Introduction to Sigstore: cryptographic signatures made easier

PyCon France 2023

Maya Costantini

Software Engineer, Red Hat

$whoami

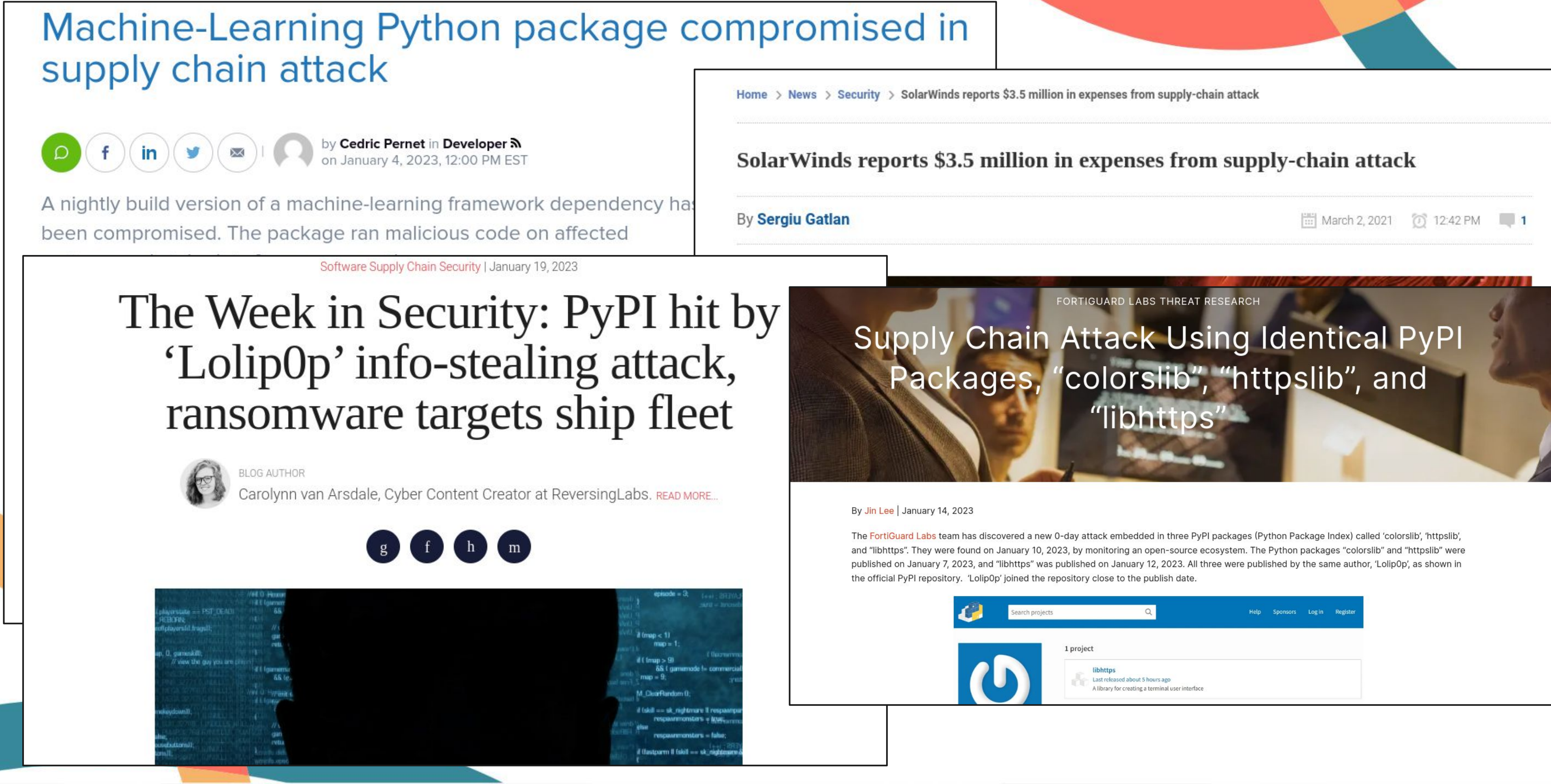
@MayaCostantini

hachyderm.io/@mayacostantini

@mayaCostantini

Software Engineer, Red Hat

Passionate about Python & Open Source contributor



Machine-Learning Python package compromised in supply chain attack

by **Cedric Pernet** in **Developer**
on January 4, 2023, 12:00 PM EST

A nightly build version of a machine-learning framework dependency has been compromised. The package ran malicious code on affected

Home > News > Security > SolarWinds reports $3.5 million in expenses from supply-chain attack

SolarWinds reports $3.5 million in expenses from supply-chain attack

By **Sergiu Gatlan** March 2, 2021 12:42 PM 1

Software Supply Chain Security | January 19, 2023

The Week in Security: PyPI hit by 'Lolip0p' info-stealing attack, ransomware targets ship fleet

BLOG AUTHOR

Carolynn van Arsdale, Cyber Content Creator at ReversingLabs. READ MORE...

FORTIGUARD LABS THREAT RESEARCH

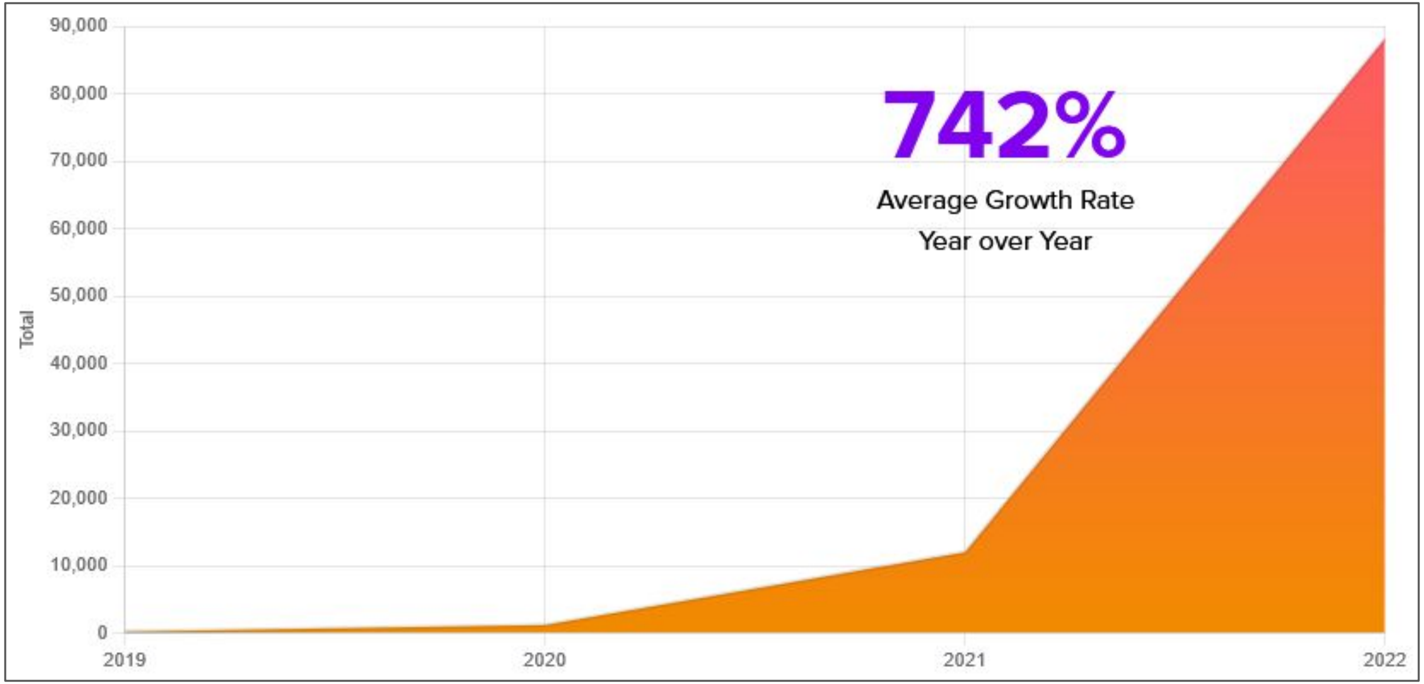
Supply Chain Attack Using Identical PyPI Packages, "colorslib", "httpslib", and "libhttps"

By Jin Lee | January 14, 2023

The FortiGuard Labs team has discovered a new 0-day attack embedded in three PyPI packages (Python Package Index) called 'colorslib', 'httpslib', and "libhttps". They were found on January 10, 2023, by monitoring an open-source ecosystem. The Python packages "colorslib" and "httpslib" were published on January 7, 2023, and "libhttps" was published on January 12, 2023. All three were published by the same author, 'Lolip0p', as shown in the official PyPI repository. 'Lolip0p' joined the repository close to the publish date.

Search projects
Help
Sponsors
Log in
Register
1 project
libhttps
Last released about 5 hours ago
A library for creating a terminal user interface

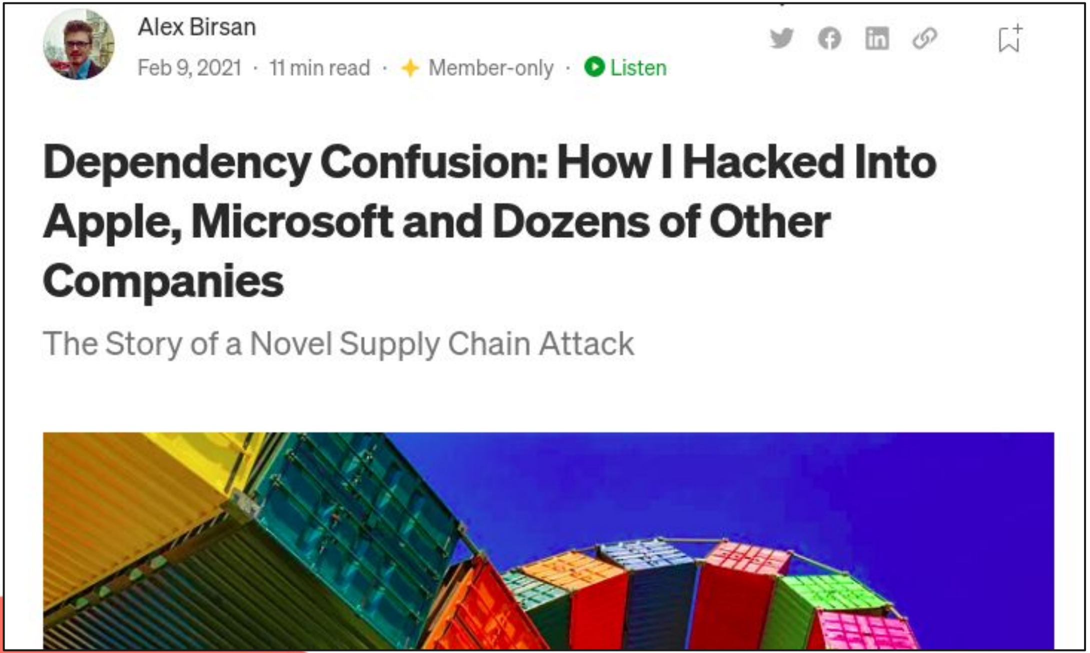
Increase in Software Supply Chain attacks (2019-2022)



Source: https://securityboulevard.com

Why are signatures important?

Attackers play on developer expectations of systematic build reproducibility to find vulnerable links in a Software Supply Chain



Alex Birsan

Feb 9, 2021 · 11 min read · ✦ Member-only · Listen

Dependency Confusion: How I Hacked Into Apple, Microsoft and Dozens of Other Companies

The Story of a Novel Supply Chain Attack

What code signing can do about it:

- Ensure the **authenticity** and **integrity** of software
- Cryptographically bind an artifact to its authors public identities

GPG: an outdated signing standard?

OpenPGP (RFC4880) is considered as the current standard for signing software. But it presents some important disadvantages:

- Complex to master and administrate
- User confusion about key creation and exchange schemes
- Compatibility issues between versions
- Problems of keys storage, distribution and revocation

What software maintainers and consumers need

- Sign different kinds of artifacts at scale
- Ease of use: no need for a professional knowledge of cryptographic schemes or of Public Key Infrastructure standards
- Avoid risks linked to private key compromise
- A strong tie between a project signature and its maintainer identities
- Easier auditing in case of compromise

Project Sigstore

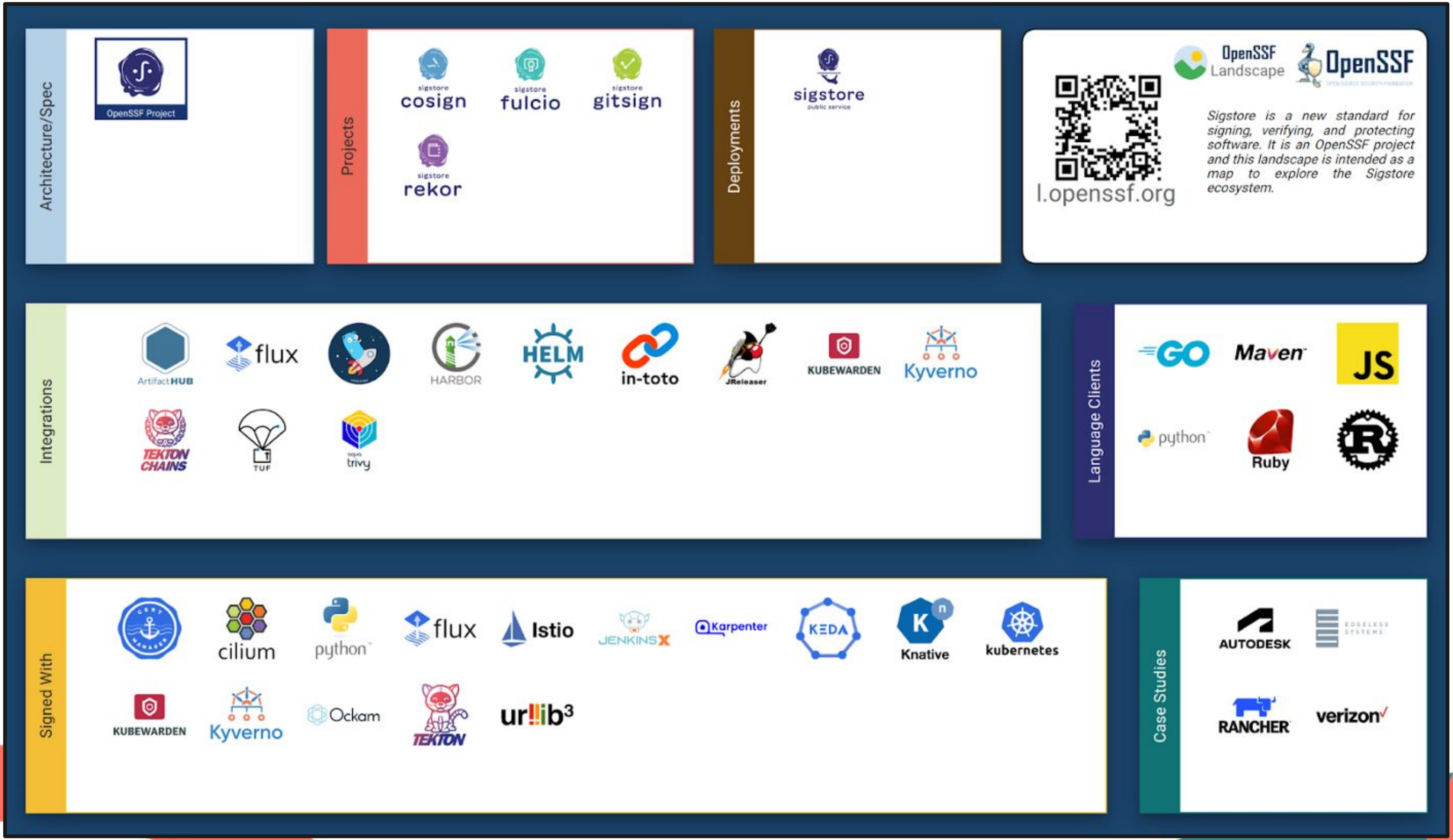
A new standard for signing, verifying and protecting software

“ Become to cryptographic signatures what Let’s Encrypt is to HTTPS ”

Securely sign and verify OCI-compliant artifacts (container images, binaries, files, SBOMs…)

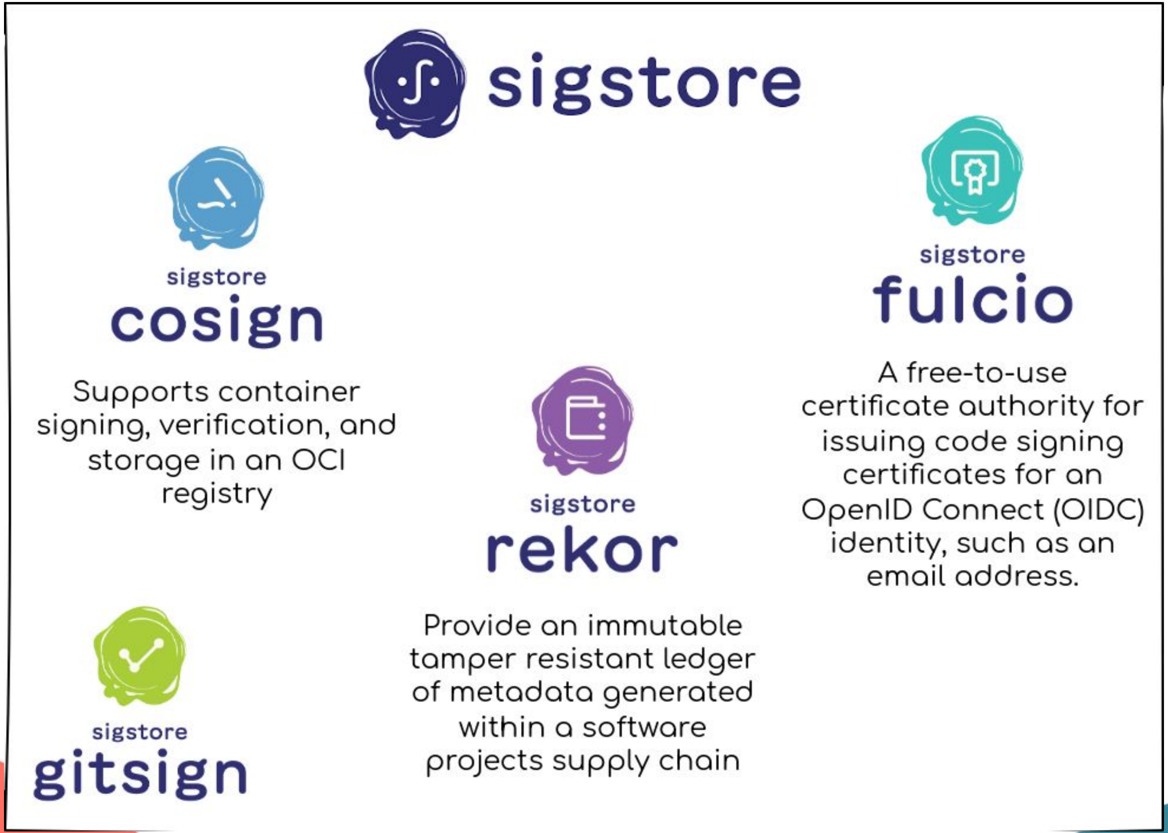
Store the signing materials in a tamper-resistant public Transparency Log for audit and verification

Sigstore landscape and open source adoption



Sigstore is a new standard for signing, verifying, and protecting software. It is an OpenSSF project and this landscape is intended as a map to explore the Sigstore ecosystem.

How does it work? Sigstore subprojects



sigstore

sigstore **cosign**

Supports container signing, verification, and storage in an OCI registry

sigstore **fulcio**

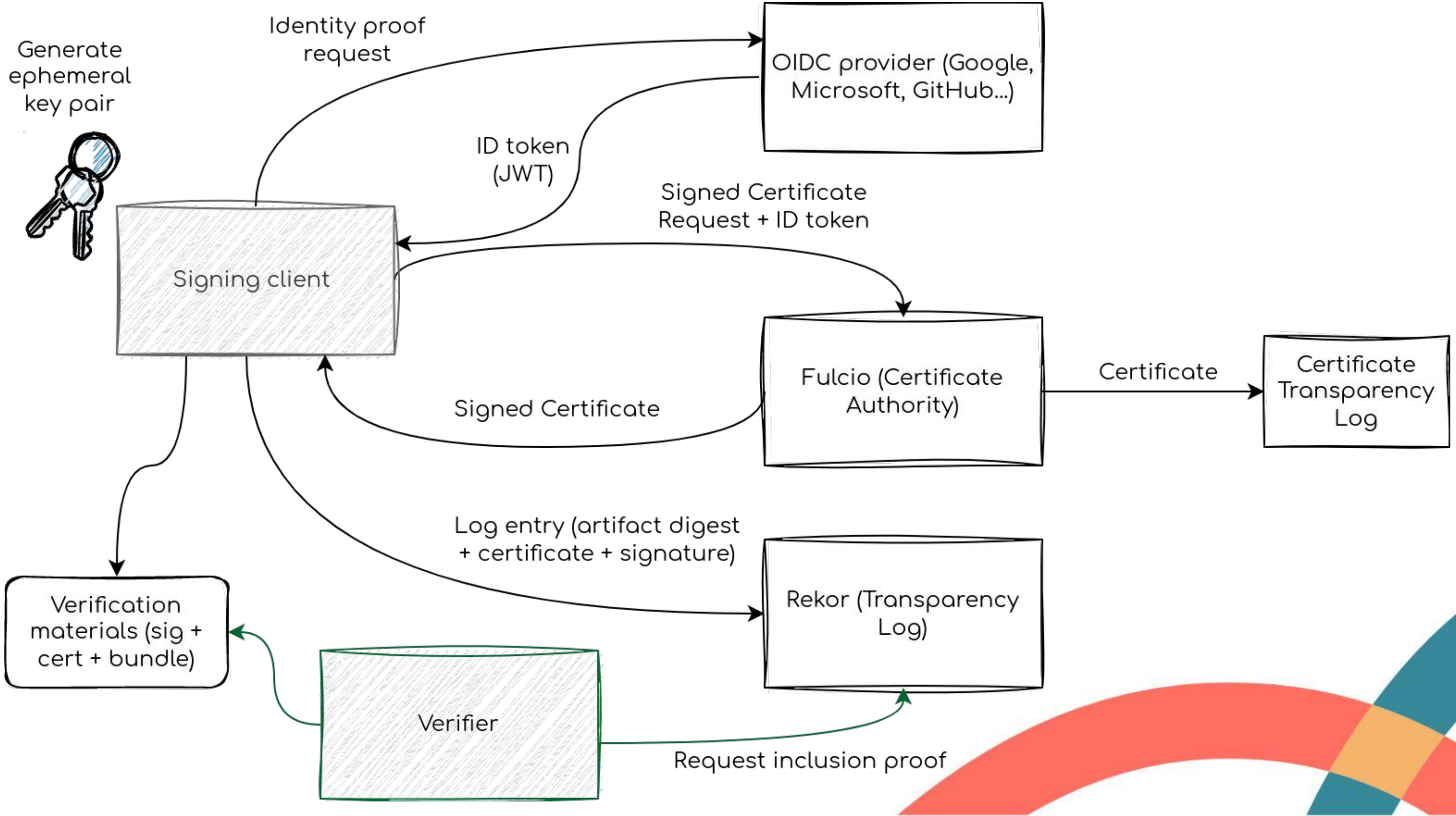
A free-to-use certificate authority for issuing code signing certificates for an OpenID Connect (OIDC) identity, such as an email address.

sigstore **rekor**

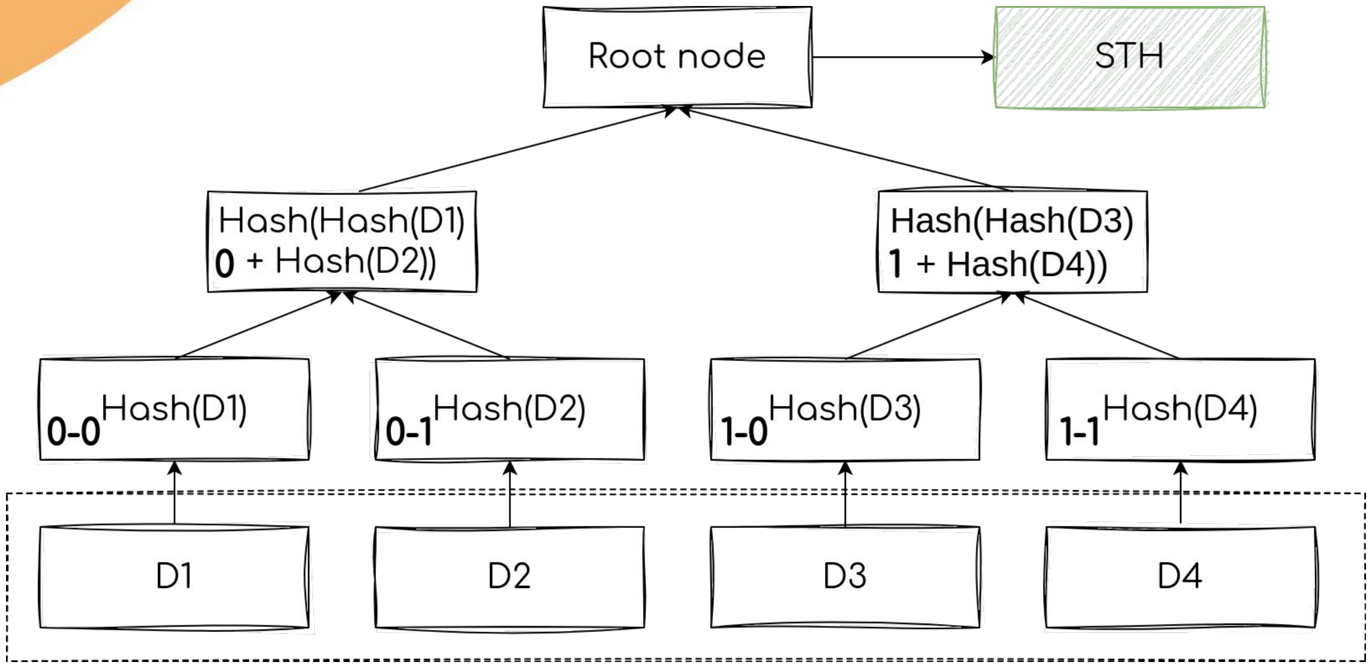
Provide an immutable tamper resistant ledger of metadata generated within a software projects supply chain

sigstore **gitsign**

How does it work? Sigstore’s keyless signing flow


Generate ephemeral key pair
Identity proof request
OIDC provider (Google, Microsoft, GitHub...)
ID token (JWT)
Signing client
Signed Certificate Request + ID token
Fulcio (Certificate Authority)
Certificate
Certificate Transparency Log
Signed Certificate
Log entry (artifact digest + certificate + signature)
Rekor (Transparency Log)
Verification materials (sig + cert + bundle)
Verifier
Request inclusion proof

How does it work? The Sigstore Trust model



Rekor’s Transparency log is backed by a **Merkle Tree**

It is **immutable**, **append-only,** and **cryptographically verifiable**

The log is monitored to detect eventual inconsistencies

How does it work? The Sigstore Trust model

- Fulcio also makes use of an immutable, append-only and cryptography verifiable Certificate Transparency Log to store signing certificates

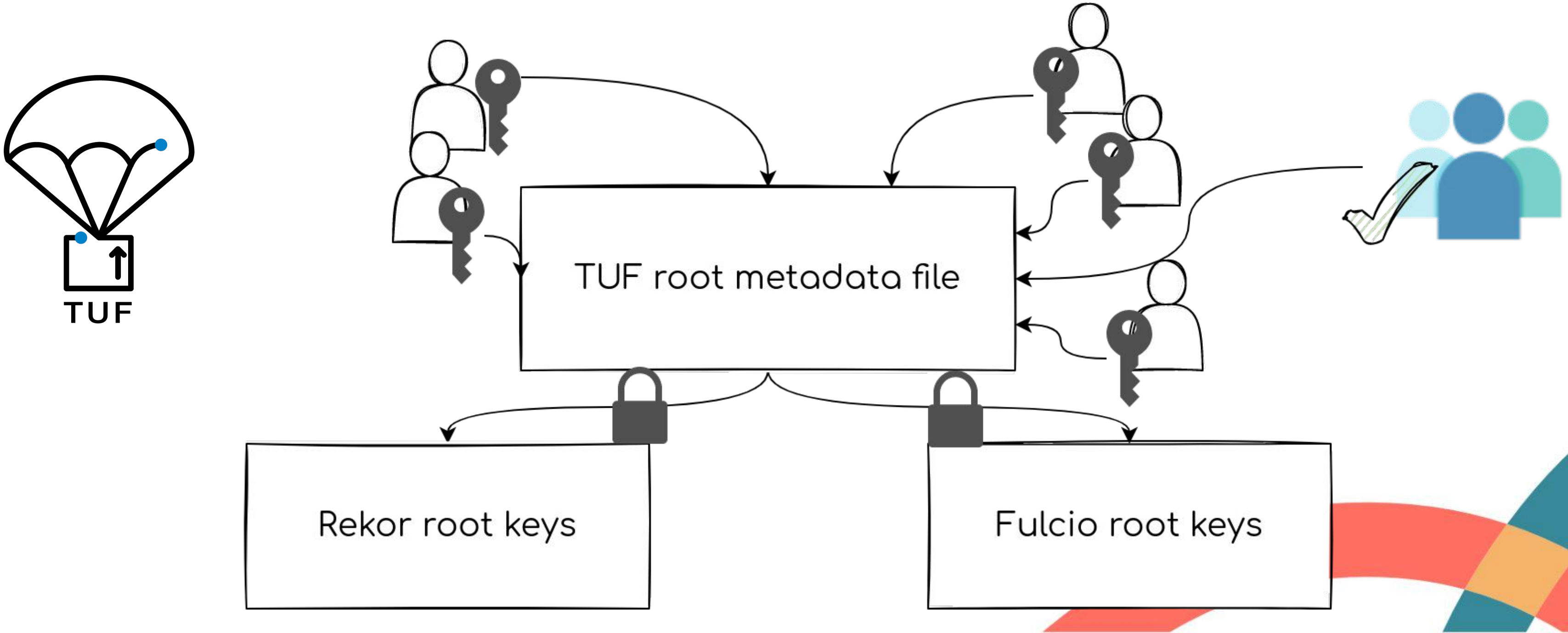
- Certificates issued by Fulcio are **ephemeral**: users can verify they were valid during the time when the artifact was signed

- Short-lived certificates avoid revocation and facilitate **auditability** instead

How does it work? The Sigstore Trust model

Establishing the Sigstore Trust Root

Sigstore root signing ceremony



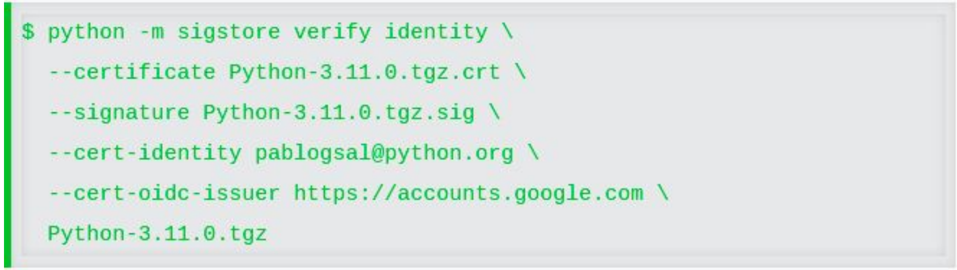
Sigstore in the Python ecosystem

A Python language client: sigstore-python to integrate Sigstore into your Python project

- Sign files and blobs from the command line using a “keyless” workflow, interactively or with ambient credentials
- Sign artifacts in a GitHub CI workflow with the sigstore-python GitHub Action
- Integrate sigstore-python natively into a Python project using the library public API, stable since v1.0.0

Sigstore adoption by the Python community

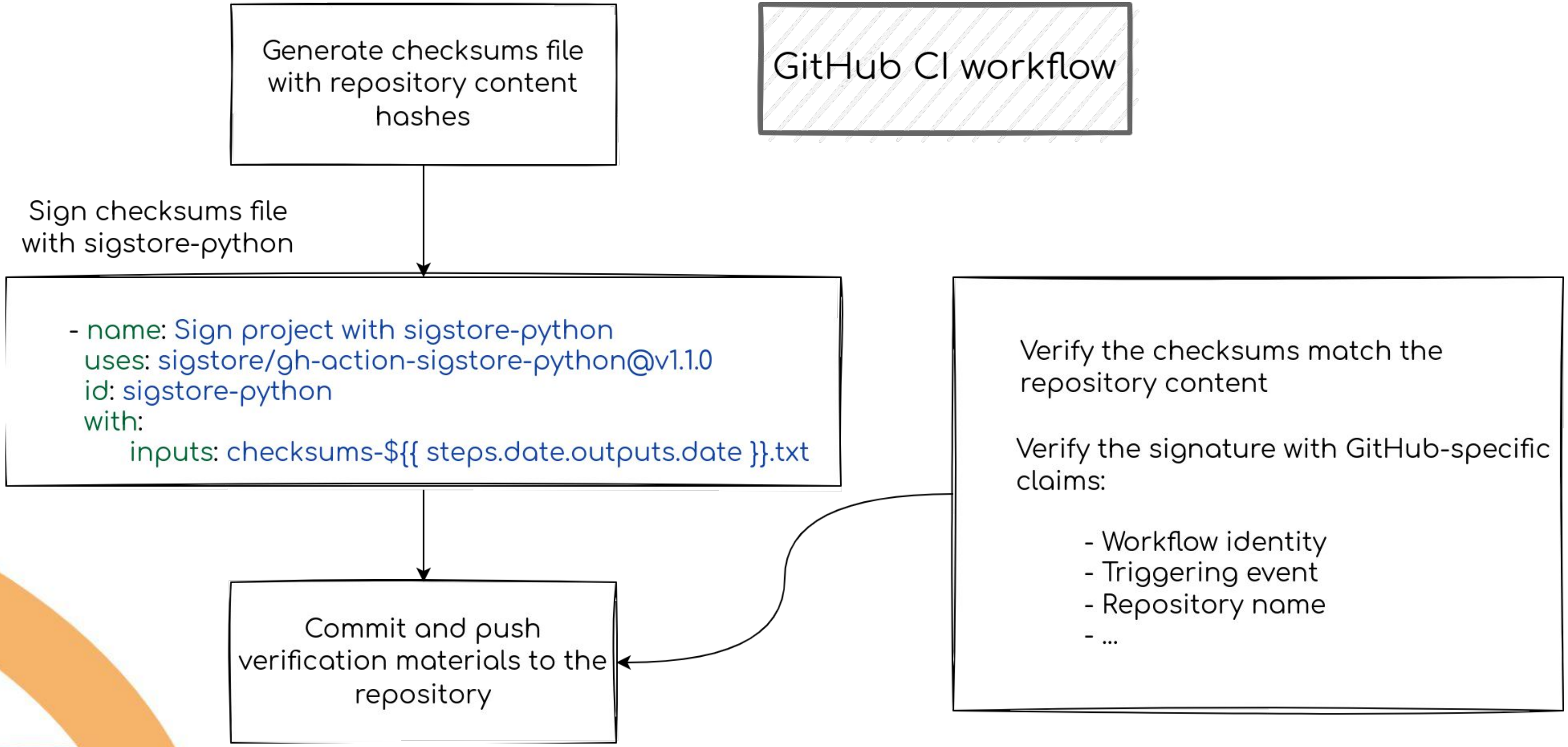
Sigstore’s Python client is now used to sign releases of CPython



```
$ python -m sigstore verify identity \
  --certificate Python-3.11.0.tgz.crt \
  --signature Python-3.11.0.tgz.sig \
  --cert-identity pablogsal@python.org \
  --cert-oidc-issuer https://accounts.google.com \
  Python-3.11.0.tgz
```

python.org/download/sigstore/

Demo: sign your project with the sigstore-python GitHub Action



GitHub CI workflow

Generate checksums file with repository content hashes

Sign checksums file with sigstore-python

```yaml
- name: Sign project with sigstore-python
  uses: sigstore/gh-action-sigstore-python@v1.1.0
  id: sigstore-python
  with:
      inputs: checksums-${{ steps.date.outputs.date }}.txt
```

Commit and push verification materials to the repository

Verify the checksums match the repository content

Verify the signature with GitHub-specific claims:

- Workflow identity
- Triggering event
- Repository name
- ...

Demo repository: mayaCostantini/pyconfr-sigstore-demo

Join the Sigstore community and get involved

sigstore.dev/community

https://links.sigstore.dev/slack-invite

Sigstore YouTube channel

sigstore Blog https://blog.sigstore.dev/

Thank you! Questions?

Find the slides for this talk: