

SPIFFE Meets OAuth: Federated Identity for Cloud Native Workloads

Prepared by
Yoshiyuki Tabata
Hitachi OSPO

Date
March 26, 2026

Agenda

1. Modern Platforms Have Multiple Security Authorities (Trust Domains)
2. What Breaks Federated Identity Across Trust Domains
3. How to Securely Build Federated Identity Across Trust Domains
4. Demo: Federated Identity in Action with Keycloak and SPIRE

Welcome – about speaker



Yoshiyuki Tabata

Senior OSS Consultant @ Hitachi OSPO

- Tech Lead of CNCF TAG Security and Compliance
- CNCF Ambassador / Cloud Native Community Japan board / Cloud Native Security Japan founder
- LinkedIn: [@ytabata](#), X: [@yo_tabata](#), GitHub: [@y-tabata](#)

Securing API and Identity in Enterprise and Cloud Native Platforms

Professional Expertise

- Over 10 years designing and consulting on secure API ecosystems and identity platforms for enterprise and cloud native environments

Community Leadership & Open Source Contributions

- CNCF TAG Security and Compliance: best practices, whitepapers, APAC facilitation
- Keycloak: core feature contributions, language maintainer, KeycloakCon Japan/EU co-chair
- Speaker at KubeCon, Open Source Summit
- Author of books and articles on IAM, Keycloak, and AI agents
- CNCF meetup organizer in Japan

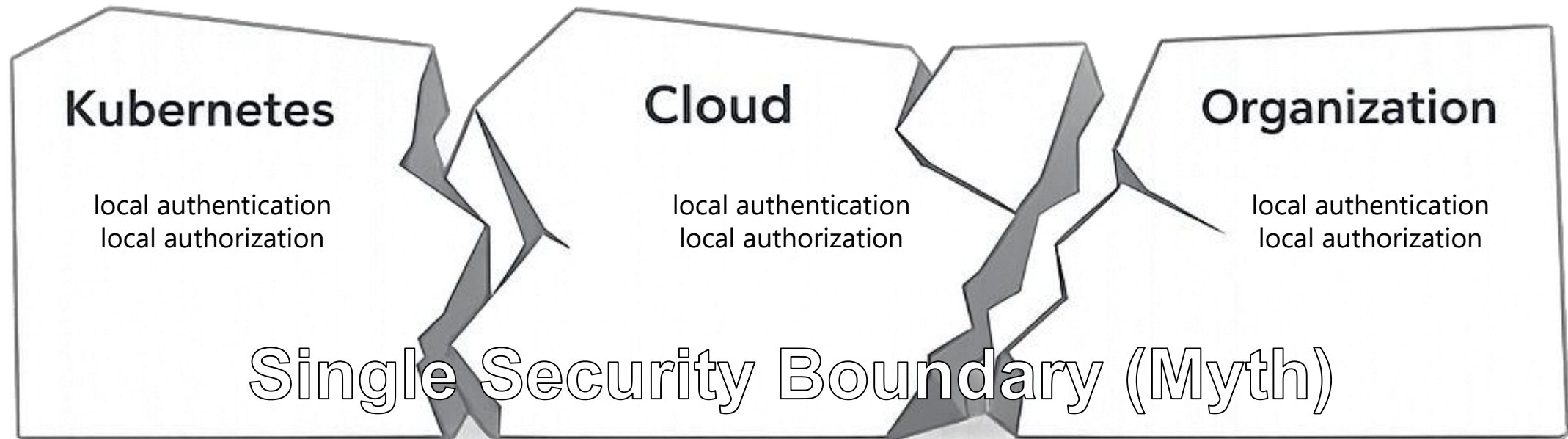
Agenda

1. Modern Platforms Have Multiple Security Authorities (Trust Domains)
2. What Breaks Federated Identity Across Trust Domains
3. How to Securely Build Federated Identity Across Trust Domains
4. Demo: Federated Identity in Action with Keycloak and SPIRE

Modern Platforms Are Not a Single Security Boundary

Applications today span multiple Kubernetes clusters, clouds, and organizations — increasingly involving interactions between applications, tools, and AI agents.

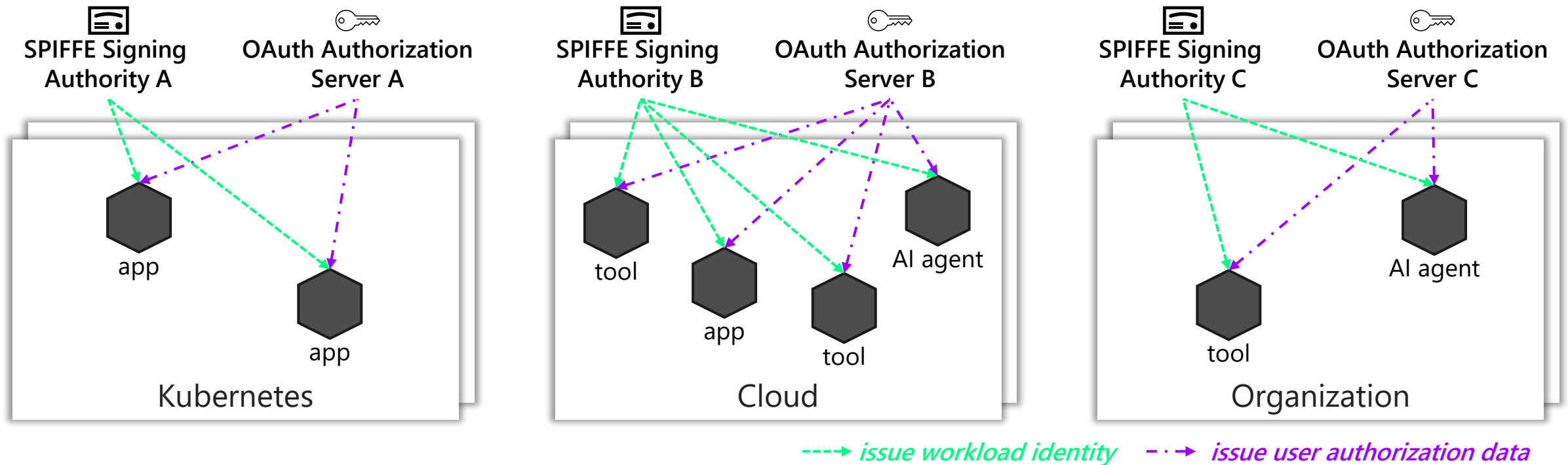
Each of these environments makes its own authentication and authorization decisions, rather than relying on a single centralized authority.



Multiple Authorities Are the Norm

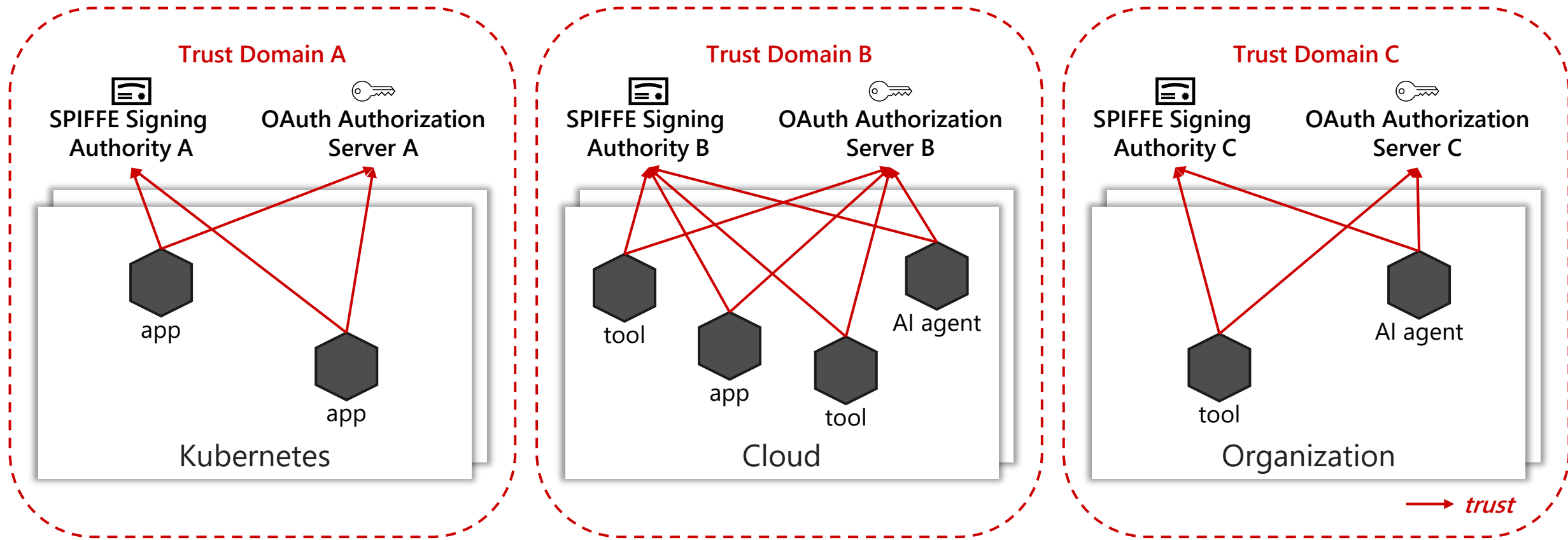
In practice, both **workload identities** and **user authorization data** (such as OAuth tokens, scopes, or claims) are issued, evaluated, and enforced independently across different parts of the system.

For this reason, we treat workload identity and user authorization as distinct concerns when discussing federated identity across trust domains.



What We'll Call a Trust Domain

We call each independent security authority boundary a trust domain: a boundary within which identity and authorization decisions are made and trusted.



What We'll Call a Trust Domain

We call each independent security authority boundary a trust domain: a boundary within which identity and authorization decisions are made and trusted.

While **OAuth** and **SPIFFE** define trust domains slightly differently, we intentionally treat them as the same boundary in this talk, because in many real platforms they are operated and governed together.

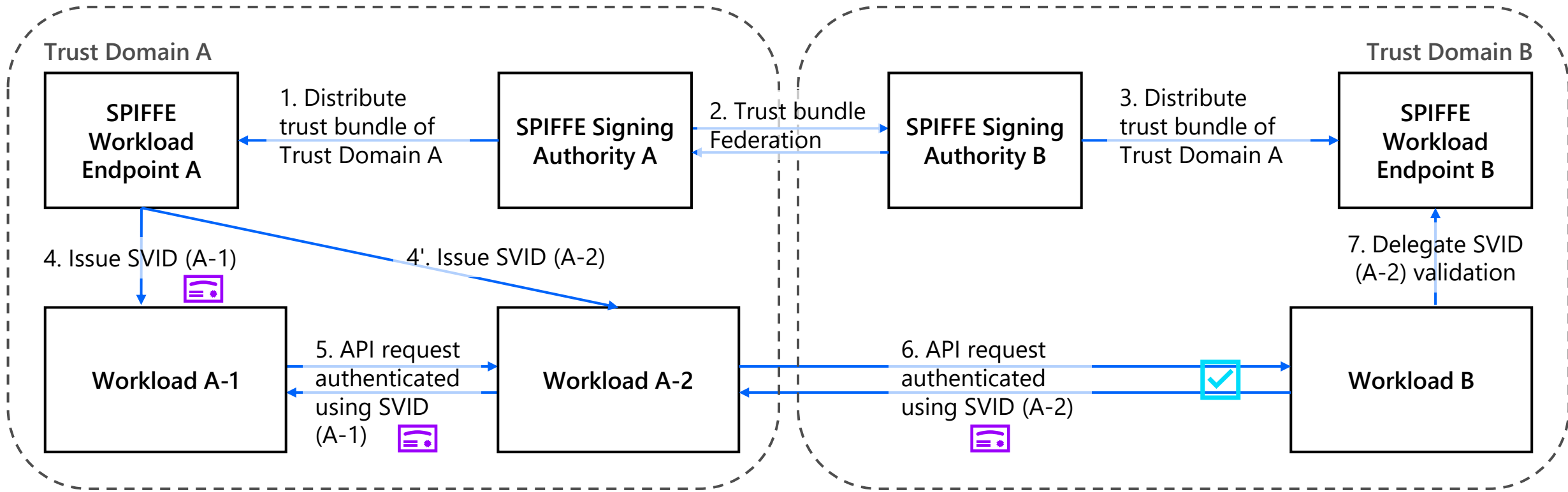
Aspect	OAuth (Identity & Authorization Chaining)	SPIFFE
What is a trust domain?	A management boundary in which one logical Authorization Server authority is responsible for issuing and standing behind authorization tokens and assertions	A management boundary defined by a SPIFFE ID namespace and the authority that issues identities within that namespace
Anchor of trust	The Authorization Server as the authoritative issuer of authorization tokens and assertions	The SPIFFE trust domain and its identity-issuing authority (CA)
What is governed within the domain	Token issuance, token semantics, and responsibility for their meaning	Identity issuance, identity namespace, and responsibility for identity validity

Agenda

1. Modern Platforms Have Multiple Security Authorities (Trust Domains)
2. **What Breaks Federated Identity Across Trust Domains**
3. How to Securely Build Federated Identity Across Trust Domains
4. Demo: Federated Identity in Action with Keycloak and SPIRE

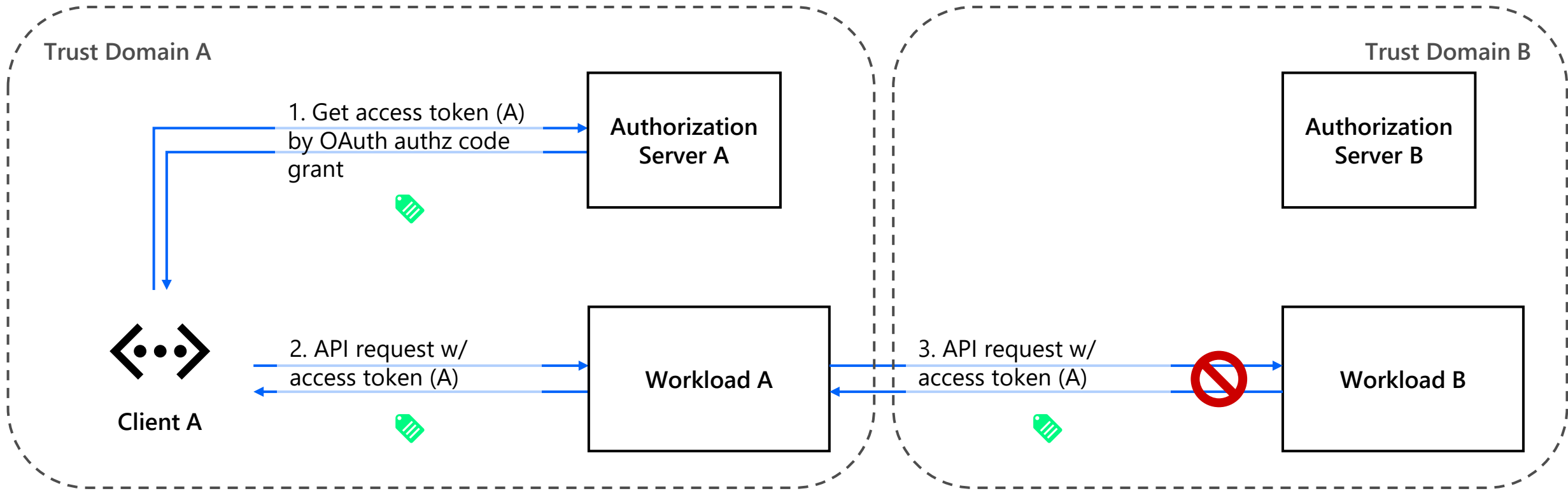
Workload Identity Federation Works Well

With mechanisms like **SPIFFE**, trust is anchored in **SPIFFE Signing Authorities** and distributed as trust bundles, while **SPIFFE Workload Endpoints** issue and verify workload identities (**SVIDs**), enabling workloads to authenticate securely across trust domains.



User Identity Federation Is Hard: Simple Token Propagation Is Not Enough

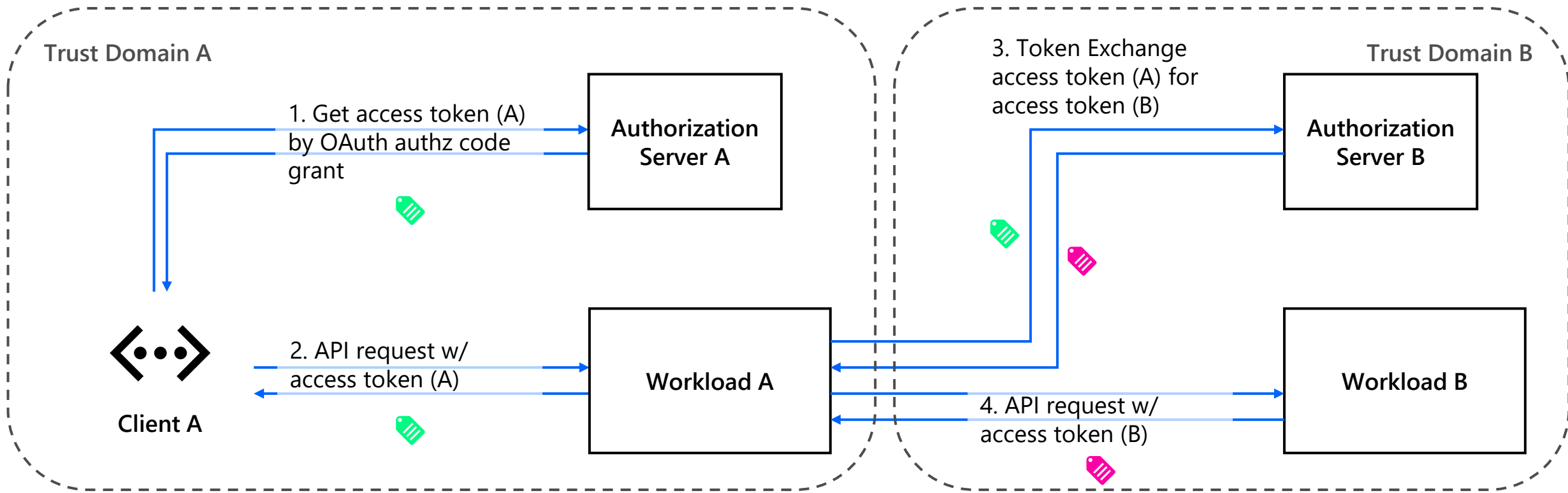
Preserving who a request is for — and the associated authorization context, as expressed through OAuth tokens and claims — across multiple service hops is significantly harder than authenticating the calling workload itself. **Simply reusing or forwarding access tokens across trust domains is strongly discouraged**, as it breaks audience constraints and obscures where authorization decisions were made.



Naive Token Exchange Is Not Enough

After rejecting simple token reuse, **token exchange** is often the next intuitive solution.

However, applying token exchange mechanically does not by itself explain why another authorization server should trust the original authorization decision. Token Exchange alone only changes the token format and audience; it does not make the original authz decision verifiable by the receiving domain.



... So what **does** work?

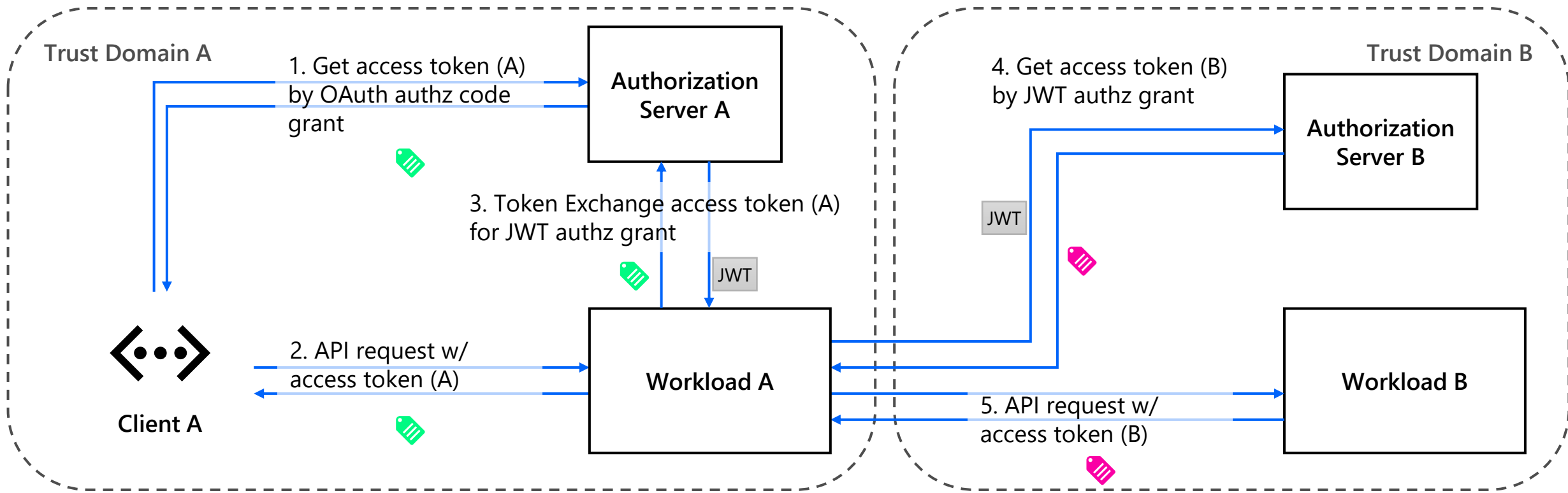
Agenda

1. Modern Platforms Have Multiple Security Authorities (Trust Domains)
2. What Breaks Federated Identity Across Trust Domains
3. **How to Securely Build Federated Identity Across Trust Domains**
4. Demo: Federated Identity in Action with Keycloak and SPIRE

Identity and Authorization Chaining

The **OAuth Identity and Authorization Chaining Across Domains*** draft defines a way to preserve identity and authorization semantics across trust domains.

By combining **Token Exchange (RFC 8693)** and **JWT-based Authorization Grants (RFC 7523)**, a request can cross trust domains while preserving who it is for and what is allowed.

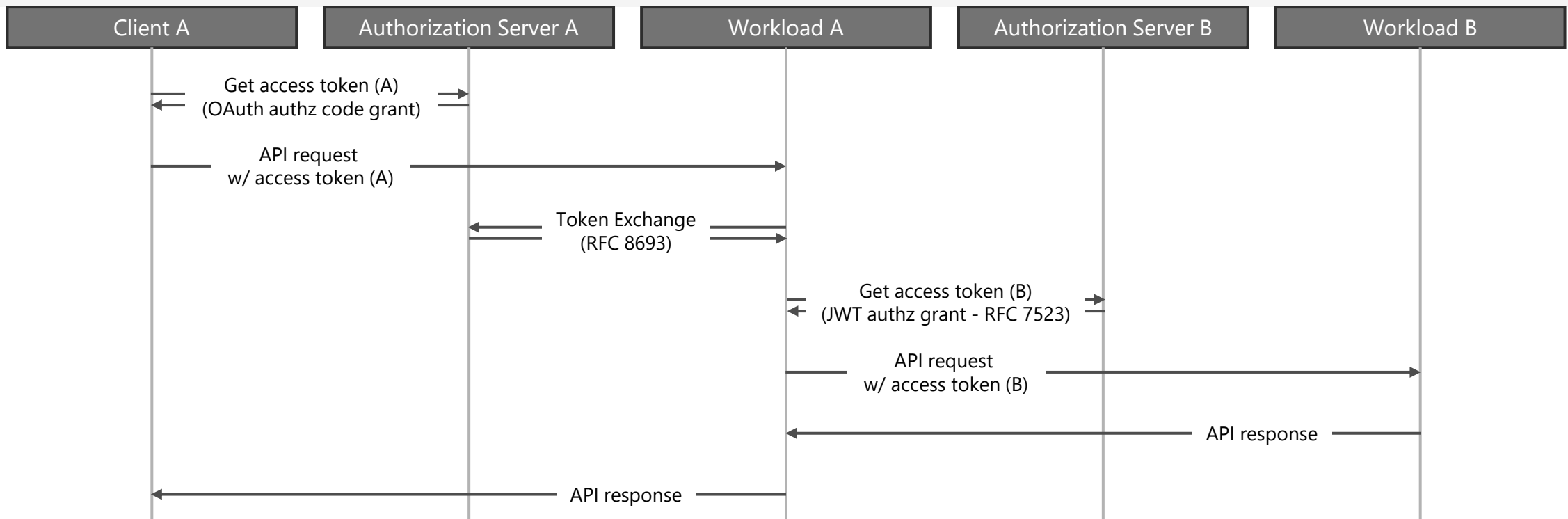


* OAuth Identity and Authorization Chaining Across Domains, <https://datatracker.ietf.org/doc/draft-ietf-oauth-identity-chaining/>

Identity Chaining Flow

The **OAuth Identity and Authorization Chaining Across Domains*** draft defines a way to preserve identity and authorization semantics across trust domains.

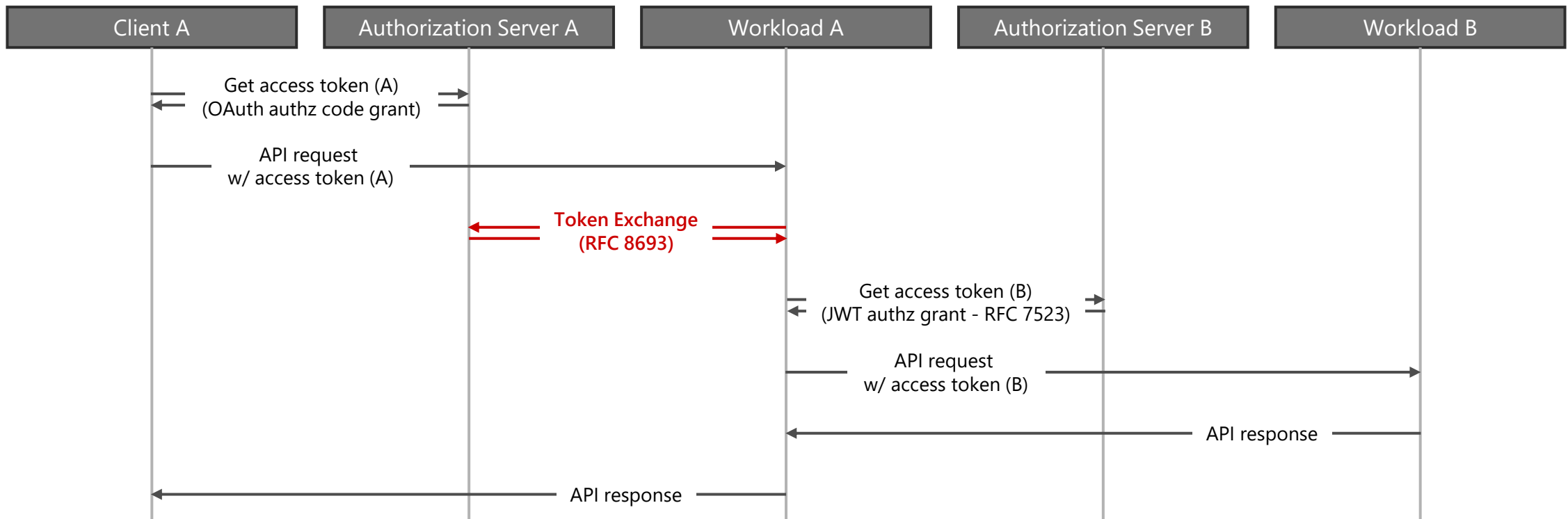
By combining **Token Exchange (RFC 8693)** and **JWT-based Authorization Grants (RFC 7523)**, a request can cross trust domains while preserving who it is for and what is allowed.



* OAuth Identity and Authorization Chaining Across Domains, <https://datatracker.ietf.org/doc/draft-ietf-oauth-identity-chaining/>

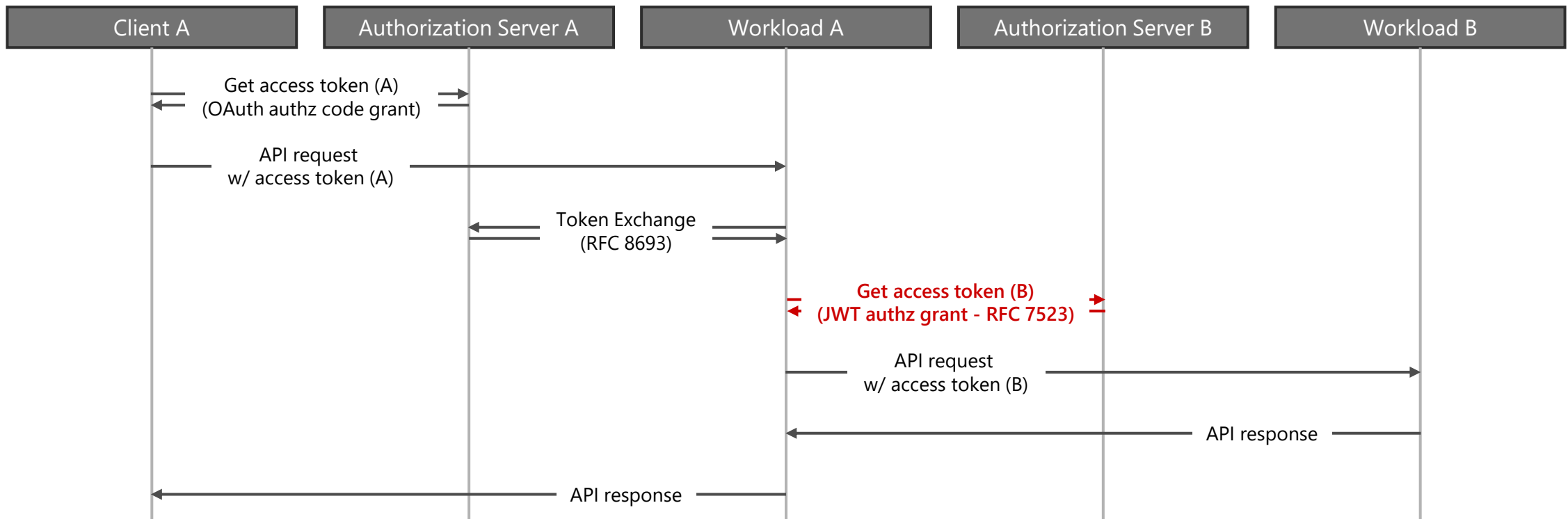
Token Exchange (RFC 8693)

In the identity chaining flow, **Token Exchange** is used to exchange an access token for a JWT-based authorization grant representing delegated authorization that is intended for another trust domain.



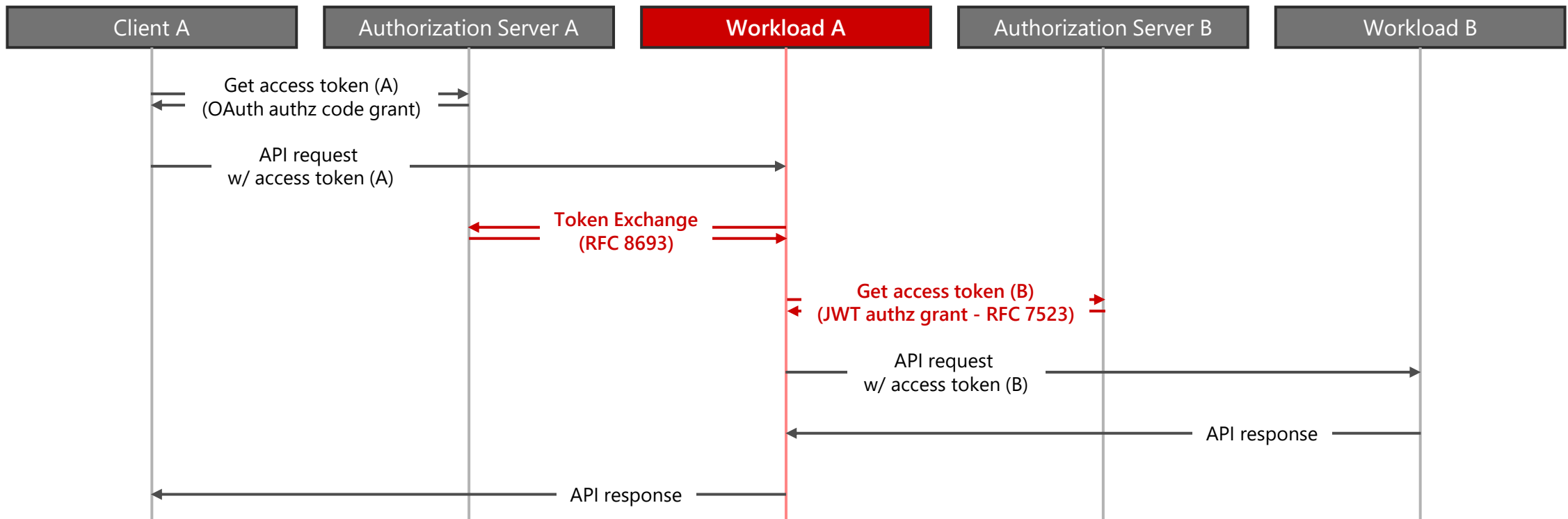
JWT-Based Authorization Grants (RFC 7523)

JWT authorization grants allow an authorization server to issue its own access token based on a cryptographically verifiable grant that represents authorization decisions made by another authorization server.



The Client Authentication Problem

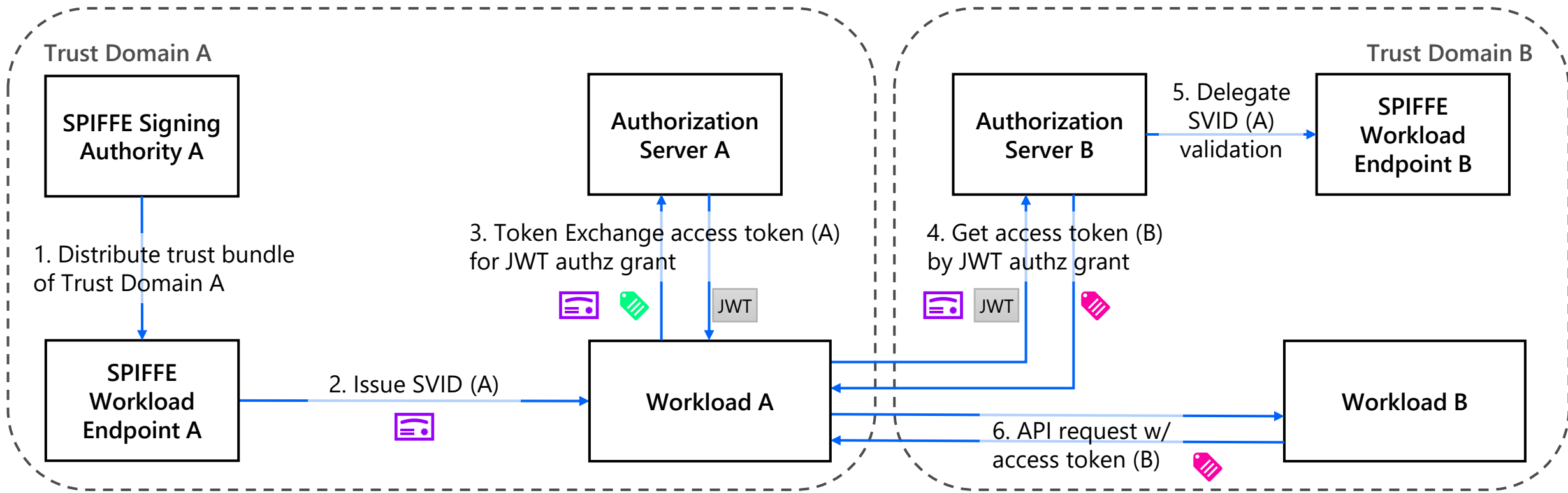
Services participating in these flows must authenticate securely when requesting or exchanging tokens, without relying on long-lived client secrets or manual secret distribution.



SPIFFE Meets OAuth: Client Authentication

The **OAuth SPIFFE Client Authentication*** draft defines how SPIFFE credentials (JWT-SVID or X.509-SVID) can be used for OAuth client authentication.

This eliminates static client secrets and ensures that token exchange and grant requests are made by a cryptographically verifiable workload identity.



* OAuth SPIFFE Client Authentication, <https://datatracker.ietf.org/doc/draft-ietf-oauth-spiffe-client-auth/>

What Secure User Delegation Requires Across Trust Domains

To securely perform federation flows such as token exchange and JWT-based authorization grants, an authorization server must have clarity on three aspects of user delegation:
what is being delegated, for whom it is delegated, and which workload performed the delegation.

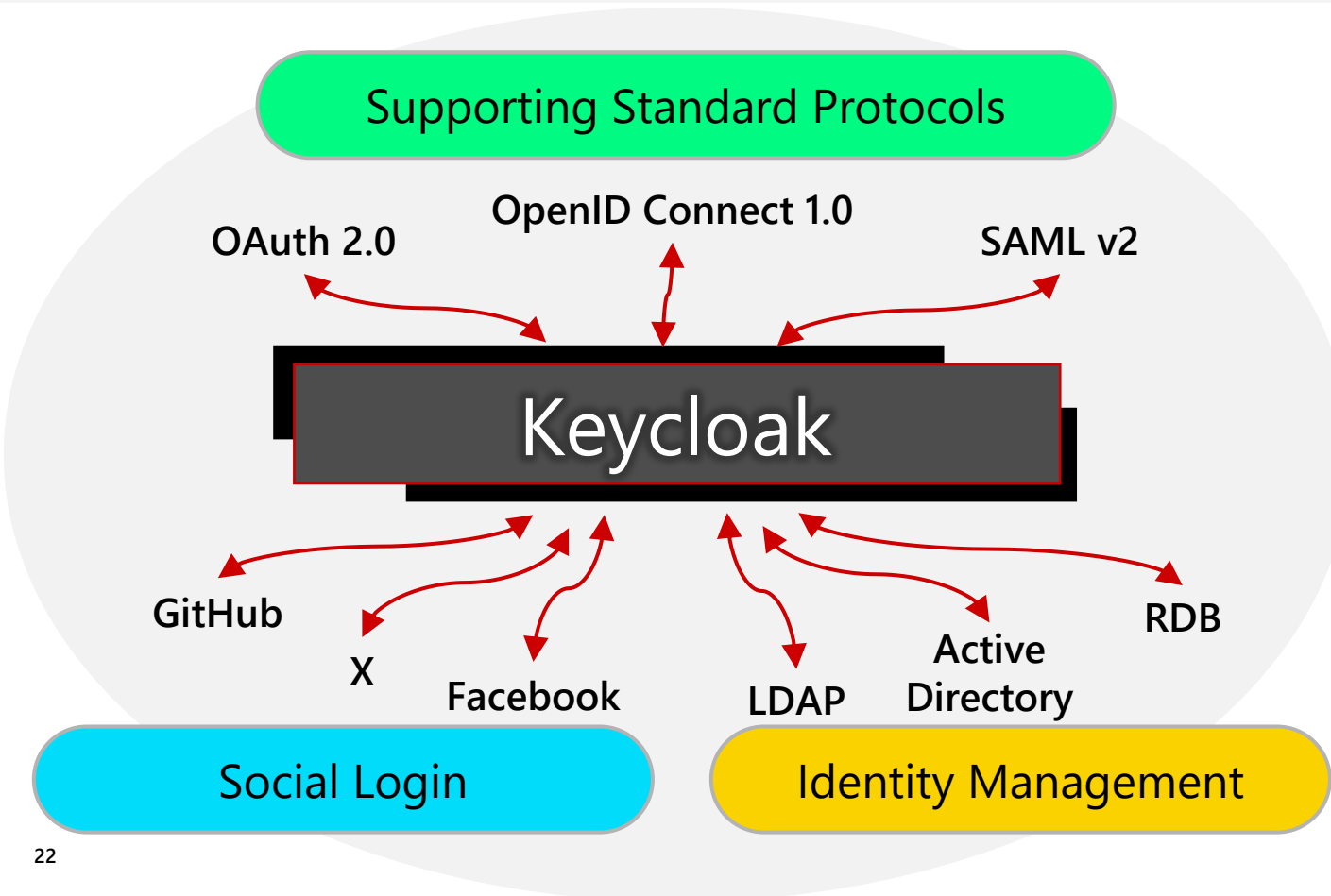
Aspect of User Delegation	Identity Chaining (Token Exchange, JWT-based Grants)	SPIFFE Client Authentication
① What is being delegated	✔ Can be represented in tokens (delegated authority, scopes, claims)	— Out of scope
② For whom it is delegated	✔ Can be represented in tokens (subject, actor, audience)	— Out of scope
③ Which workload performed the delegation	✖ Cannot be determined or verified	✔ Cryptographically verifiable

Agenda

1. Modern Platforms Have Multiple Security Authorities (Trust Domains)
2. What Breaks Federated Identity Across Trust Domains
3. How to Securely Build Federated Identity Across Trust Domains
4. Demo: Federated Identity in Action with Keycloak and SPIRE

What Is Keycloak?

- Keycloak is IAM (Identity and Access Management) OSS.
- Keycloak provides OAuth2 authorization server features and single sign-on features.
- Keycloak is a CNCF incubating project.

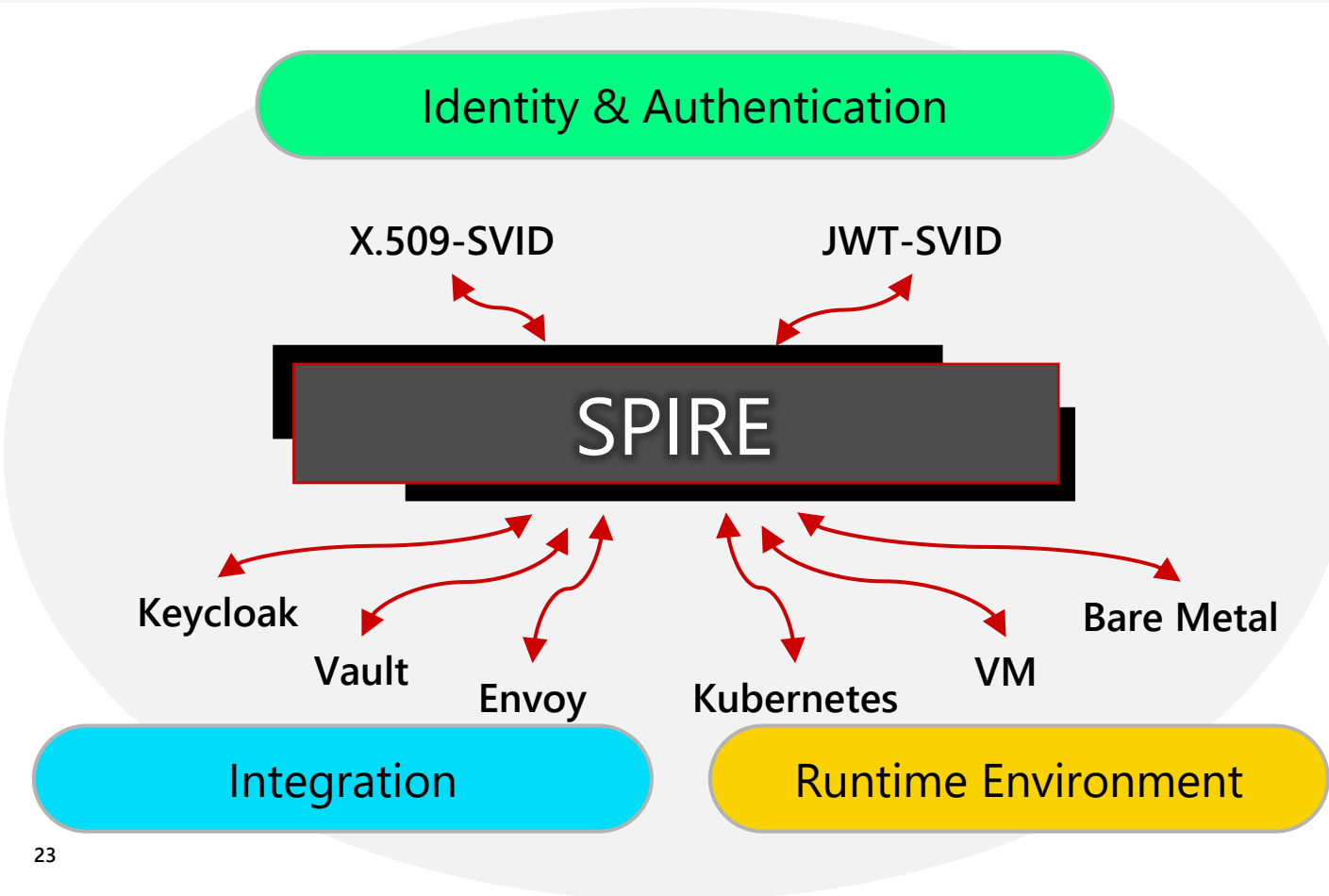


Major features

- Supporting standards. ex. OAuth 2.0, OpenID Connect 1.0, SAML v2, ...
 - Including:
 - [OAuth 2.0 Token Exchange \(RFC8693\)](#)
 - [JWT Profile for OAuth 2.0 Authorization Grants \(RFC7523\)](#)
 - [OAuth Identity and Authorization Chaining Across Domains \(draft-ietf-oauth-identity-chaining-08\)](#)
 - [OAuth SPIFFE Client Authentication \(draft-ietf-oauth-spiFFE-client-auth-00\)](#)
- Social Login. ex. GitHub, X, Facebook, ...
- Connect to existing user stores. ex. LDAP, Active Directory, ...

What Is SPIRE?

- SPIRE is an open source system for workload identity management.
- SPIRE implements the SPIFFE standard and issues SVIDs to workloads.
- SPIRE is a CNCF graduated project.

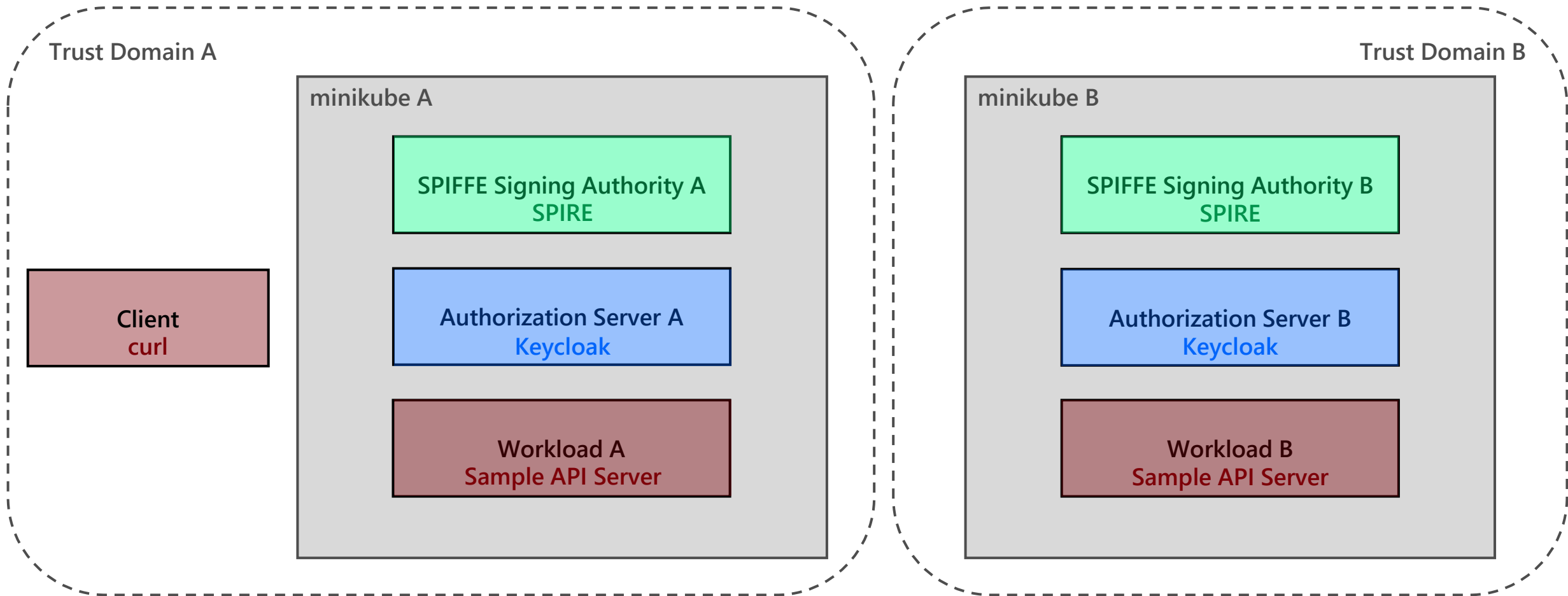


Major features

- Implements SPIFFE standard.
- Issue & Verify X.509-SVID and JWT-SVID.
- Integrate with CNCF tools. ex. Keycloak, Vault, Envoy, ...
- Support multiple runtime environments. ex. Kubernetes, VM, Bare Metal, ...

Demo Overview

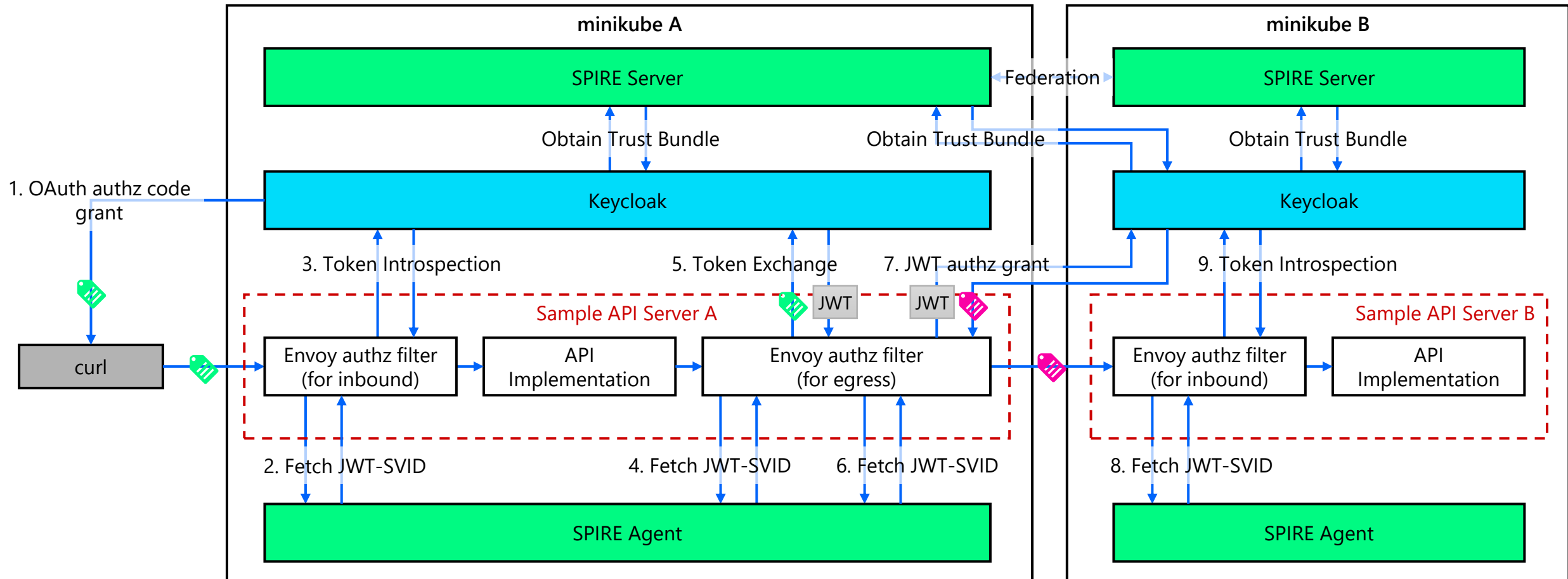
Two Kubernetes clusters represent separate trust domains, with Keycloak and SPIRE forming the basis of federated identity.



Demo Overview

OAuth authorization, token exchange, and SPIFFE-based client authentication are combined in a concrete Kubernetes request flow.

Demo assets are available: <https://github.com/Hitachi/oss-assets/tree/main/sessions/2026/kubecon-eu/spiffe-meets-oauth>



Key Takeaways

- **Consider the Identity Chaining pattern for propagating user identity across trust domains.**
 - ✓ Identity chaining forms the foundation for user delegation across trust domains, providing the basis for expressing what is delegated and for whom.
- **Use SPIFFE SVIDs for OAuth client authentication to eliminate static secrets.**
 - ✓ SPIFFE-based client authentication provides cryptographically verifiable workload identity, making explicit which workload actually performs delegation steps in OAuth flows.
- **Keycloak and SPIRE provide a practical open-source foundation for federated identity today.**
 - ✓ Together, they enable a realistic composition: OAuth for user delegation semantics and SPIFFE client authentication for workload attribution across trust domains.



feedback

Trademarks

- OpenID is a trademark or registered trademark of OpenID Foundation in the United States and other countries.
- GitHub is a trademark or registered trademark of GitHub, Inc. in the United States and other countries.
- X is a trademark or registered trademark of X Corp. in the United States and other countries.
- Other brand names and product names used in this material are trademarks, registered trademarks, or trade names of their respective holders.

Thank you

KubeCon + CloudNativeCon Europe 2026

SPIFFE Meets OAuth: Federated Identity for Cloud Native Workloads

Yoshiyuki Tabata

Hitachi OSPO

March 26, 2026

HITACHI