

Welcome!

Anthony Harrison and **Olle E. Johansson**
SBOM Europe

Who we are

Anthony Harrison APH10, UK

- 40 years delivering mission critical solutions across multiple sectors
- Founder and Director APH10
- Co-founder SBOMEurope
- Open Source Software developer
- Mentor
- Lives outside Manchester, UK
- Runner/Hiking



Olle E. Johansson "oej", Edvina, SE

- Open Source developer, consultant, IETF activist and much more
- Former asterisk.org developer
- Active in OWASP CycloneDX, OpenSSF, ORCWG, IETF and more
- Lives in Sollentuna, North of Stockholm, Sweden
- Co-founder SBOMEurope
- Storyteller and gardener



What's the idea?

Building a forum for
knowledge-sharing, discussions
and standard feedback.
Looking at tools, standards,
implementations, use-cases
and more.



Agenda Thursday Sep 3rd

- **Introduction and welcome**
- **Updates**
- **Presentation: SBOM and digital signatures**
- **Open discussion**

Sign your SBOM... With WHAT?

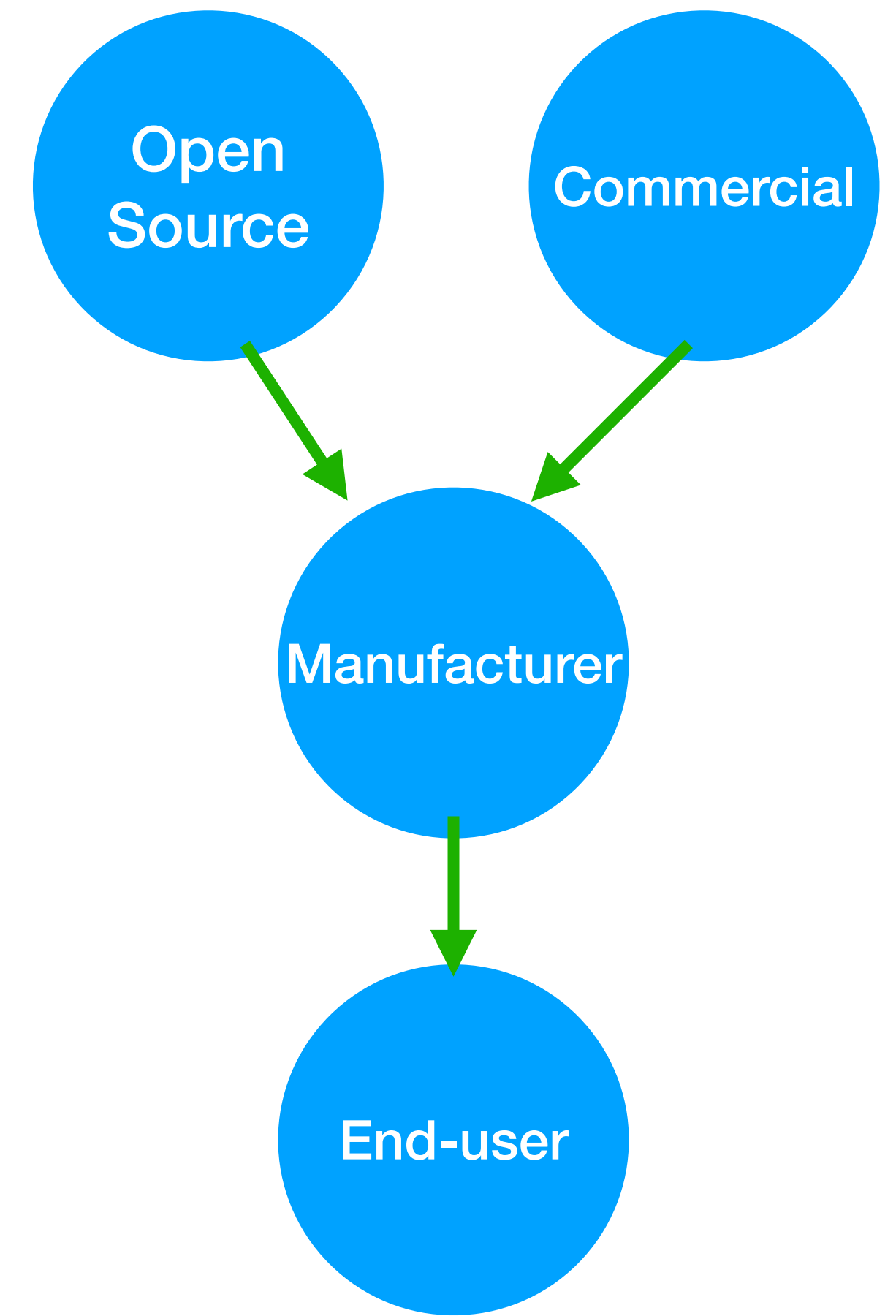
Trusting the software supply chain
2026-09-03 v3 oej

@oej@infosec.exchange



The problem as I see it

- Many customers need to trust many vendors, much like web trust
- But web trust is broken
- Commercial software signing certs cost too much
- Managing private keys for a long time is complex
- **As the SBOM is exchanged across the software supply chain, we need to implement trust and transparency.**



Aaaaaaaargh



OWASP SCVS talks
about signing SBOMs,
validating

...but not about how and
with what...

ENISA SBOM landscape
(review)
talks about digital
signatures for SBOMs
and trust...

But not how and with
what.

OWASP TEA
has some ideas
published by OEJ

...but not a lot of
feedback....

<https://scvs.owasp.org/>

<https://github.com/cyclonedx/transparency-exchange-api>

HELP ME!

Looking for a solution



SIGN

VALIDATE

TRUST

- We need to be able to sign, validate and trust digital signatures on documents from Open Source projects as well as commercial vendors
- How do we build such a trust system without a single point of failure, without a central actor?

Transparency is important



- *“How do I trust that the vendor did not change the SBOM without telling me?”*
- Someone needs to monitor the actors, much like certificate transparency
- It's not clear who that actor is

A global problem

Business
manufacturers

Open Source
projects and
stewards



Why not use WebPKI?

- The trust chain is ... something to discuss
- The cost of digital signing certs is very high
- The cost of maintaining private key in HSM is high
- Why give these companies more monopoly?

Sigstore is cool

- Based on another ID, like OpenID connect auth, creates a **temporary key pair and a certificate**
- This key pair is used for signing - a commit, container or something else
- Proof is stored in a signed log for transparency and verification
- The certificate expires within minutes and the key pair is no longer valid, can be deleted and forgotten
- *Signing with key and cert, but almost no key management needed*



...but Sigstore is not a solution for everyone

- Sigstore (the service) works well for some Open Source projects
 - *Integration to Github and more*
- But it's not a solution for everyone
 - *Privacy - stores email addresses in US cloud*
 - *Won't be accepted by commercial actors*
- The Open Source software is hard to install and put in production

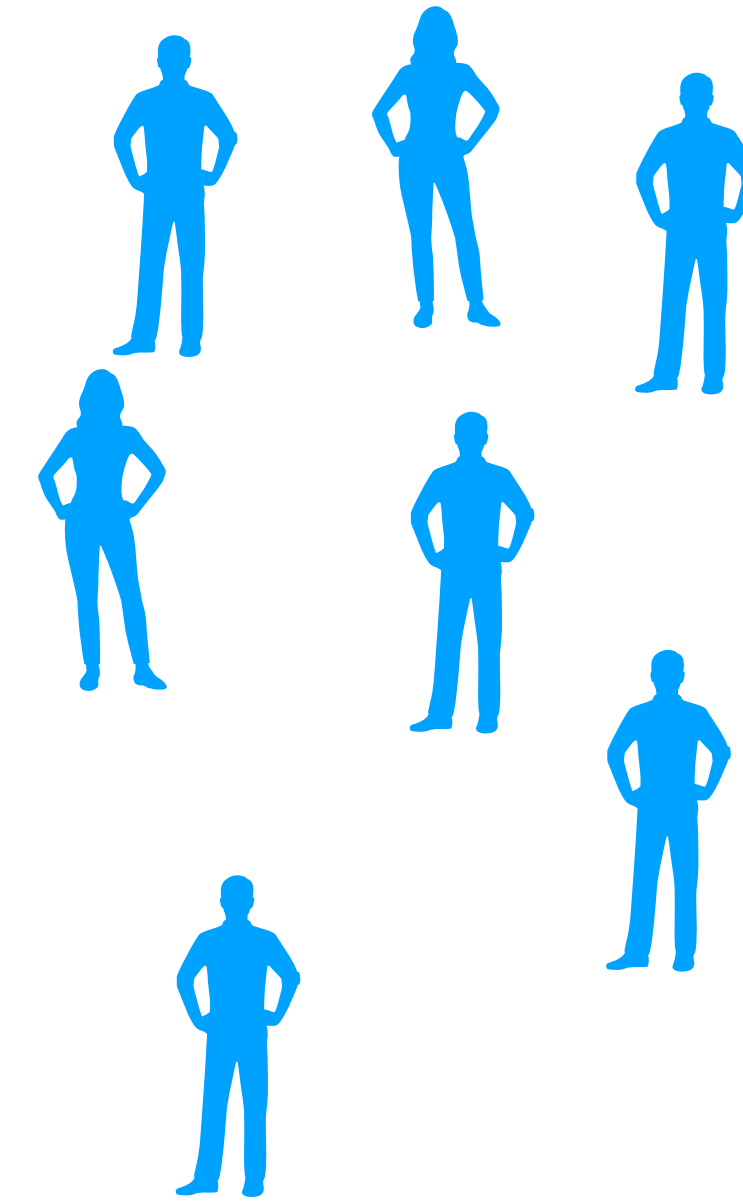
Letsencrypt turned off signing

- A free PKI for Web certificates
 - enabling TLS everywhere at no cost
- Killing business for a lot of commercial Certificate Authorities
- Funded by large companies
- Only US based system
- Turned off signing in key usage bits



Open Source use PGP/GPG

- A web of trust which is not really used
 - *Very few keys are signed by trusted parties*
- A single key pair with limited trust anchors
- Best current practise is not well documented
 - *“Download this file and add blindly to your key store as a trusted key”*
 - *NO FINGERPRINT GIVEN*
- Technology is not the primary problem



FOSDEM used to have
huge key signing events

Managing a private PKI is hard

- Protecting a private key from abuse is hard
- If it's stored on a HSM - it's a physical device to protect
- Backup is needed, but needs protection
- Needs some kind of policy in an organisation



Trust is earned

Trust on first use

SSH

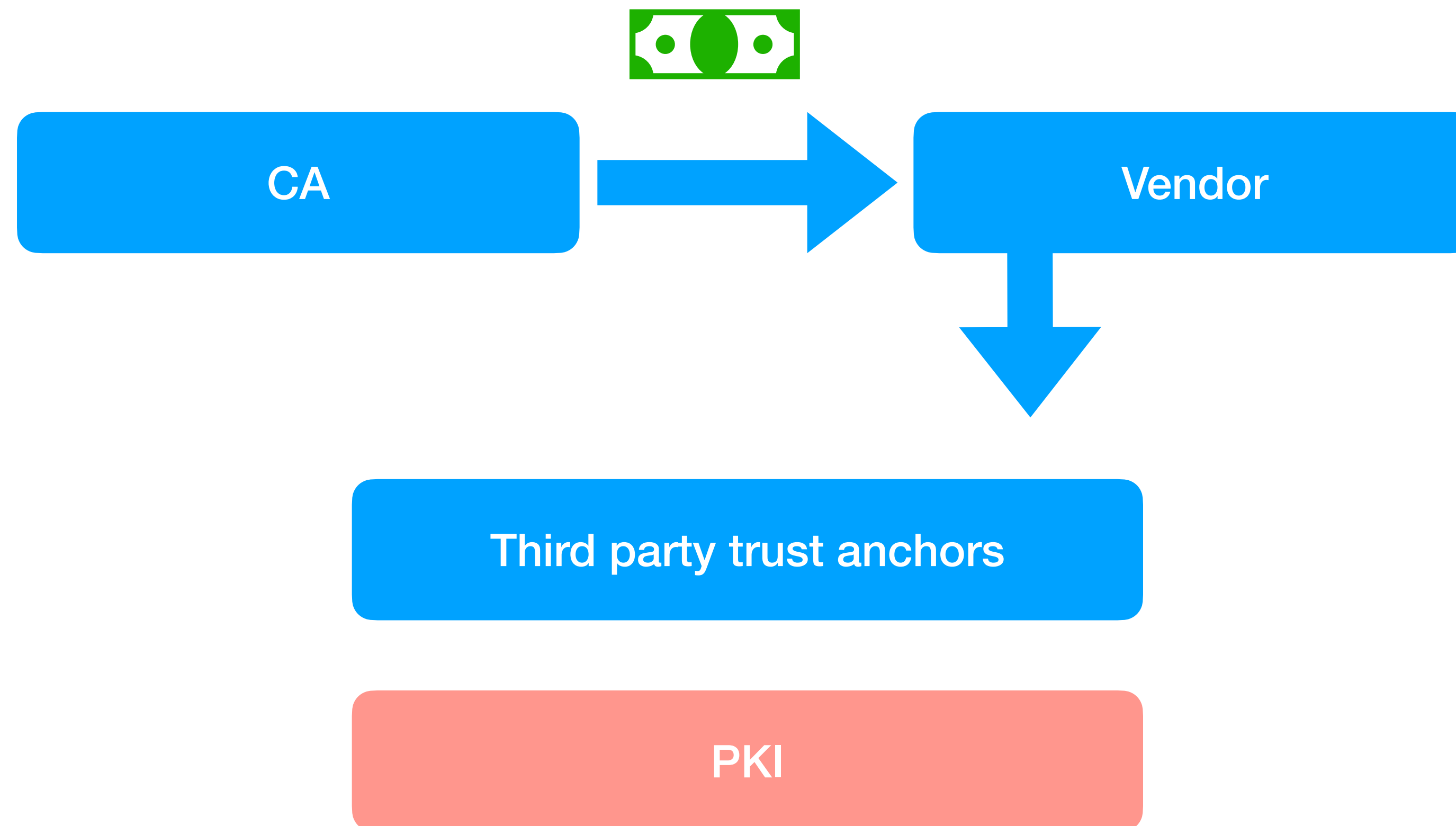
Third party trust anchor

PKI

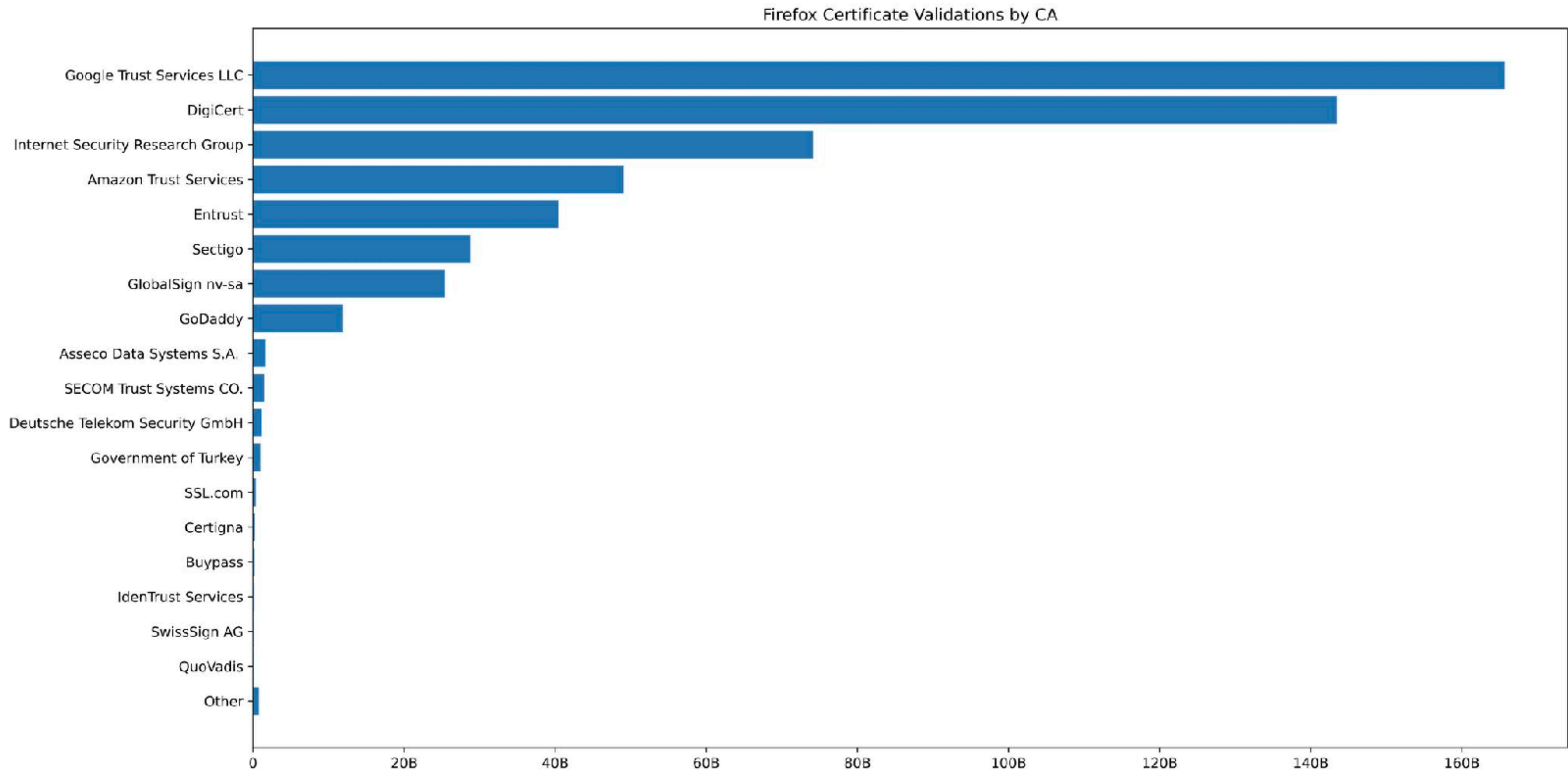
Web of trust

GPG

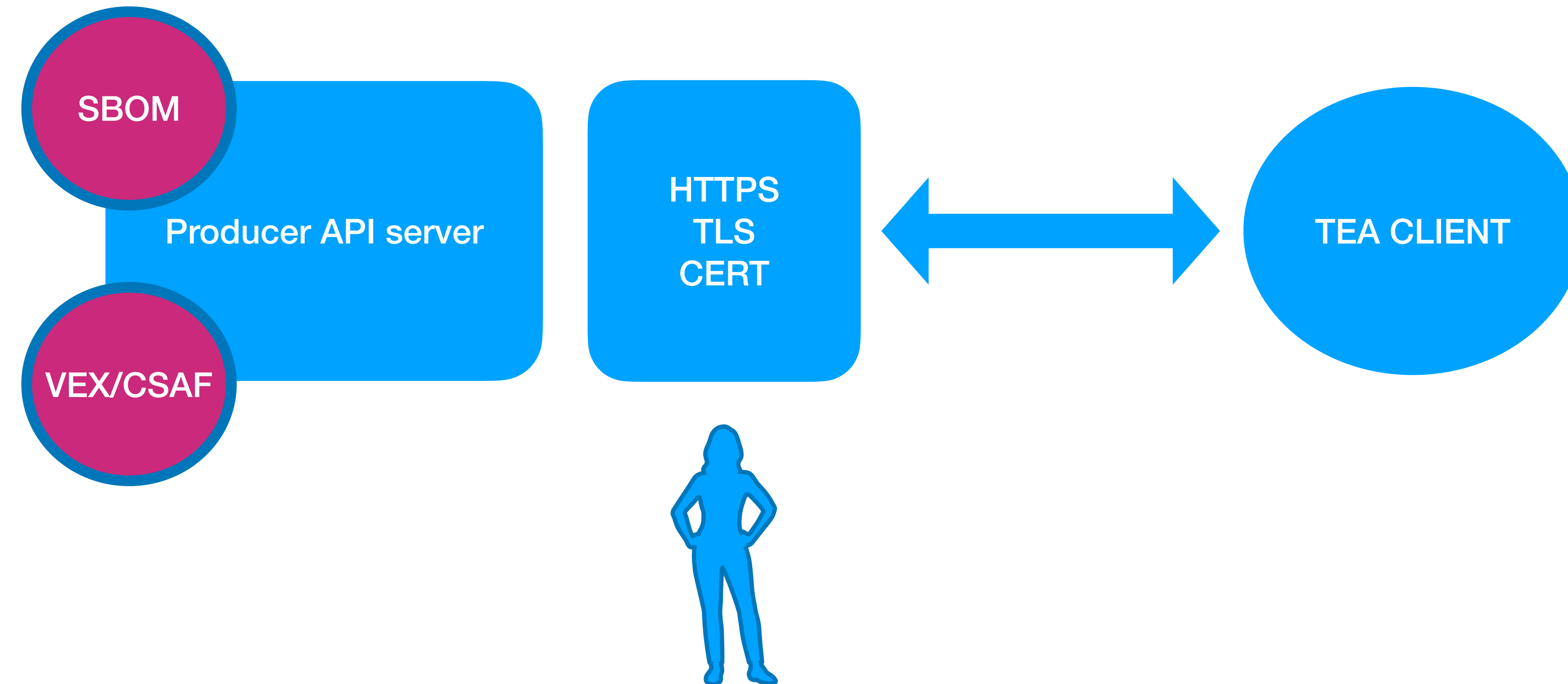
Trust is earned -
but in the web it is forced upon us



The largest CAs - did you know?

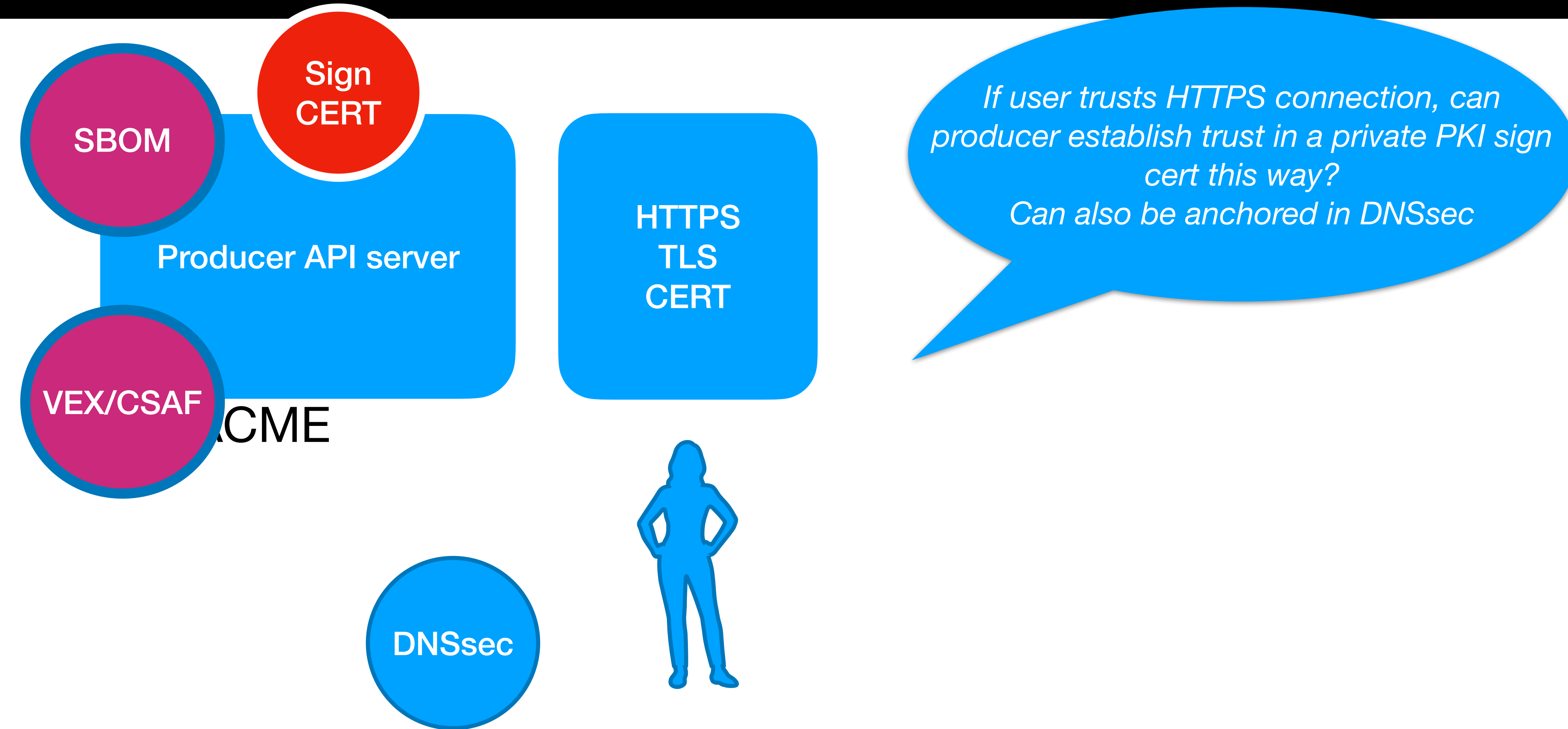


OWASP Transparency Exchange API



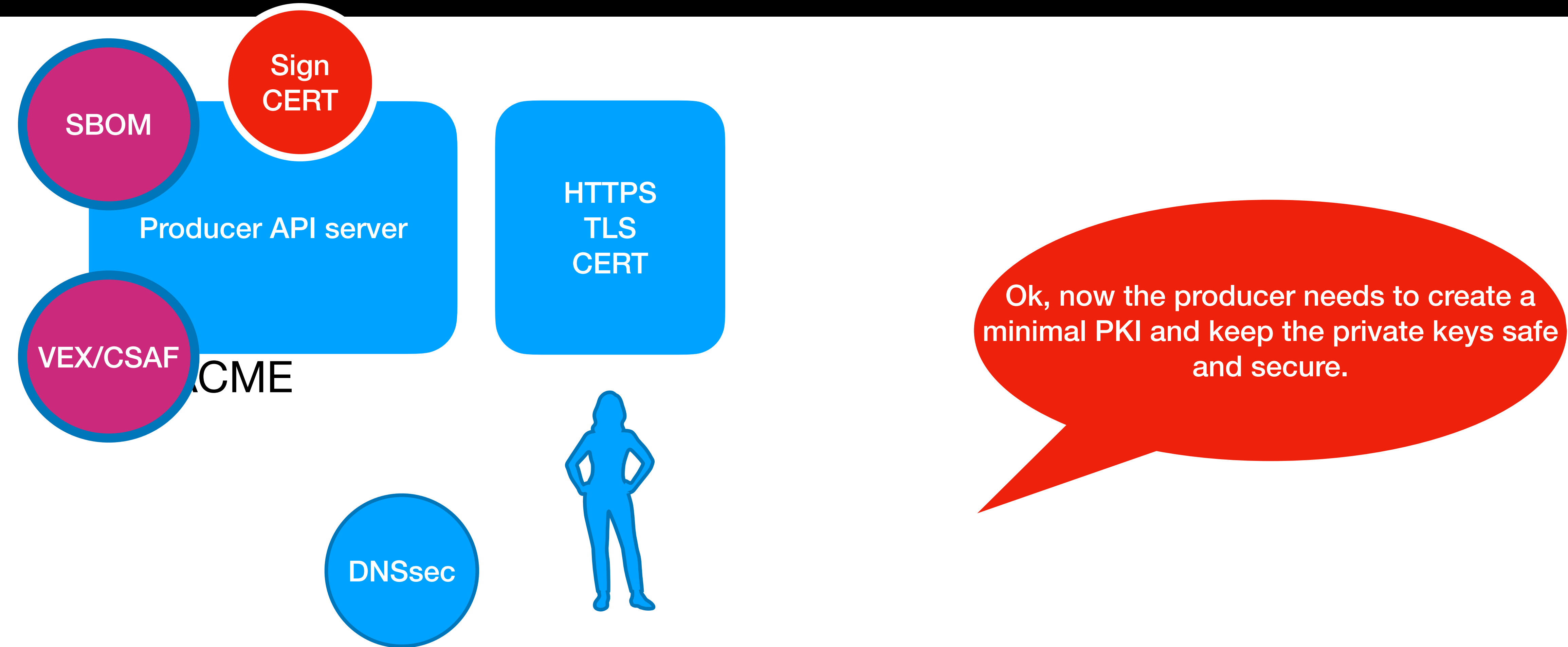
The user must trust the HTTPS connection

From HTTPS cert to sign cert



The user must trust the HTTPS connection.
Producer may add anchor in DNSsec

And a new PKI is born!



The user must trust the HTTPS connection.
Producer may add anchor in DNSsec

Can this be simplified?

My wishlist

- Simple key management
- Simple trust anchor establishment (PKI CA root)
- **Distributed**, not centralised
- With respect of **privacy**



My proposal

- Three options:
 - Web PKI usage (software signing certs)
 - Sigstore-alike solution
 - Sigstore - mainly for Open Source projects
- Requirements
 - *Everyone should be able to sign*
 - *Everyone should be able to validate*



Delivering software transparency artifacts

- TEA → automated delivery
- Signing: verifiable trust

Open Standards
Restful API
Data formats

Building blocks

SBOM

- What is present in software
- Release-level visibility

VEX

- Does the vulnerability matter?
- Reachability and mitigation context
- Assessment

TEA

- Automated API access to artifacts
- Optionally signed and verifiable

TEA Trusted Architecture

- **The TEA Trusted Architecture is an overlay that builds a trusted platform for publishing and consuming of software and systems transparency artefacts**
- Digital signatures with cryptographic verification
- Signed Timestamps
- Transparency log entries verifiable 3rd parties
- TLS confidentiality
- This is a proposal only at this time

WebPKI or Sigstore-alike model

- If a large manufacturer has certificates and infrastructure for software signing, they can use that for TEA
- But we don't want to put burden on smaller entities - SME's and Open Source projects - to pay for or manage a long term private key
- Short-lived keys, certificates, transparency log and signed time stamps in combination solves the problem
- Each object has an “evidence bundle” containing the information.
- Optionally use DNS/DNSsec for long term validation
- Validation works for a long time

Where does this take us?

- A CRA-compatible automation of the software supply chain, to deliver everything from SBOMs to vulnerability information
- Support both for in-house and TEA-as-a-service platforms
 - Trust is anchored with the manufacturer domain, not the service provider
- For Open source projects and commercial manufacturers
- A discovery from product to SBOM and more

Signed objects

- TEA API objects
- Artifacts: SBOM, VEX, Certificate of compliance
- CLE data
- Discovery JSON (.well-known)
- Corporate level compliance docs
- Will work with TEA and outside of TEA

Let's try something

- Create a key pair for signing
- Embed the public key in a self-signed certificate with a SHORT lifetime (hours)
- Include a time stamp from a TSA in the cert - standard
- log the creation to a transparency service
- sign artifacts and collections with the private key
- store cert in a DNS record in manufacturer's domain (maybe signed with DNSsec)

Let's try something

- Questions:
Was the key pair associated with the vendor at the time of signing?
- If this can be answered after ten years, we can validate a signature.
- **Transparency log:** Yes, it was created (event series)
- **Time stamp:** It was created within the validity time of the cert
- **DNS entry:** The manufacturer issued this certificate (lookup by public key fingerprint)

No more private key management

- **The private key can be deleted after time of signing.**
- No HSM, no policy, no revocation, no complex processes
- Inspired by Sigstore

No more private key management

Transparency
artifact

Digital signature
(inline or detached)

Evidence bundle

Three models of trust

Sigstore

US based

Privacy issues

TEA signatures and evidence
bundles (DNS-anchored)

Sigstore-alike

Web PKI
Software signing certificates

High cost

Requires HSM
for private key

Adding optional trust to TEA

- A layer added to TEA, but will affect some object structures
- **Discovery trust:**
From TEI to the right TEA service for a product
- **Publisher trust:**
Publishing the right artifacts and releases, by authorised staff, in the right TEA service
- **Consumer trust:**
Am I getting these artifacts from the vendor of my product? Can I trust this?

CONSUMER

From chaos...

LOG4SHELL
CVE-2021-44228

Do we have Log4j?



Do we have Log4j?

...to trusted, automated decisions



Discovery
Find artifacts



TEA API
Collect artifacts



VEX
Assess risk



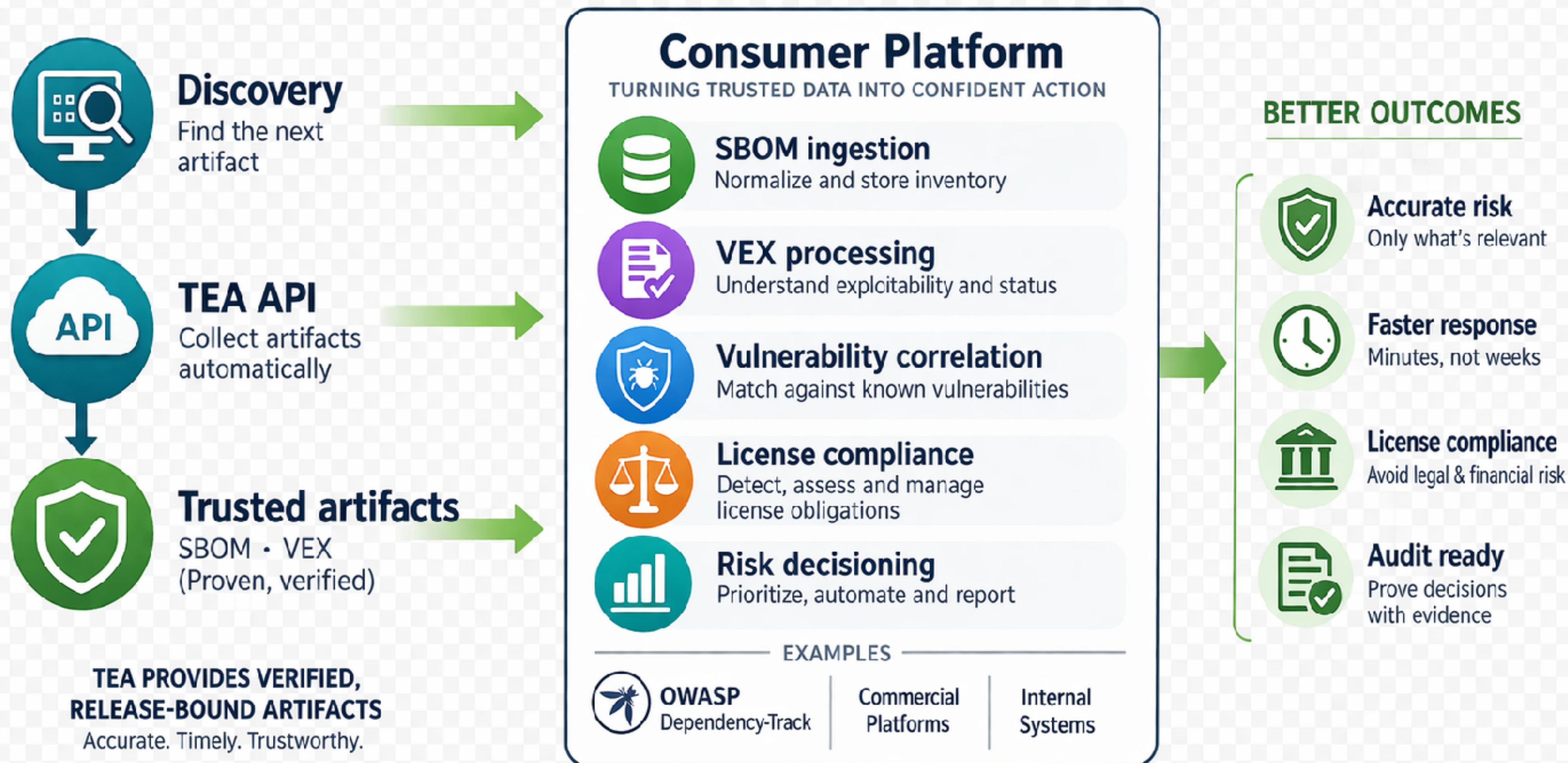
Trust
Automate decisions



OWASP TRANSPARENCY EXCHANGE API

How TEA powers Consumer Platforms

Trusted data. Smarter decisions.



TEA DELIVERS TRUSTED INPUTS. PLATFORMS DELIVER BETTER DECISIONS.
TOGETHER, THEY CREATE RESILIENT, COMPLIANT, AND SECURE ORGANIZATIONS.

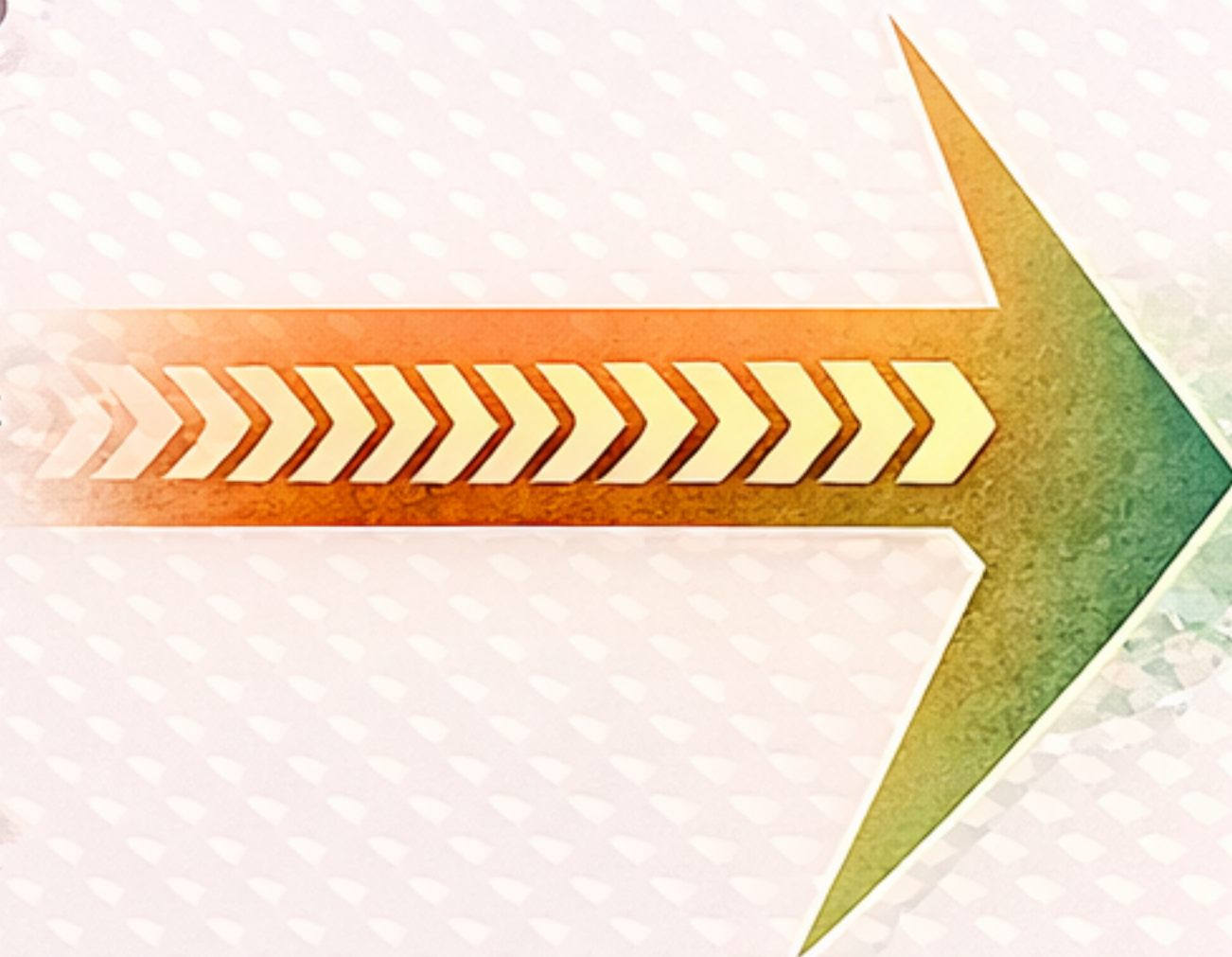
From **Uncertainty** to Automated Trust



Discovery
Find the next artifact



Do we have Log4j?



Consumer Platform

TURNING TRUSTED DATA INTO CONFIDENT ACTION



SBOM ingestion
Normalize and store inventory



VEX processing
Understand exploitability and status



Vulnerability correlation
Match against known vulnerabilities



Risk decisioning
Prioritize, automate and report



OWASP
Dependency-Track

Commercial
Platforms

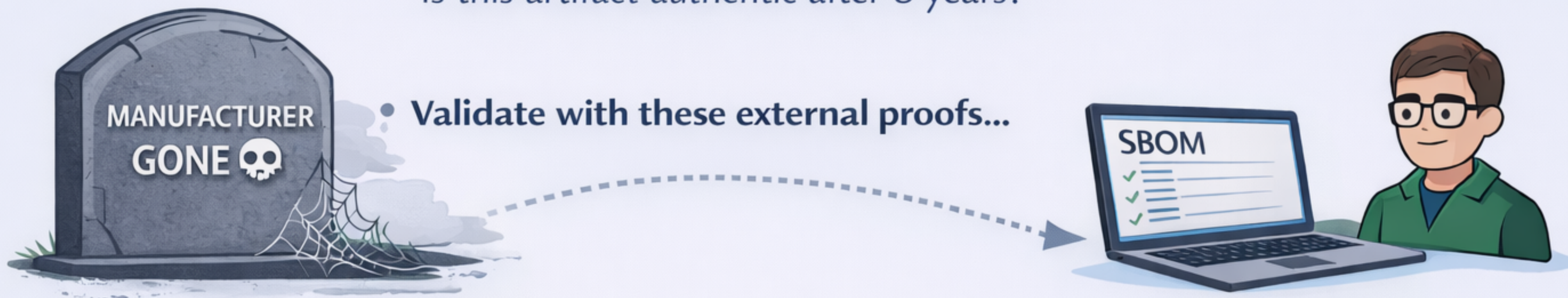
Internal
Systems



Minutes instead of weeks

Validating a Release After 8 Years

Is this artifact authentic after 8 years?



External Proofs



TSA Timestamp

Proves signing time came from TSA



Short-Lived Certificate

Defines signing ID for that timestamp



Lifespan was valid when timestamp was signed by TSA



Short-Lived Certificate

Defines signing ID for that timestamp



Validates using DNS



Lifespan was valid when timestamp was signed by TSA



Transparency Log

Audits the timestamp & short-lived cert

Is this proof coherent with 10 years ago?

- ✓ TSA's key was valid then
- ✓ DNS matches old records
- ✓ Rekord log was consistent

The repo

- <https://github.com/oej/tea-trust-architecture/tree/main>
- Contains AI-generated docs for the current API - possible skeletal input for ECMA standard docs ?
- Contains a proposal for a TEA trusted architecture
- Contains a few scripts for generating and validating TEA evidence bundles

When the next Log4shell happens

- Downstream actors - vendors and end users - will have automatic updates
- There will be trusted data to base mitigation actions upon
- We've gone from unstructured chaos to an ordered automatic transparency system

**Will only happen if
customers require this!**

Some cool stuff

- We need more dancers: <https://datatracker.ietf.org/group/dance/about/>
DANE Authentication for Network **Clients** Everywhere (dance)
- Certs and trust anchor selection in DNSsec
<https://datatracker.ietf.org/wg/dane/documents/>
- Public transparency log of signatures
<https://www.sigsum.org/>
- Sigstore: <https://www.sigstore.dev/>
OpenSSF project
- TEA, OWASP Transparency Exchange API
<https://github.com/CycloneDX/transparency-exchange-api>



Let's work on this!



- Mastodon @oej@infosec.exchange
- Matrix [oej@matrix.org](https://matrix.to/#/oej@matrix.org)
- LinkedIn (QR)
- Email oej@edvina.net
- Github oej

