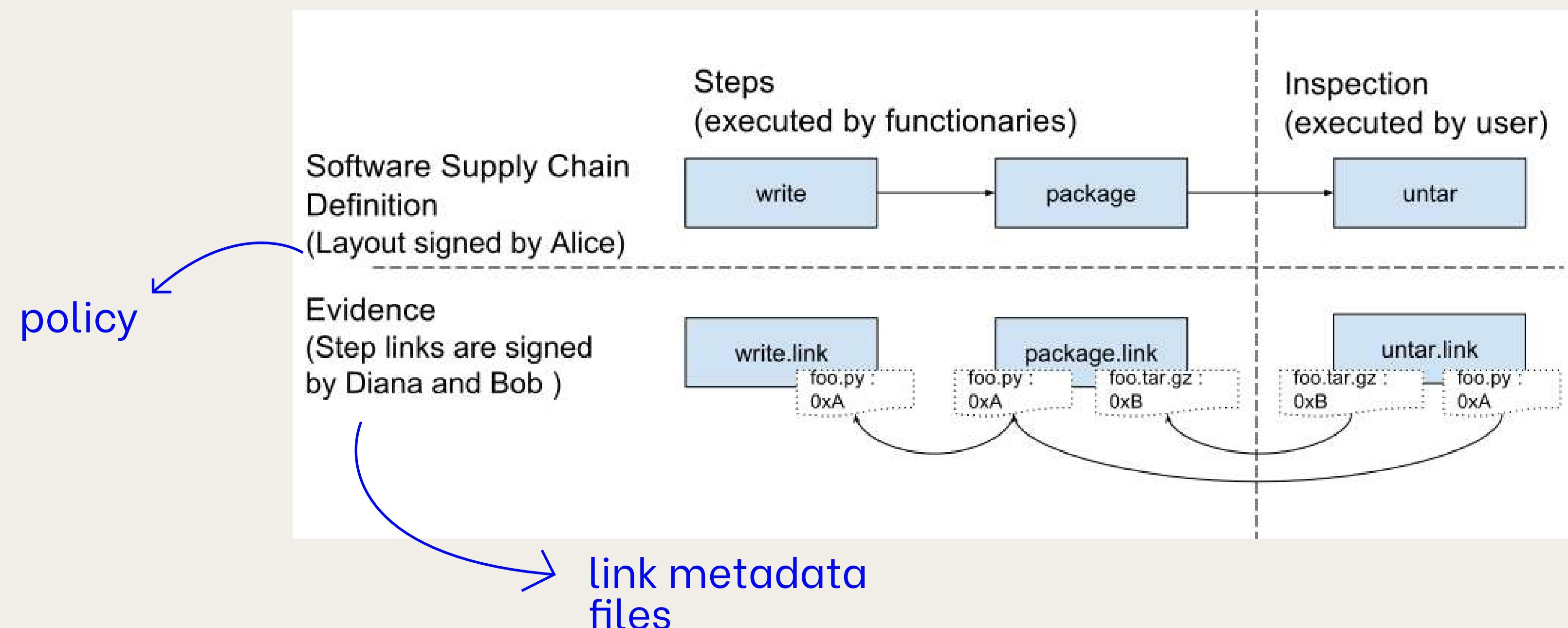
Check out the [glossary](#) for an overview of the different tools, terms and techniques covered in this RFC document.

Motivation

So far our supply chain security offerings have focused on remediation - patching known vulnerabilities in dependencies (e.g. `npm audit`). The majority of vulnerabilities are accidental, so that was the right place to start. Now we need to turn our attention to deliberate supply chain

In-toto

- Record what steps were performed, by whom and in what order
- Verify steps and its creator permissions



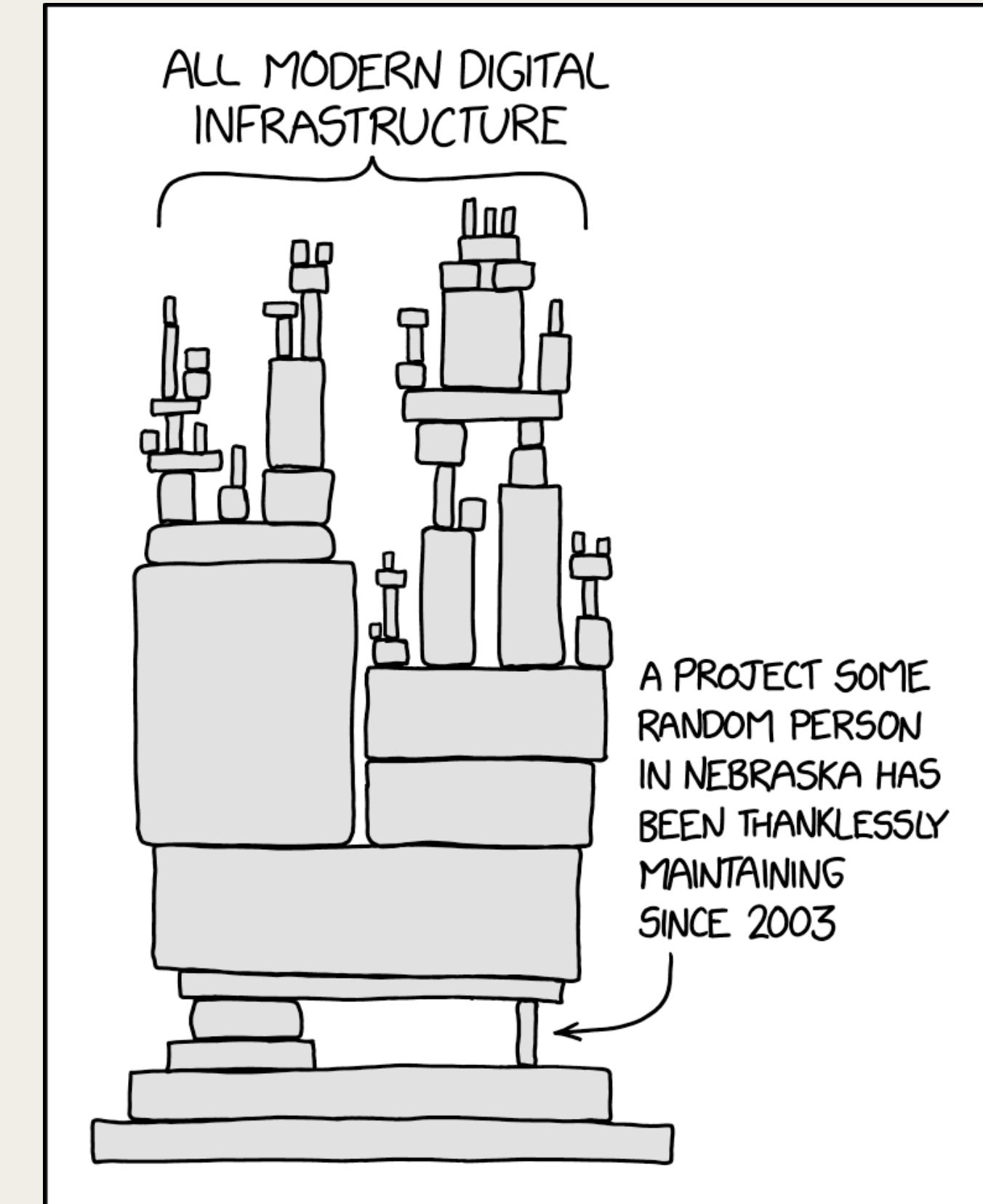
TUF

- Last mile – package distribution
- **Roles:**
 - root: specify other roles
 - targets: indicate packages validity
 - snapshot: provide package versions
 - timestamp: when package last updated
- Revocation + security compared to sigstore

4. Misconceptions & Discussion

1. It's just about the code, right?
2. We're probably not a good target
3. We can fully remove supply chain risks in our software

It's a shared responsibility!



<https://xkcd.com/2347/>

What else? What's next?

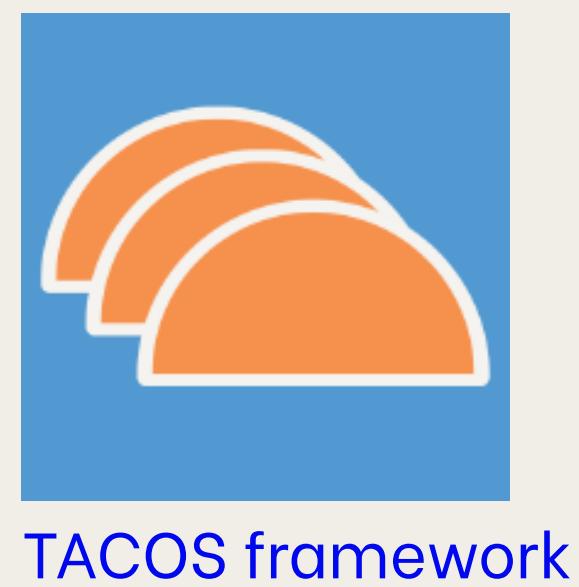
- Progress and new standards are being made *right now*
- Other components: vuln scanning, securing build environments...

What else? What's next?

- Progress and new standards are being made *right now*
- Other components: vuln scanning, securing build environments...



GUAC



TACOS framework



! Warning

The `docker sbom` command is currently in beta releases.

GitHub Code Security

Generate, export and share SBOM

Supply chain @ Kubecon

- Wed @ 15:25: [From SBOMs to IBOMs - Know What's Happening in Your Clusters](#)
- Thurs @ 15:25: [In-Toto: Attestations and More for Software Supply Chain Security](#)
- Thurs @ 15:25: [Checking the Chains at the Gate: Building Supply Chain Policies with Gatekeeper and Ratify](#)
- Thurs @ 16:30: [Secure Your Project with the SIG Release Supply Chain Kit](#)
- Fri @ 14:55: [Building SLSA 3 Conformance Attestors for Artifacts Generated on GitHub](#)

What on earth is software supply chain security?



That's a wrap

Twitter @ [gracennng](#)

Hire me for security new-grad roles
@ May 2024 lol pls

[https://www.youtube.com/watch?
v=wwtL9iM_rg&t=327s&ab_channel=CNCF%5BCloudNativeComputingFoundation%
5D](https://www.youtube.com/watch?v=wwtL9iM_rg&t=327s&ab_channel=CNCF%5BCloudNativeComputingFoundation%5D)

<https://blog.sigstore.dev/sigstore-blockchain-vs-transparency-logs-d673ea41a9be/>

[https://docs.google.com/spreadsheets/
d/1CzvnlnT7QOmTOz20W5TiX8tJiG9XZvdqYA3TivLx-PI/edit#gid=0
asdfg](https://docs.google.com/spreadsheets/d/1CzvnlnT7QOmTOz20W5TiX8tJiG9XZvdqYA3TivLx-PI/edit#gid=0)

<https://www.sonatype.com/state-of-the-software-supply-chain/introduction>

[https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/
executive-order-on-improving-the-nations-cybersecurity/](https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/)

[https://docs.google.com/presentation/d/1MFNUVBas_PXT9AbrLIUOtqWPrtB-
AJSI6EMncvjdtoE/edit#slide=id.g1251d519bb9_0_13](https://docs.google.com/presentation/d/1MFNUVBas_PXT9AbrLIUOtqWPrtB-AJSI6EMncvjdtoE/edit#slide=id.g1251d519bb9_0_13)