OAuth and SPIFFE

Workload Identities and Authorization

About Us

Evan Gilman

SPIFFE/SPIRE Maintainer
Co-founder @ SPIRL

Pieter Kasselman

IETF OAuth Usual Suspect
Identity Standards @ Microsoft

What is SPIFFE/SPIRE?

SPIFFE is a set of **platform-agnostic** specifications defining the documents and interfaces necessary for a **federated workload identity** scheme

… and SPIRE is an **open source SPIFFE implementation** that is designed to run in a multitude of computing environments and platforms

SPIFFE does:

- Solve secure introduction
- Work in highly elastic and distributed environments
- Support (very) short-lived credentials
- Automate key rotation from root down

SPIFFE does *not*:

- Solve authorization
- Reason about human identities
- Reason about or rely on network locators (e.g. DNS names)
- Invent or define new documents

… and why should you care?

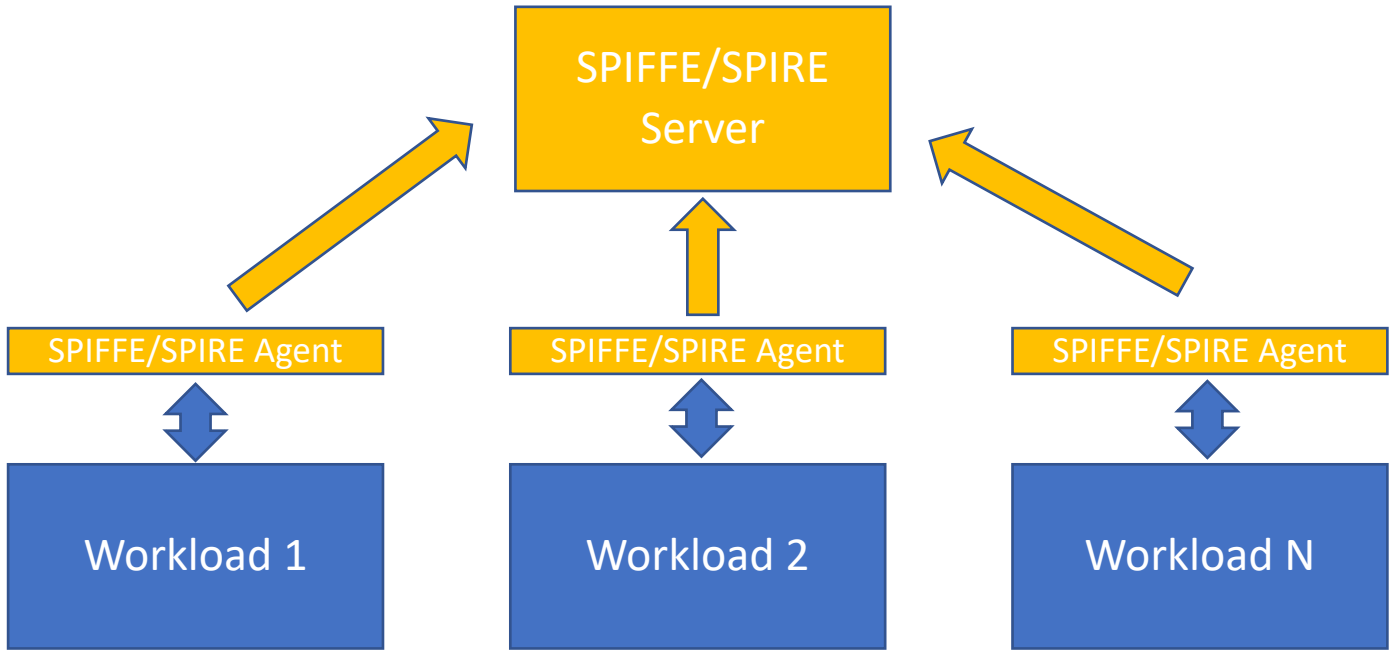
- SPIFFE uses standard X.509 and JWT documents
  - SPIFFE calls these SVIDs (SPIFFE Verifiable Identity Documents)
- … but SPIFFE is different
  - Trades revocation for a short-lived, online, highly rotational key infrastructure
  - Provides not just identity credentials to workloads, but also trust bundle(s)

- SPIFFE and OAuth have overlapping problem space
  - SPIFFE != OAuth, but two sides of same identity coin
  - Shared problems (e.g. uncontrolled token propagation)
  - Interop (e.g. SPIFFE to bootstrap OAuth Client)

(this is why we are here)

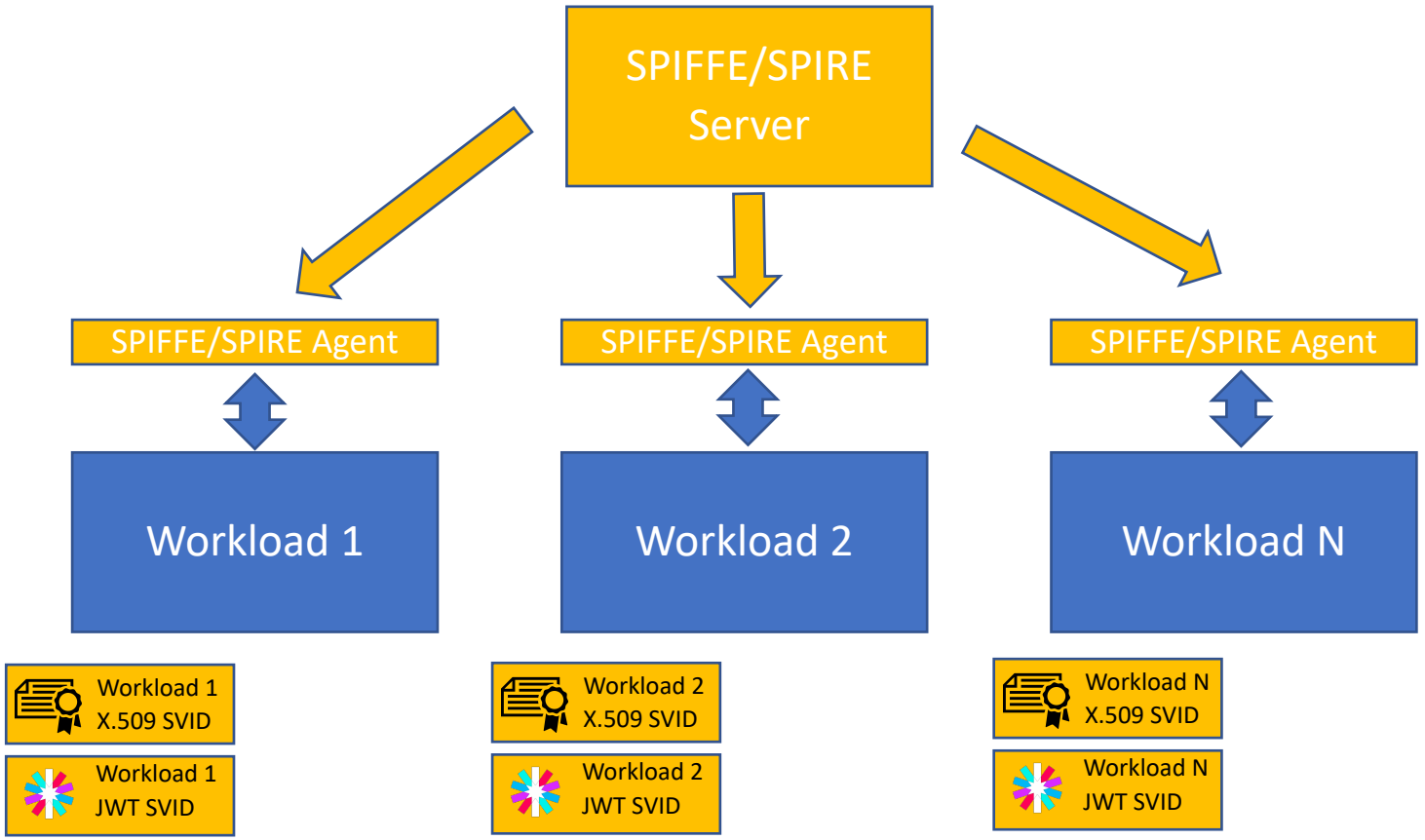
SPIFFE/SPIRE – A worked example

1. Agents attest node identity to SPIRE Server (e.g. TPM challenge or using cloud APIs as trusted third party)
2. Workload request identity, agents attests workload properties to determine identity



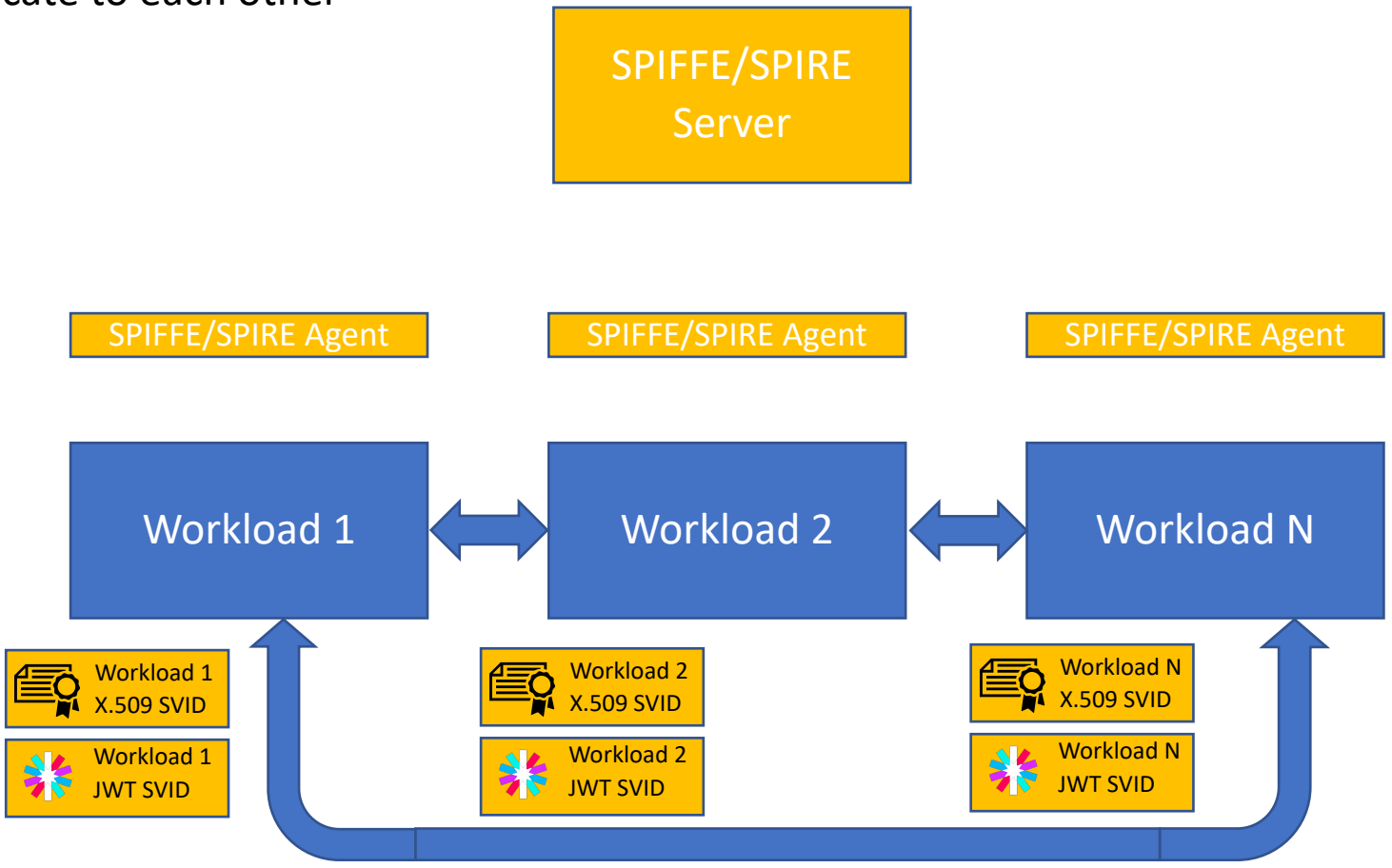
SPIFFE/SPIRE – A worked example

1. Agents attest node identity to SPIRE Server (e.g. TPM challenge or using cloud APIs as trusted third party)
2. Workload request identity, agents attests workload properties to determine identity
3. SPIFFE Authority issues SVIDs as X.509 certs and JWT bearer tokens to workloads



SPIFFE/SPIRE – A worked example

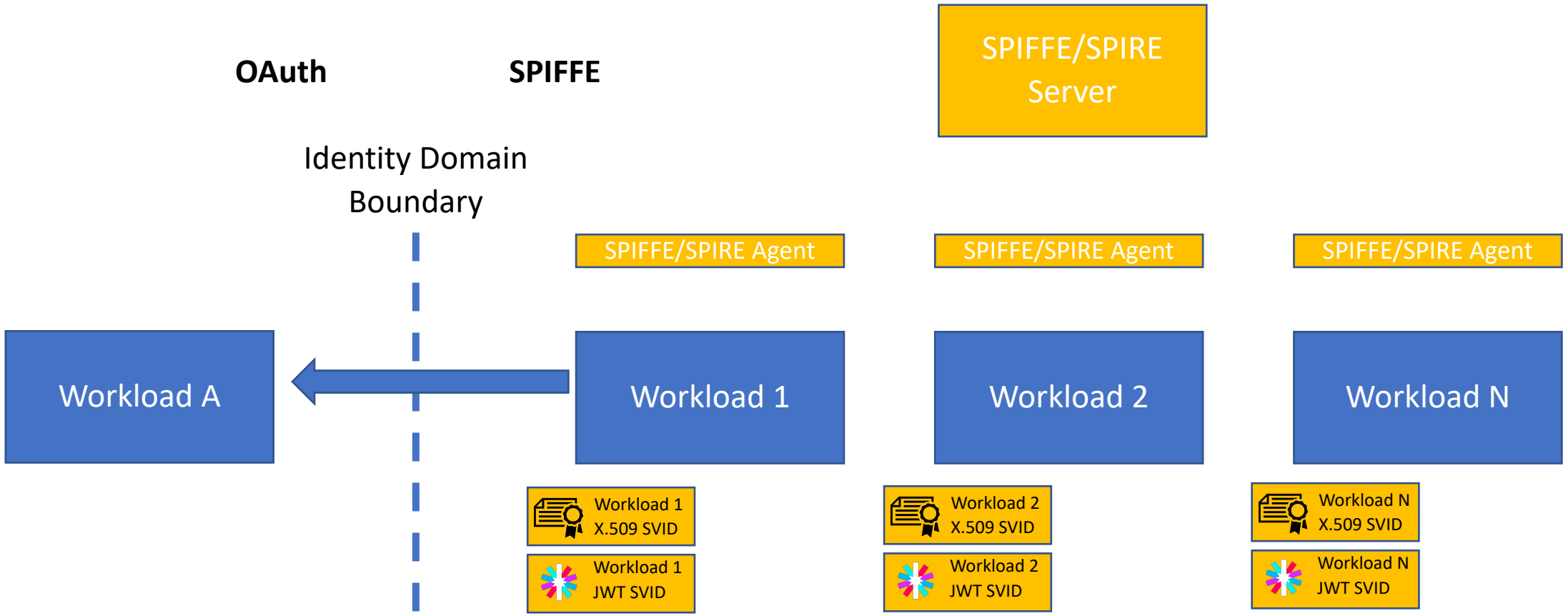
1. Agents attest node identity to SPIRE Server (e.g. TPM challenge or using cloud APIs as trusted third party)
2. Workload request identity, agents attests workload properties to determine identity
3. SPIFFE Authority issues SVIDs as X.509 certs and JWT bearer tokens to workloads
4. Workloads use SVIDs to authenticate to each other



7 ~~Wonders~~ Use Cases of the Workload Identity World

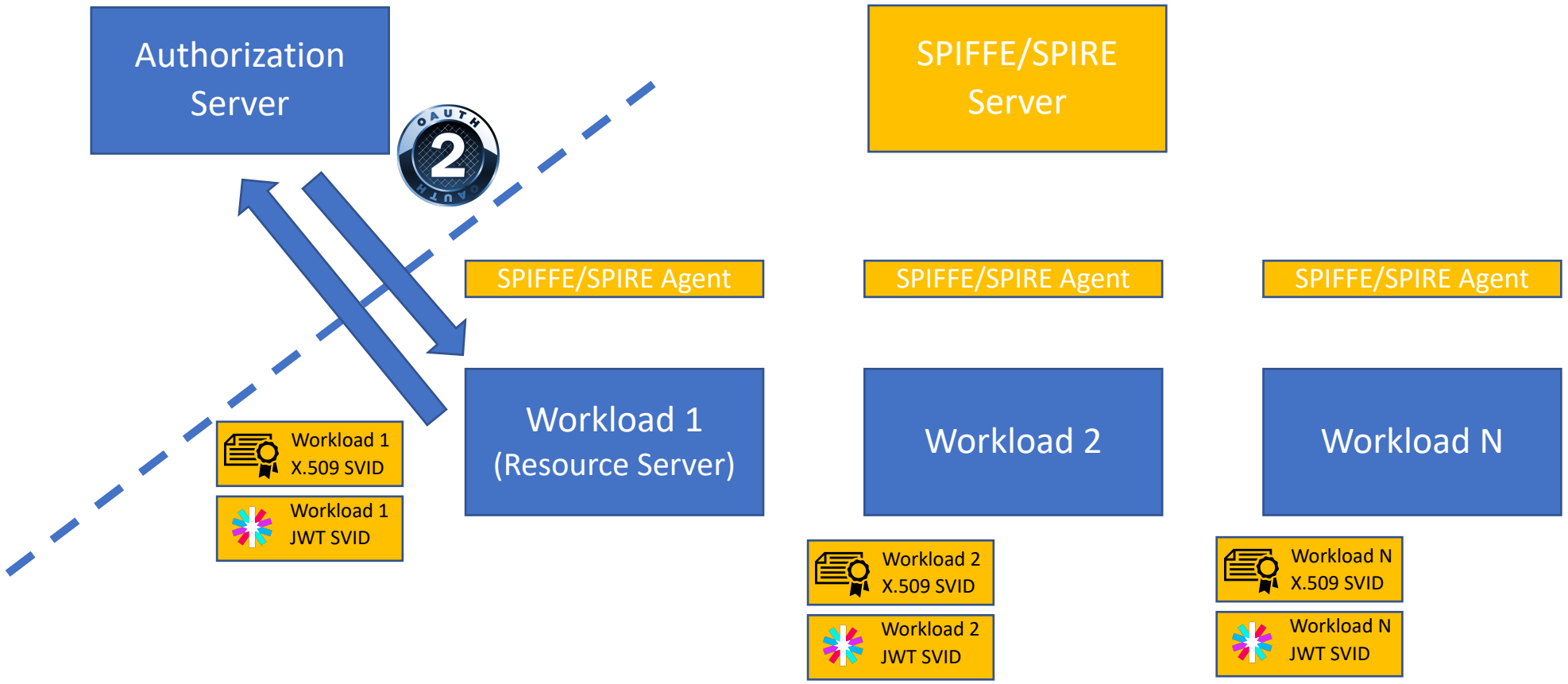
Use Case 1: Access workloads in another domain

- Workload 1 needs to access Workload A, which may not be using SPIFFE



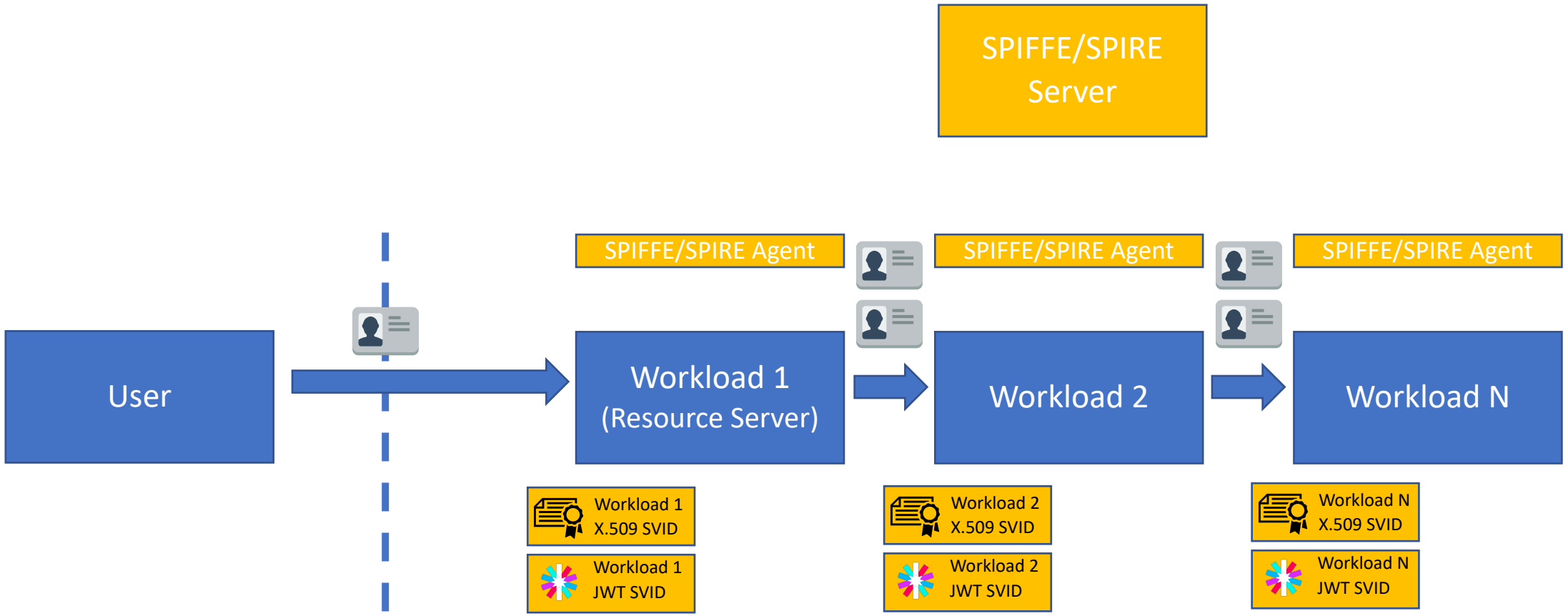
Use Case 2: No SPIFFE client registration

- Workloads spin up and down dynamically, and new identities issued in response
- SPIFFE carries identity - how can SPIFFE client AuthN to OAuth Authorization Server w/o registration
- SPIFFE keys rotate frequently .. how does OAuth Authorization server track and validate SPIFFE identities



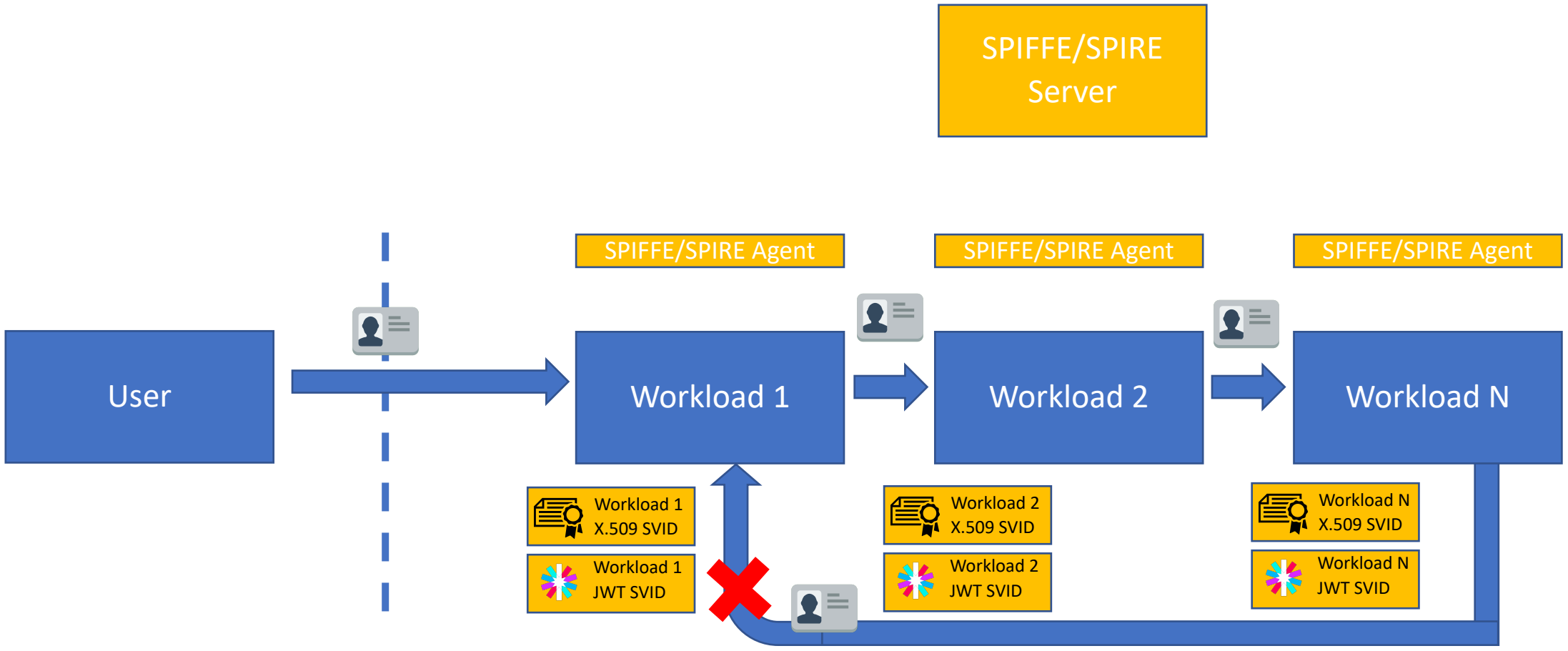
Use Case 3: Preserve Call Stack Identities

- A resource server may be implemented as multiple micro-services.
- Every micro service needs an assertion of the user identity for authZ and audit
- Every micro service also needs the identity of the service calling it
- Sometimes, a micro service needs the identity of N-* service calling it



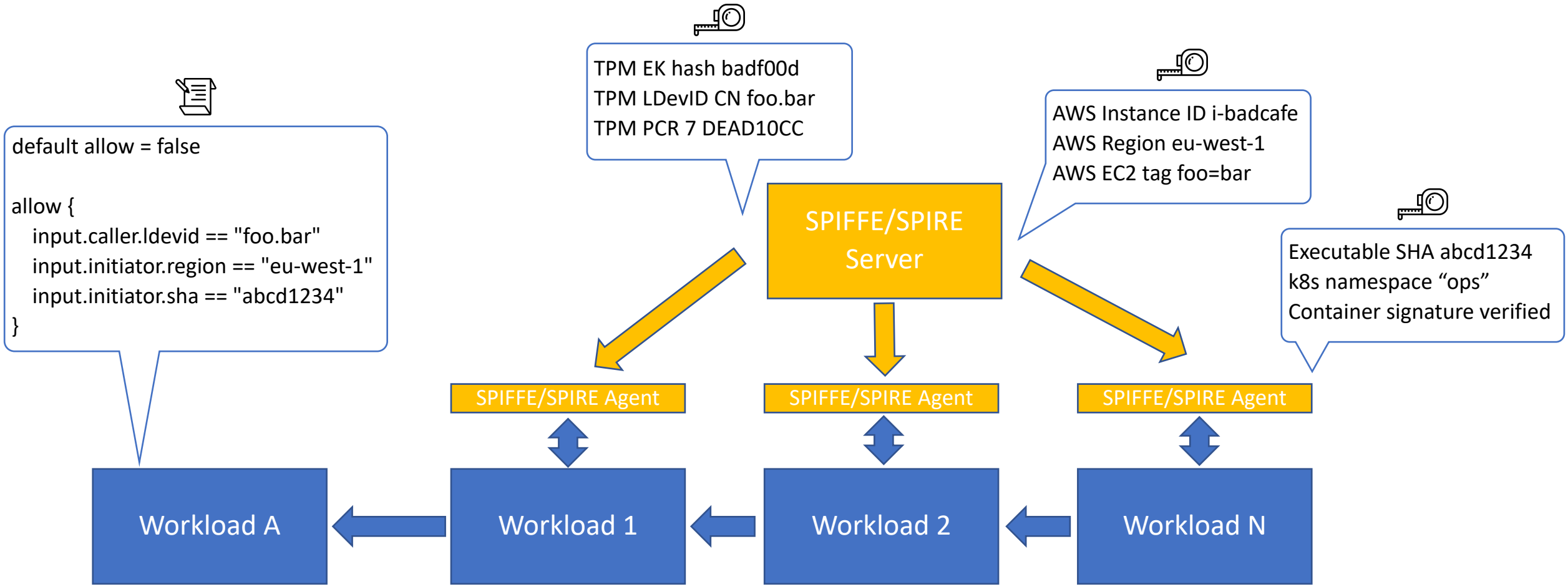
Use Case 4: Protect against token theft

- Workload *and* user tokens transit service graph
- How can these tokens be bound to the request at hand?
- Important for SPIFFE to avoid expensive network calls



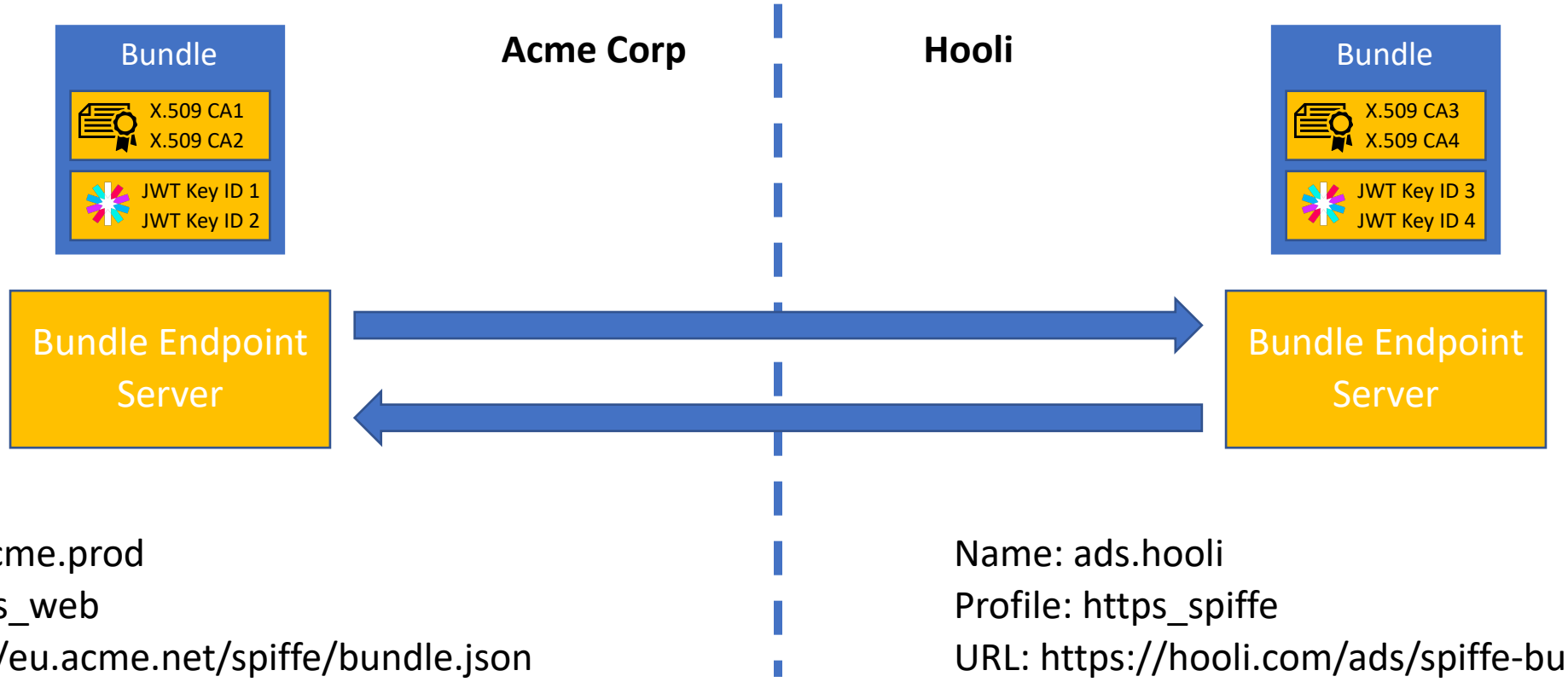
Use Case 5: Fine(r) Grained Authorization

- SPIFFE/SPIRE produces trustworthy data as part of the attestation/issuance process
- Not currently captured, but highly valuable - can be used to build e.g. ABAC
- How do we convey this data for consideration at the PDP ?



Use Case 6: Inter-domain (meta)data distribution

- SPIFFE supports federation ~similar to OIDC
- Extra information needed to complete federation configuration
- How can we distribute this information to ease the creation and maintenance of federation relationships
- Information could be endpoint coordinates, or bundle data itself

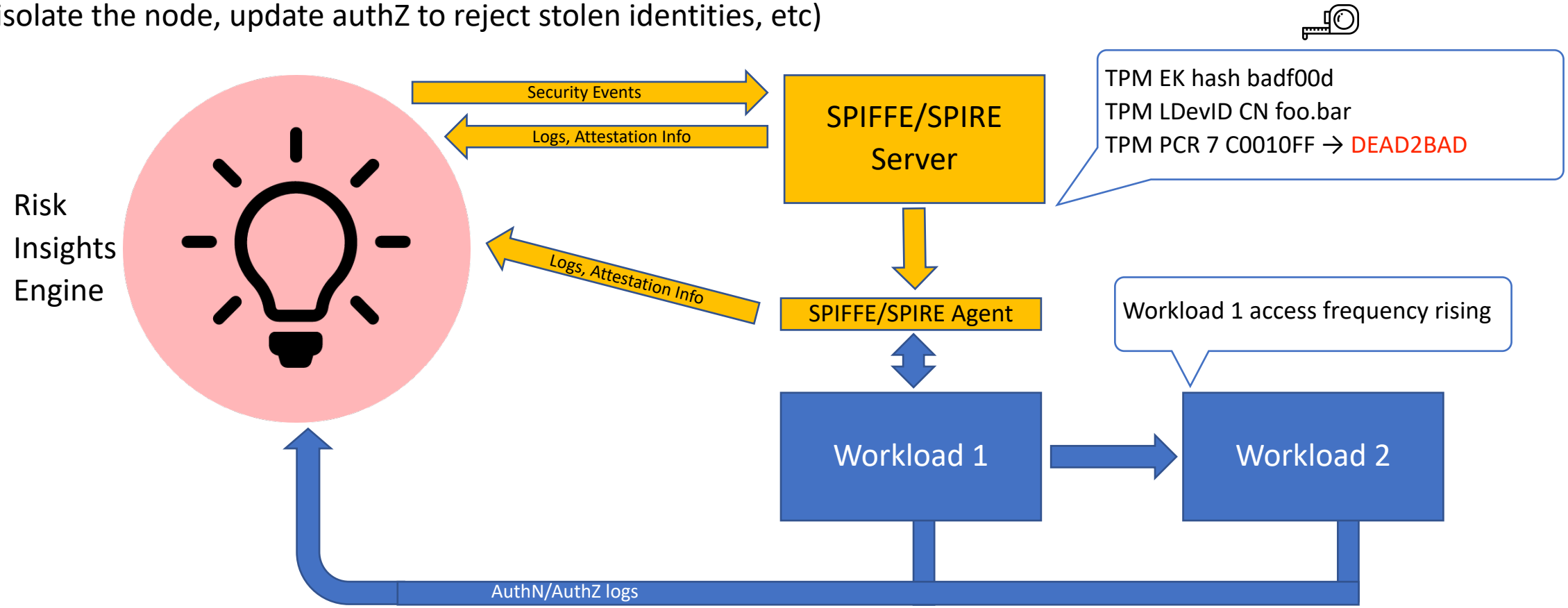


Name: eu.acme.prod
Profile: https_web
URL: https://eu.acme.net/spiffe/bundle.json

Name: ads.hooli
Profile: https_spiffe
URL: https://hooli.com/ads/spiffe-bundle.json
Bootstrap: <bundle_content>

Use Case 7: Monitoring and remediation

- SPIFFE/SPIRE attestation data can be regularly published
- SPIFFE AuthN logs can also be published
- Together, they provide a strong risk signal on workloads and related infra
- Possible to trace from workload ID all the way down to h/w
- Send signals to SPIFFE/SPIRE & others to remediate suspected compromises (e.g. isolate the node, update authZ to reject stolen identities, etc)



Use Case *

- … and many more!
- What comes to *your* mind?

Next Steps?

- Document uses cases
- Identify existing OAuth standards to profile/extend
  - OAuth 2.0 Dynamic Client Registration:
  - OAuth 2.0 Authorization Server Metadata:
  - OAuth 2.0 DPoP:
  - OAuth MTLS:
  - OAuth JWT Bearer Flow
  - OAuth Rich Authorization Request
  - Others?
- Create “OAuth Profile for SPIFFE Best Current Practice” Draft?
- Identify and align new standards work to fill gaps

APPENDIX

Topics to cover

- Introduce SPIFFE (focus on federation) – 5 min
- Use Cases Overview 15 min
  - Authentication/Front Door
  - Identity chaining
  - Federation uses cases
  - Dynamic registration
  - Metadata
  - Supply chain (Assertions)
  - Authorization decisions (Assertion)
- Strawman for the BCP – best current practice 5 min
  - JWT SVID
  - JWT X.509 (auth)
  - Dynamic Client Registration
  - Metadata
  - MTLS
  - DPoP
- New Work beyond BCP? (0 – minutes)
  - Identity chaining
  - Fine grained authorization
- Side Meeting
  - Double click on scenarios – explore assertion idea
  - SCITT