

KeyConf 26

@Holiday Inn Prague, Prague, Czech Republic 

Keycloak Becomes Familiar with AI: The Advancement of Integrating Keycloak with AI

Takashi Norimatsu

Open Source Program Office (OSPO)

Hitachi, Ltd.

Atul Ram

Software Development Engineer – Specialist

Hitachi Vantara LLC

Date

October 8, 2026

Self Introduction

Takashi Norimatsu Ph.D. (**tnorimat** in GitHub)

- Keycloak maintainer (since Oct 2021)
- MCP project maintainer (conformance) (since Oct 2026)
- Technical lead of Keycloak community "OAuth SIG"
- Certified Information Systems Security Professional (CISSP)
- Chief OSS Specialist, Hitachi, Ltd., Japan 

Self Introduction

Contributing security features to Keycloak since 2017:

- W3C Web Authentication API support (Passkeys)
- IETF standards:
RFC 7636 PKCE, RFC 8705 OAuth MTLS, RFC 9126 PAR, RFC 8032/8037 EdDSA,
RFC 9207 OAuth2 Authz Server Issuer Identification, RFC 9449 DPoP,
RFC 8414 OAuth 2.0 Authorization Server Metadata
Internet Draft: The OAuth 2.1 Authorization Framework
Internet Draft: OAuth Client ID Metadata Document (CIMD)
- OpenID Foundation standards:
FAPI 1.0 Baseline, FAPI 1.0 Advanced,
FAPI 2.0 Security Profile, FAPI 2.0 Message Signing,
Client Initiated Backchannel Authentication, FAPI-CIBA
- Agentic AI Foundation (AAIF) Model Context Protocol (MCP) standards:
MCP 2025-06-18, 2025-11-25, 2026-07-28

Self Introduction

Contributing MCP conformance tests for Authorization Server since 2026:

- MCP: conformance
MCP 2025-11-28, EMA/ID-JAG (in progress)

Self Introduction

Atul Ram ([atulram](#) in GitHub)

- Software Development Engineer – Specialist, Hitachi Vantara
- Builds authentication and authorization for iQ Studio, Hitachi Vantara's GenAI agent platform
- Built the platform's MCP authorization broker: OAuth 2.1 with PKCE, CIMD and DCR for MCP clients
- 9+ years across Python, Go and Node.js, focused on IAM, OAuth 2.x and MCP

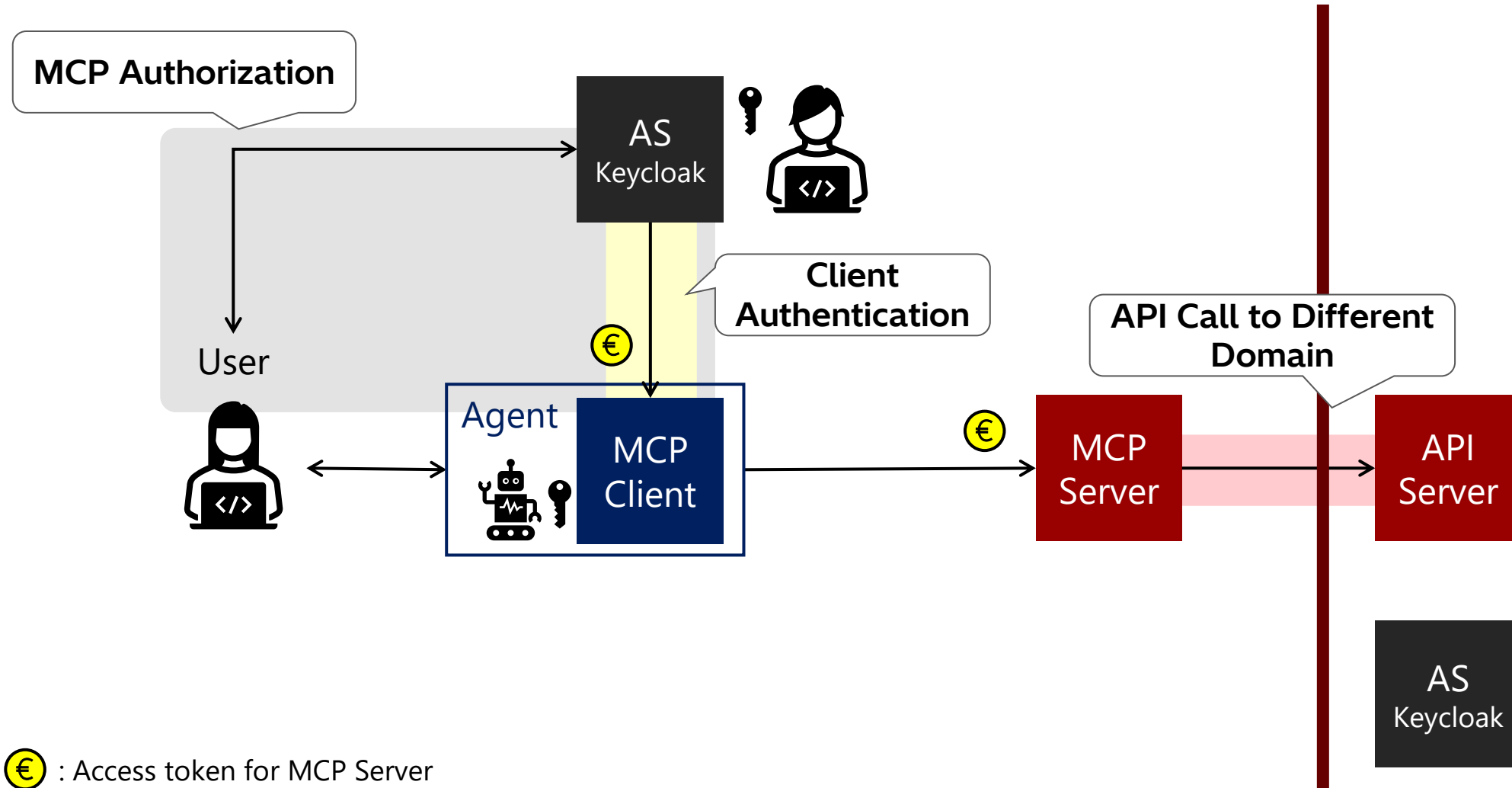
Contents

1. What features are beneficial for Agentic AI Authentication and Authorization
2. MCP Authorization
3. Cross-Domain API Call
4. Secret-less Workload Authentication
5. Demonstration

Contents

1. What features are beneficial for Agentic AI Authentication and Authorization
2. MCP Authorization
3. Cross-Domain API Call
4. Secret-less Workload Authentication
5. Demonstration

What features are beneficial for Agentic AI AuthN & AuthZ



Contents

1. What features are beneficial for Agentic AI Authentication and Authorization
2. MCP Authorization
3. Cross-Domain API Call
4. Secret-less Workload Authentication
5. Demonstration

MCP Authorization Standards Compliance^{*1}

Standard	2026-07-28	2025-11-25	2025-06-18	2025-03-26	Keycloak 26.8
Internet Draft: OAuth 2.1 Authorization Framework	MUST	MUST	MUST	MUST	Supported
RFC 8414 OAuth 2.0 Authorization Server Metadata	MUST	MUST	MUST	MUST	Supported
RFC 8707 Resource Indicators for OAuth 2.0	MUST?	MUST?	MUST?	-	Experimental
RFC 7591 OAuth 2.0 Dynamic Client Registration Protocol (DCR)	MAY	MAY	SHOULD	SHOULD	Supported
Internet Draft: OAuth Client ID Metadata Document (CIMD)	SHOULD	SHOULD	-	-	Experimental
RFC 9207 OAuth 2.0 Authorization Server Issuer Identification	SHOULD	-	-	-	Supported

MCP Revision	Keycloak 26.8
2025-03-26	Supported
2025-06-18	Experimental ^{*1}
2025-11-25	Experimental
2026-07-28	Experimental

^{*1}: <https://www.keycloak.org/securing-apps/mcp-authz-server>

^{*2}: Because it includes experimental feature.

MCP Authorization Extensions^{*1} Standards Compliance

Standard			Keycloak 26.8
Stable: MCP Enterprise-managed Authorization (EMA) ^{*2}	Internet Draft: Identity Assertion JWT Authorization Grant (ID-JAG) ^{*3}	as Resource Authorization Server (AS)	Experimental
		as IdP Authorization Server (IdP)	Not supported
Draft: OAuth Client Credentials Extension ^{*4}			Not supported ^{*5}
Draft: DPoP Profile for MCP ^{*6}			Supported
RFC 9449 OAuth 2.0 Demonstrating Proof of Possession (DPoP) ^{*7}			Supported
Draft: Workload Identity Federation (WIF) ^{*8}			Not supported ^{*9}
RFC 7523 JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants ^{*10}	Using JWTs as Authorization Grants		Supported

*1: <https://github.com/modelcontextprotocol/ext-auth/>

*2: <https://github.com/modelcontextprotocol/ext-auth/blob/main/specification/stable/enterprise-managed-authorization.md>

*3: <https://www.ietf.org/archive/id/draft-ietf-oauth-identity-assertion-authz-grant-04.html>

*4: <https://github.com/modelcontextprotocol/ext-auth/blob/main/specification/draft/oauth-client-credentials.md>

*5: not support "resource" parameter in client credentials grant

*6: <https://github.com/modelcontextprotocol/ext-auth/blob/pieterkas-dpop-extension/specification/draft/dpop-extension.md>

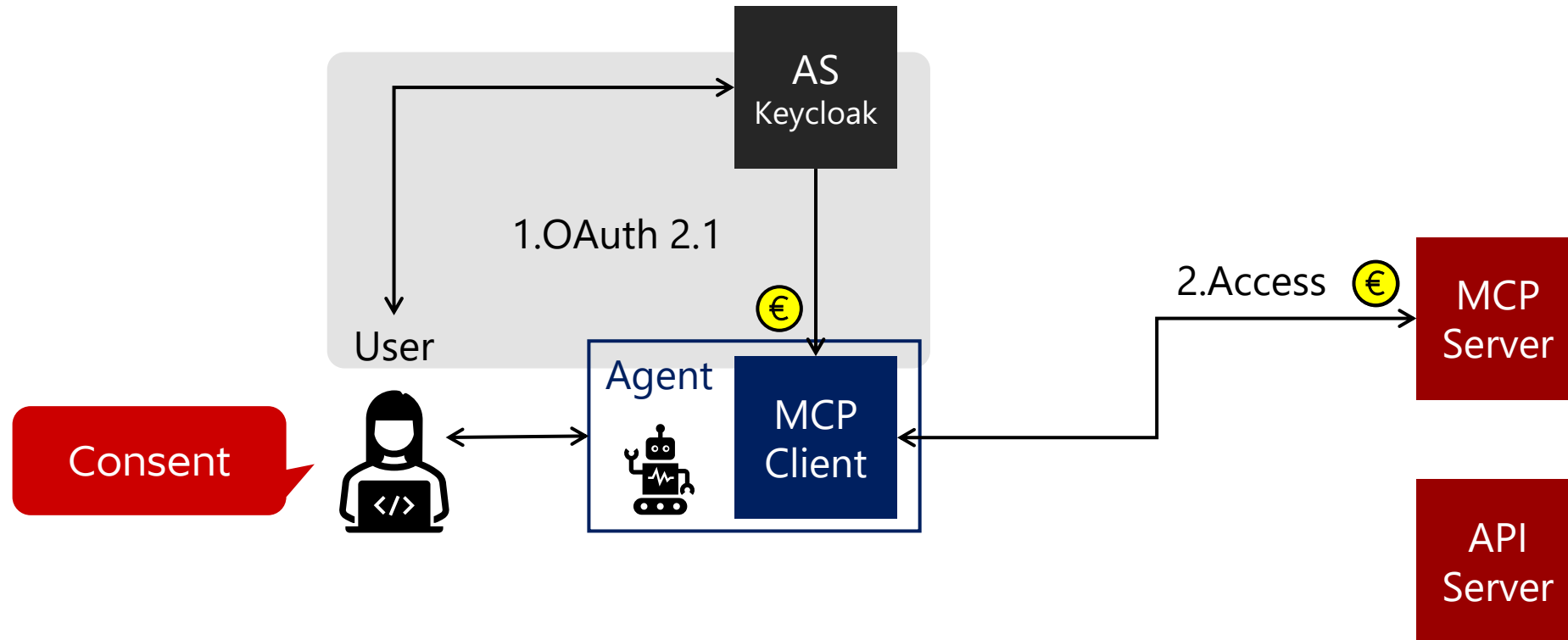
*7: <https://datatracker.ietf.org/doc/html/rfc9449>

*8: <https://github.com/modelcontextprotocol/ext-auth/blob/pieterkas-wif-extension/specification/draft/workload-identity-federation.md>

*9: Keycloak recognizes "sub" claim as a user identity while the spec recognizes "sub" claim as a workload identity.

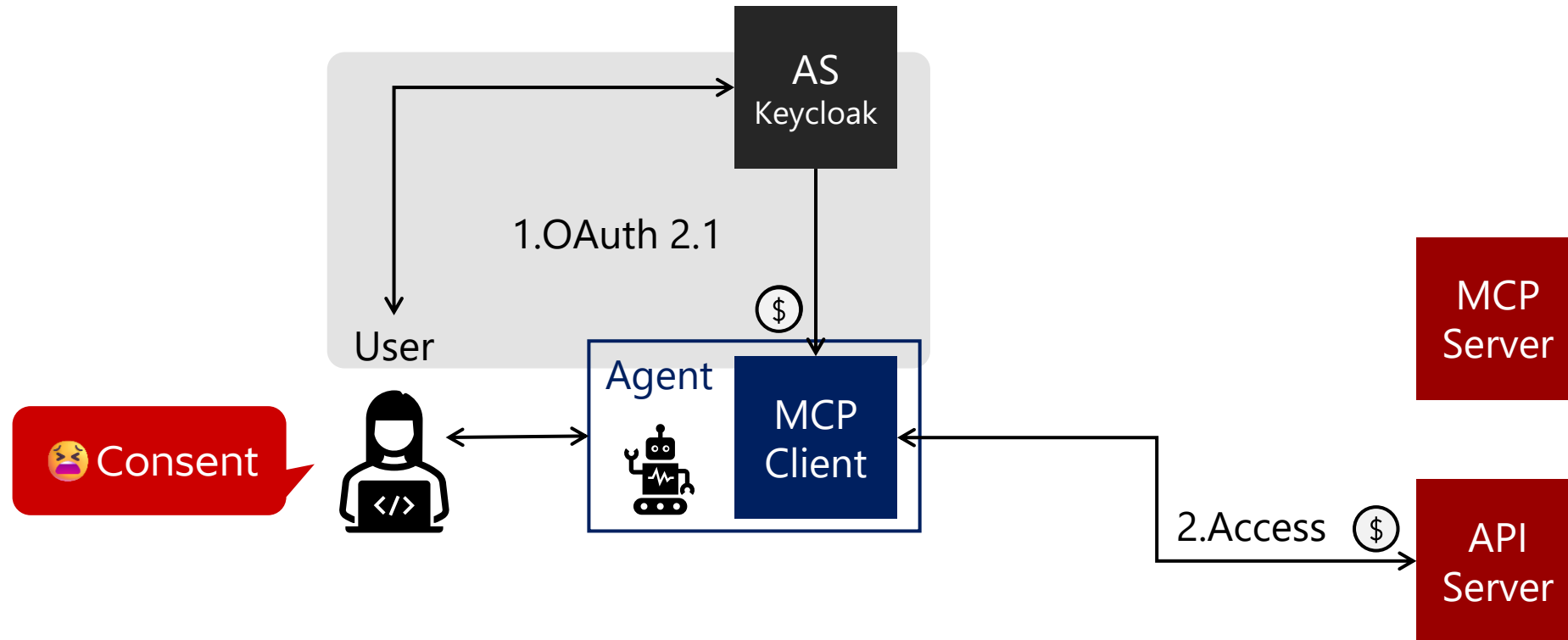
*10: <https://datatracker.ietf.org/doc/html/rfc7523>

EMA: Identity Assertion JWT Authorization Grant (ID-JAG)



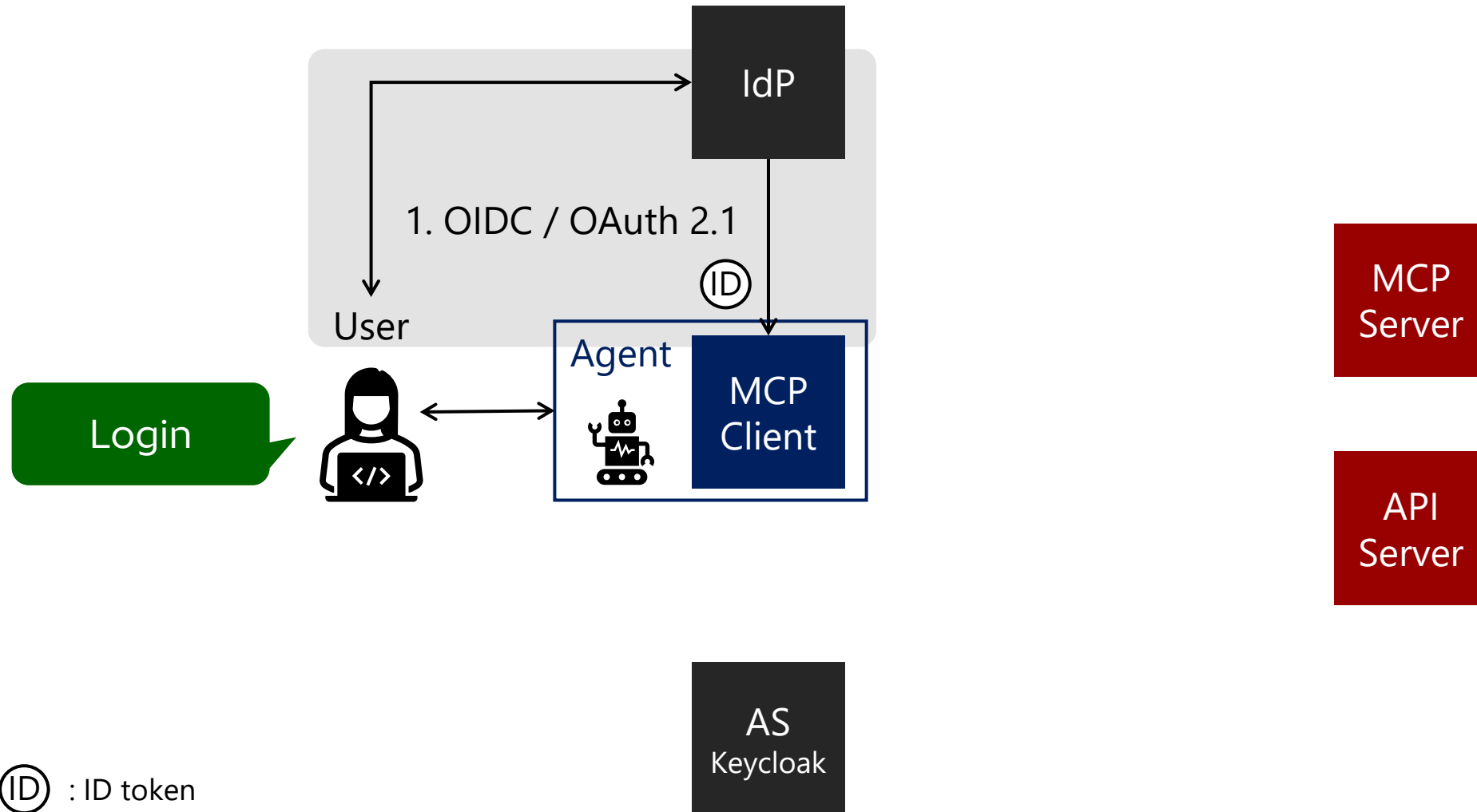
€ : Access token for MCP Server

EMA: Identity Assertion JWT Authorization Grant (ID-JAG)

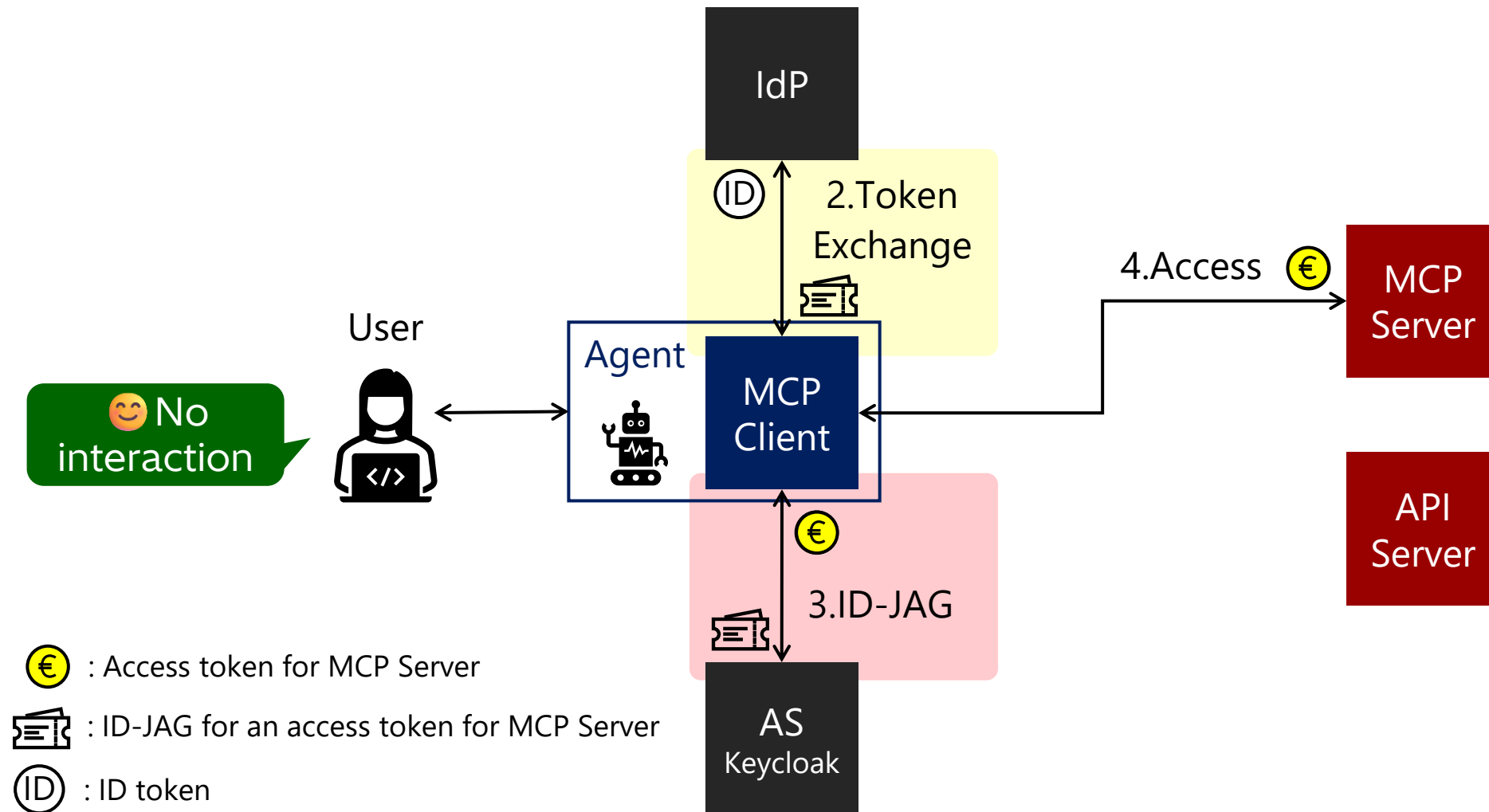


⌘ : Access token for API Server

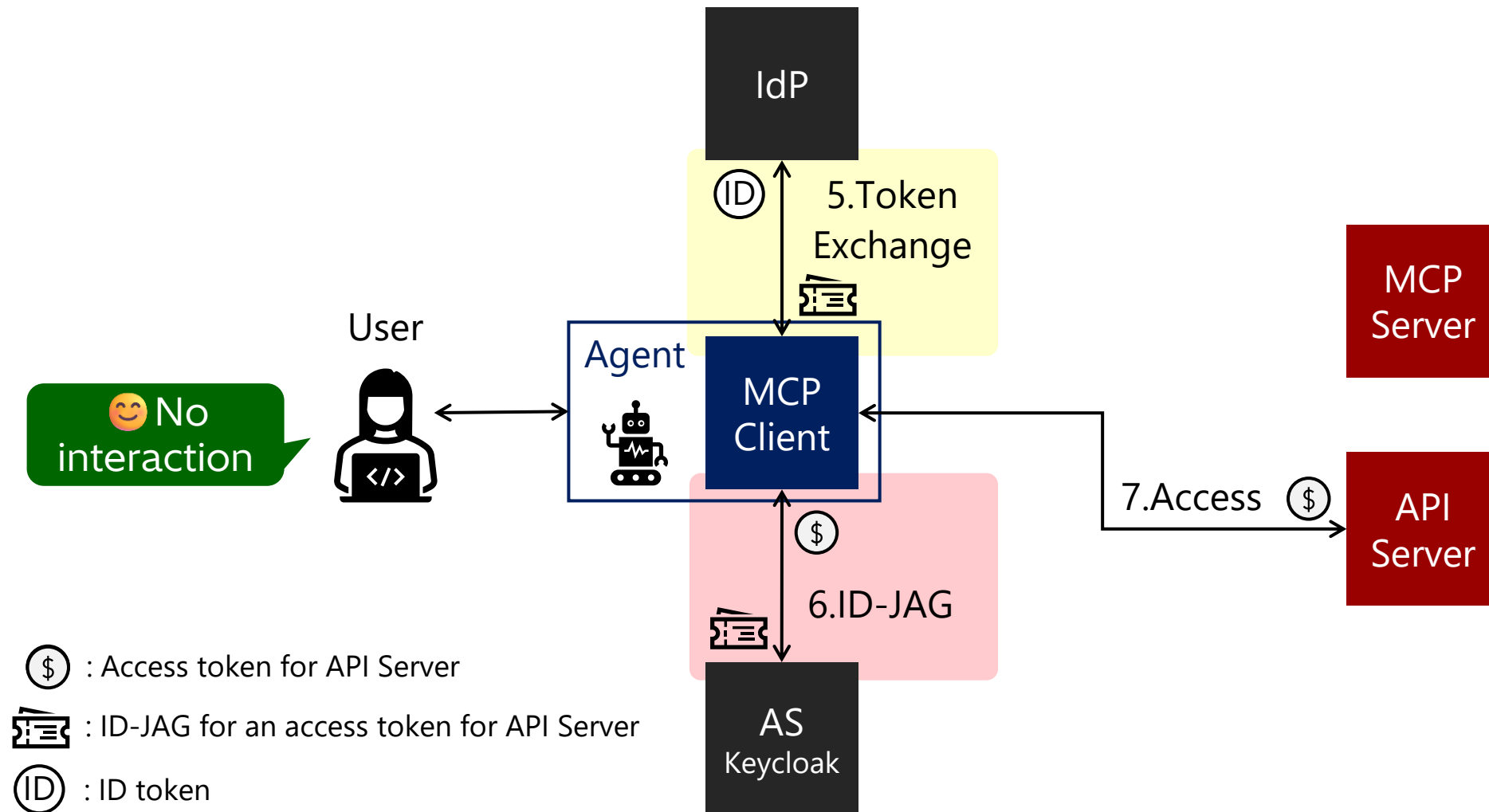
EMA: Identity Assertion JWT Authorization Grant (ID-JAG)



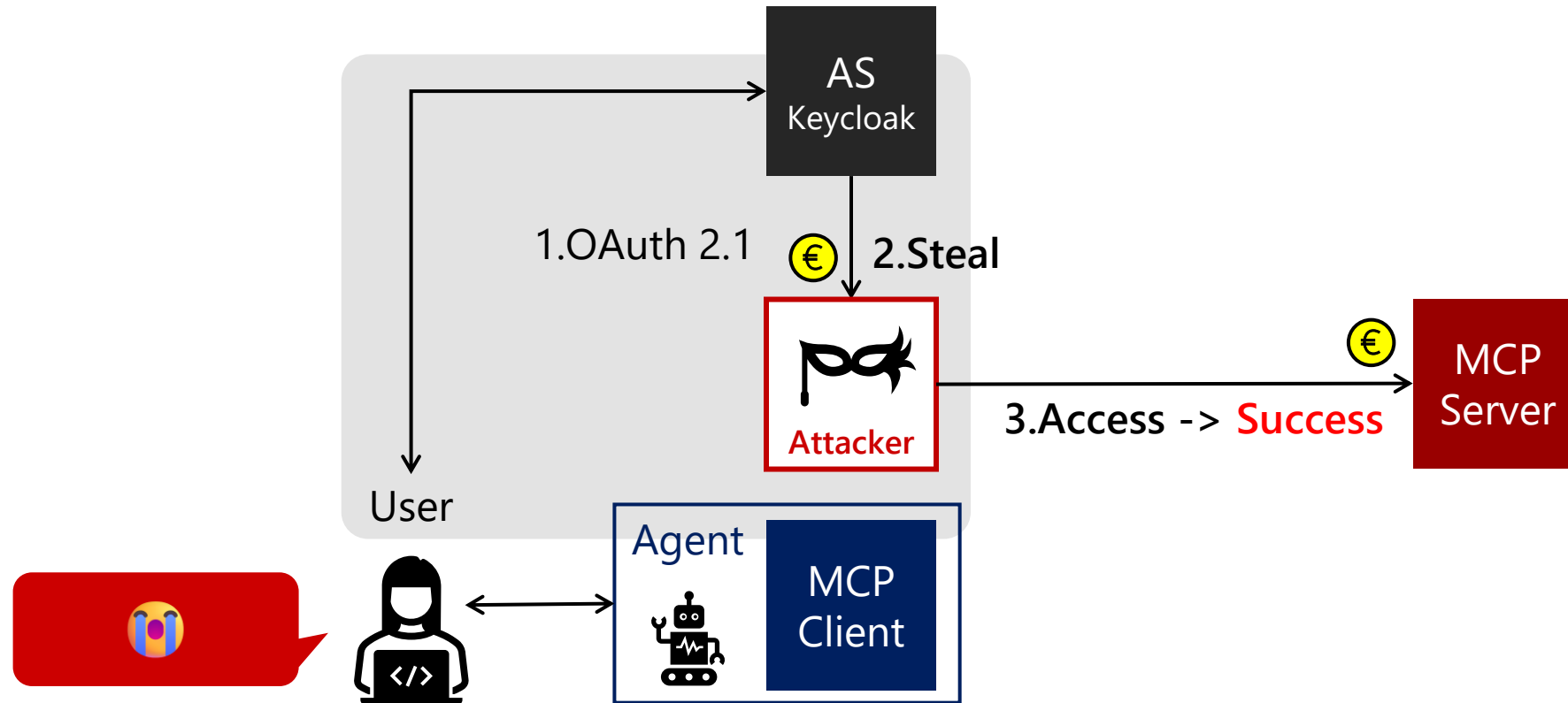
EMA: Identity Assertion JWT Authorization Grant (ID-JAG)



EMA: Identity Assertion JWT Authorization Grant (ID-JAG)

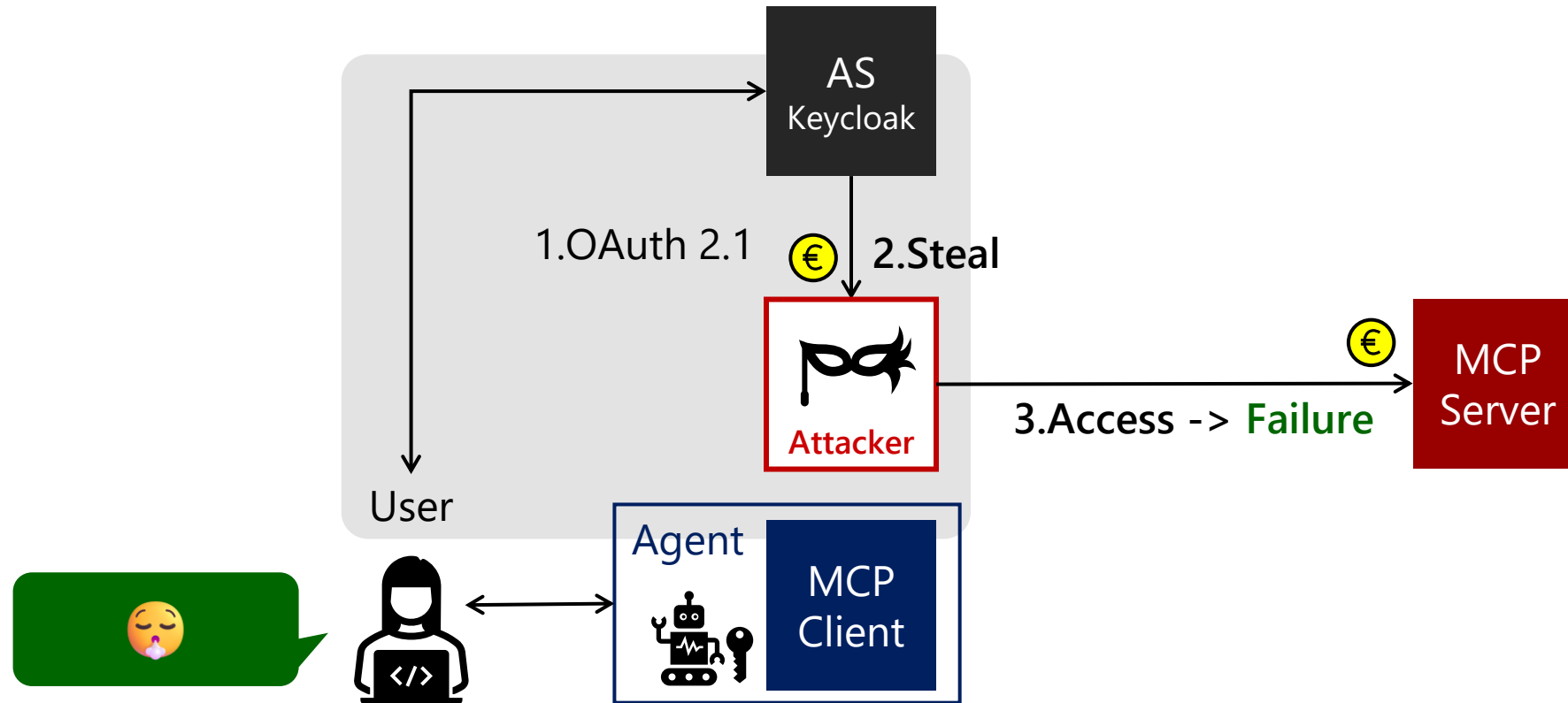


DPoP Profile for MCP



€ : Bearer access token for MCP Server

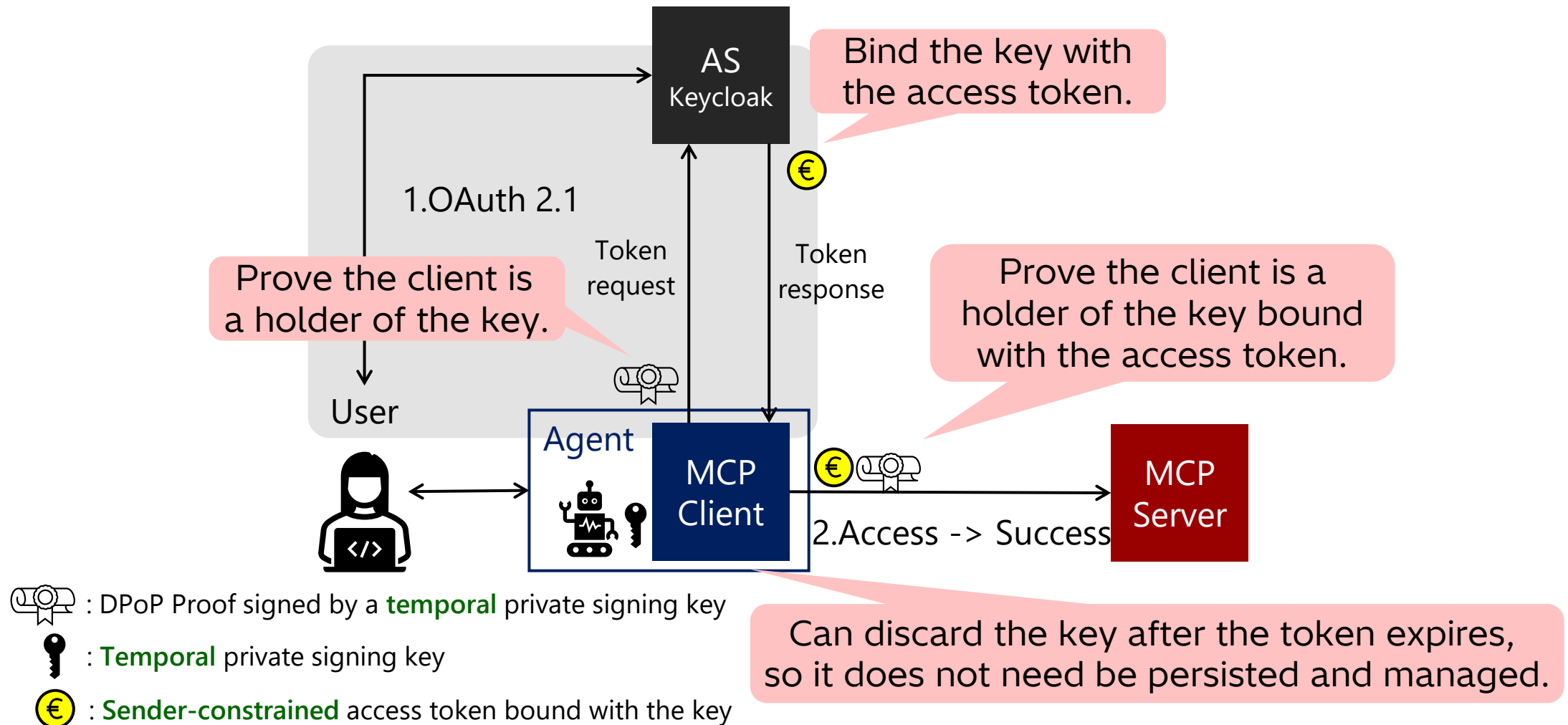
DPoP Profile for MCP



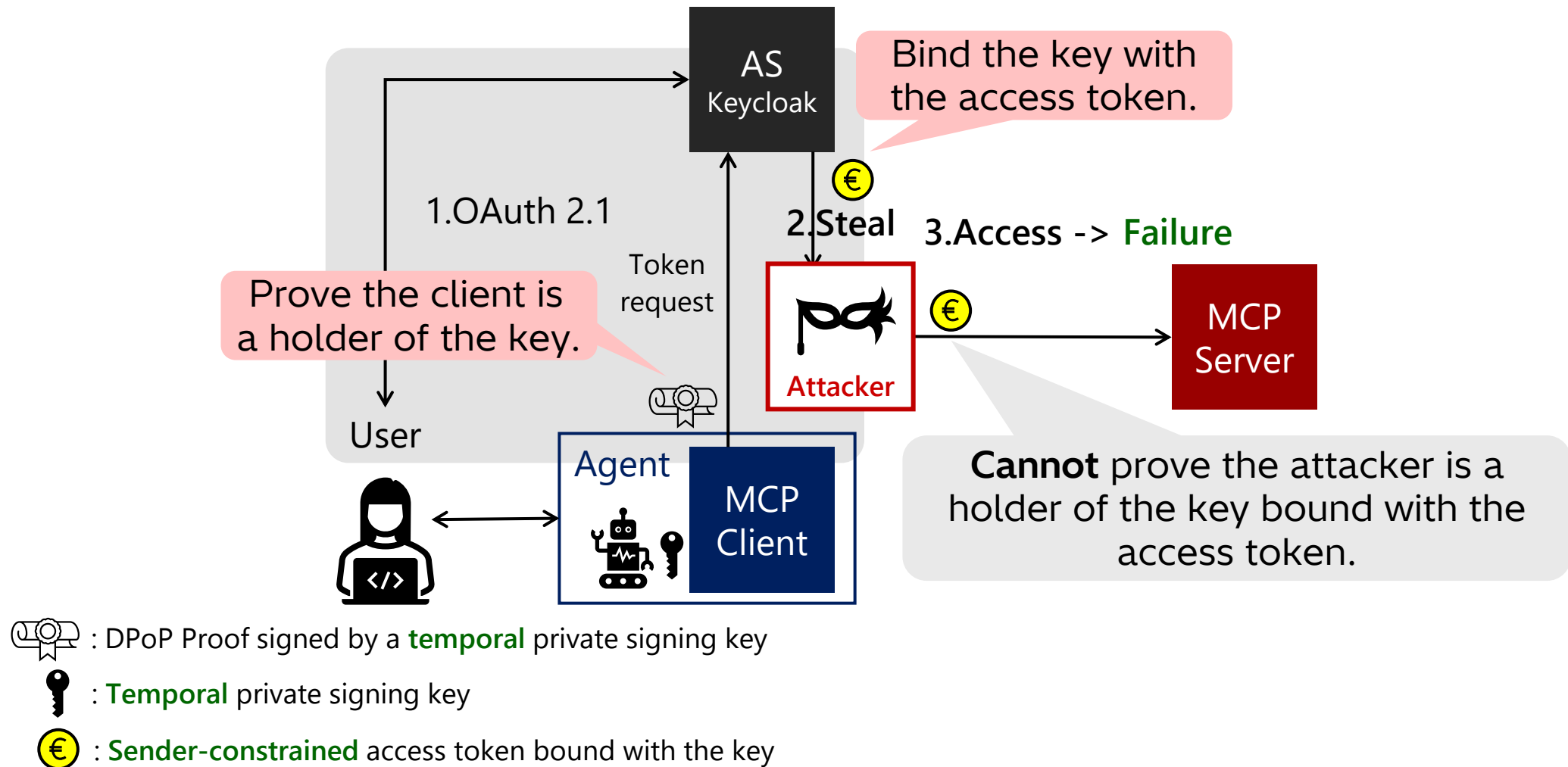
🔑 : Temporal private signing key

€ : Sender-constrained access token for MCP Server

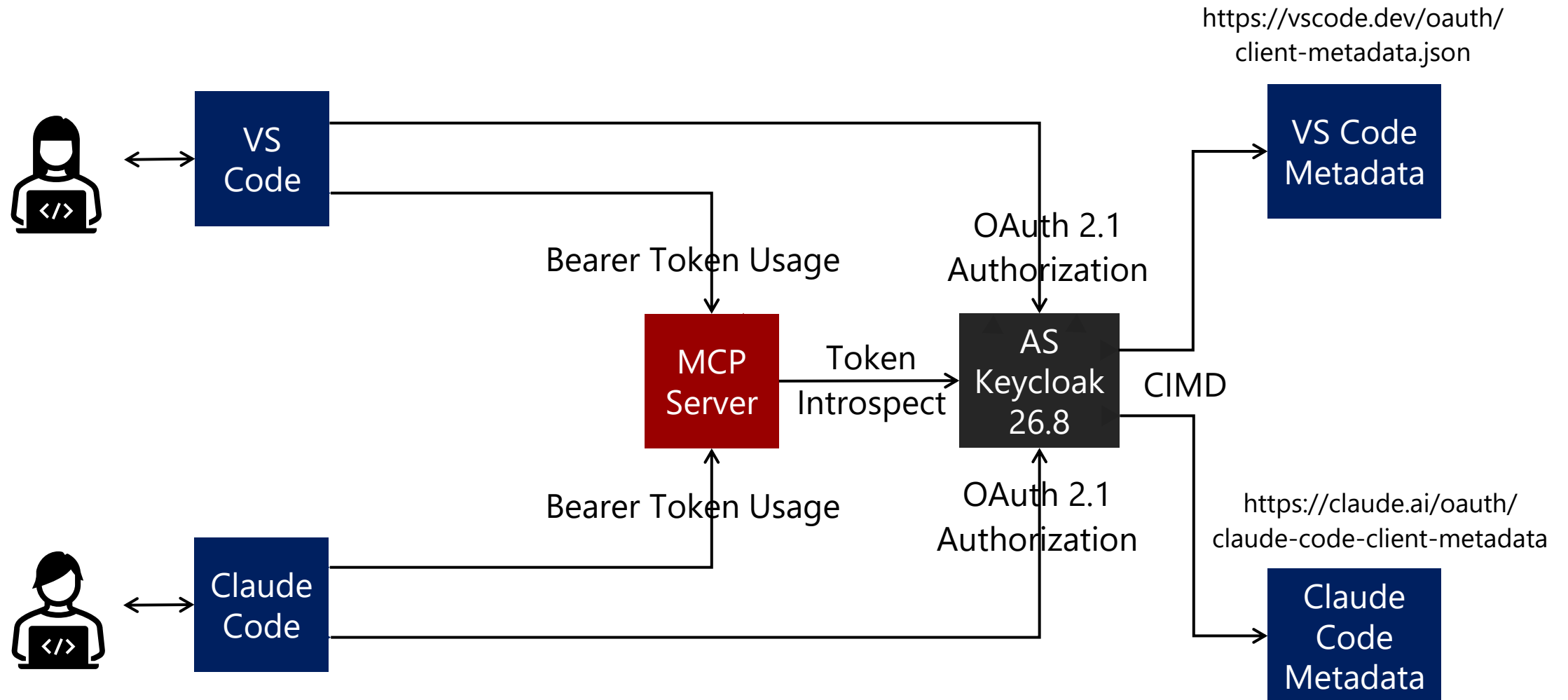
DPoP Profile for MCP



DPoP Profile for MCP



AI Tools Integration*



*: <https://www.keycloak.org/securing-apps/mcp-authz-server>

Contents

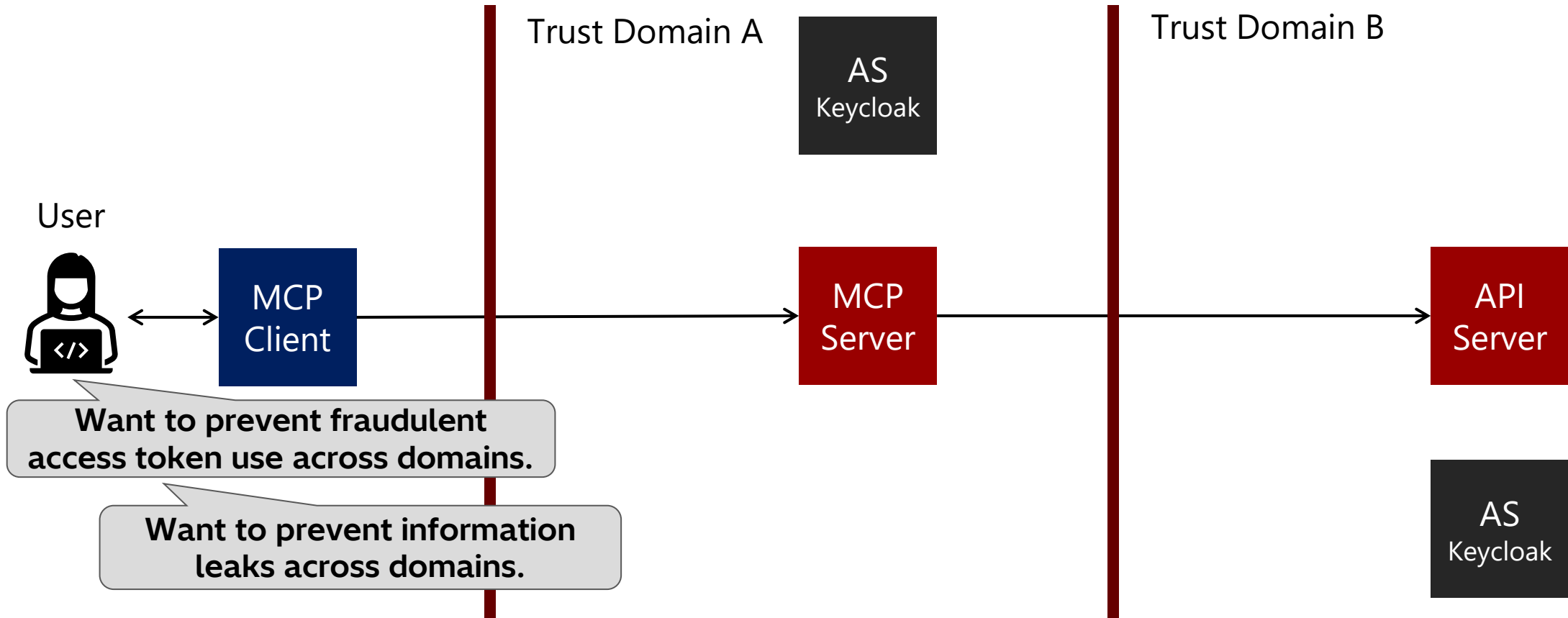
1. What features are beneficial for Agentic AI Authentication and Authorization
2. MCP Authorization
3. **Cross-Domain API Call**
4. Secret-less Workload Authentication
5. Demonstration

Cross-Domain API Calls

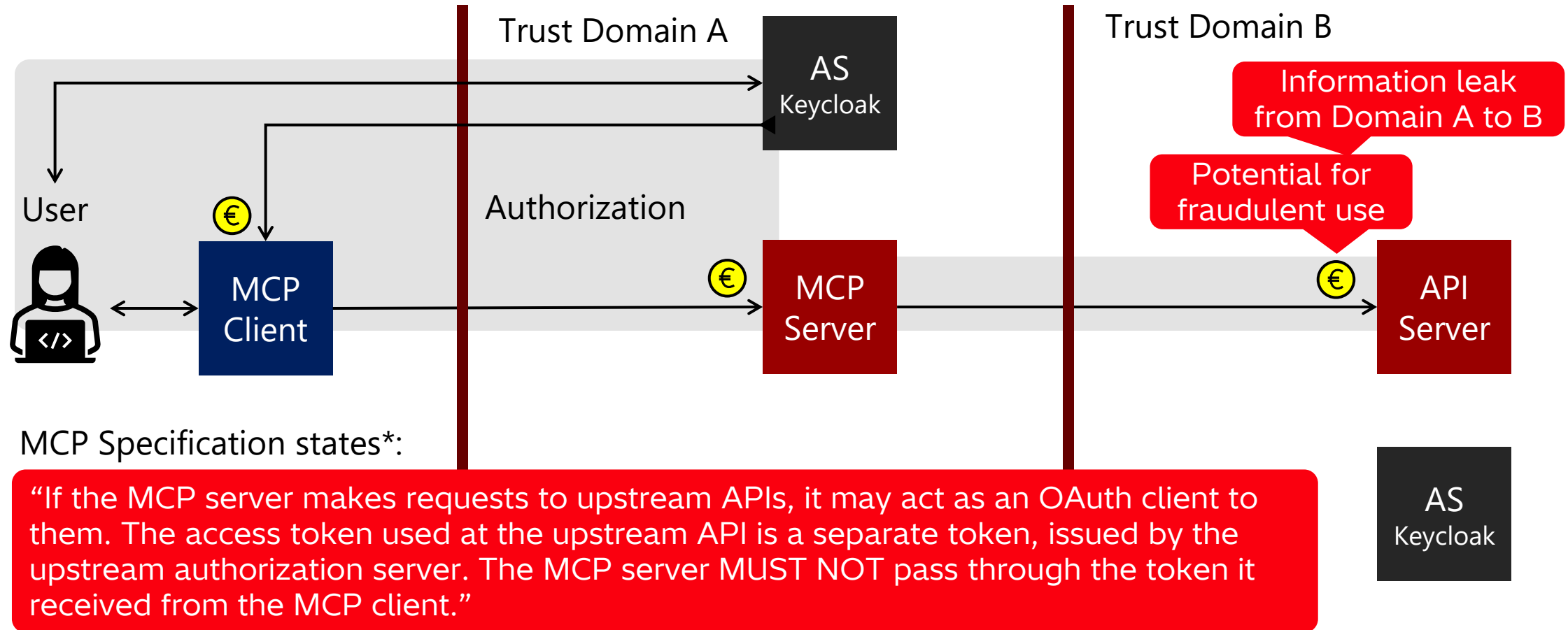
Standard / Feature		Keycloak 26.8
RFC 8693 OAuth 2.0 Token Exchange ^{*1}		Supported
RFC 7521 Assertion Framework for OAuth 2.0 Client Authentication and Authorization Grants ^{*2}	Using Assertions as Authorization Grants	Supported
RFC 7523 JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants ^{*3}	Using JWTs as Authorization Grants	Supported
Internet Draft: OAuth Identity and Authorization Chaining Across Domains ^{*4*5}		Supported
Feature: Lightweight Access Token ^{*6}		Supported

*1: <https://datatracker.ietf.org/doc/html/rfc8693>
*2: <https://datatracker.ietf.org/doc/html/rfc7521>
*3: <https://datatracker.ietf.org/doc/html/rfc7523>
*4: <https://datatracker.ietf.org/doc/draft-ietf-oauth-identity-chaining/>
*5: <https://www.keycloak.org/2026/01/jwt-authorization-grant>
*6: https://www.keycloak.org/docs/latest/server_admin/index.html#_using_lightweight_access_token

Cross-Domain API Calls



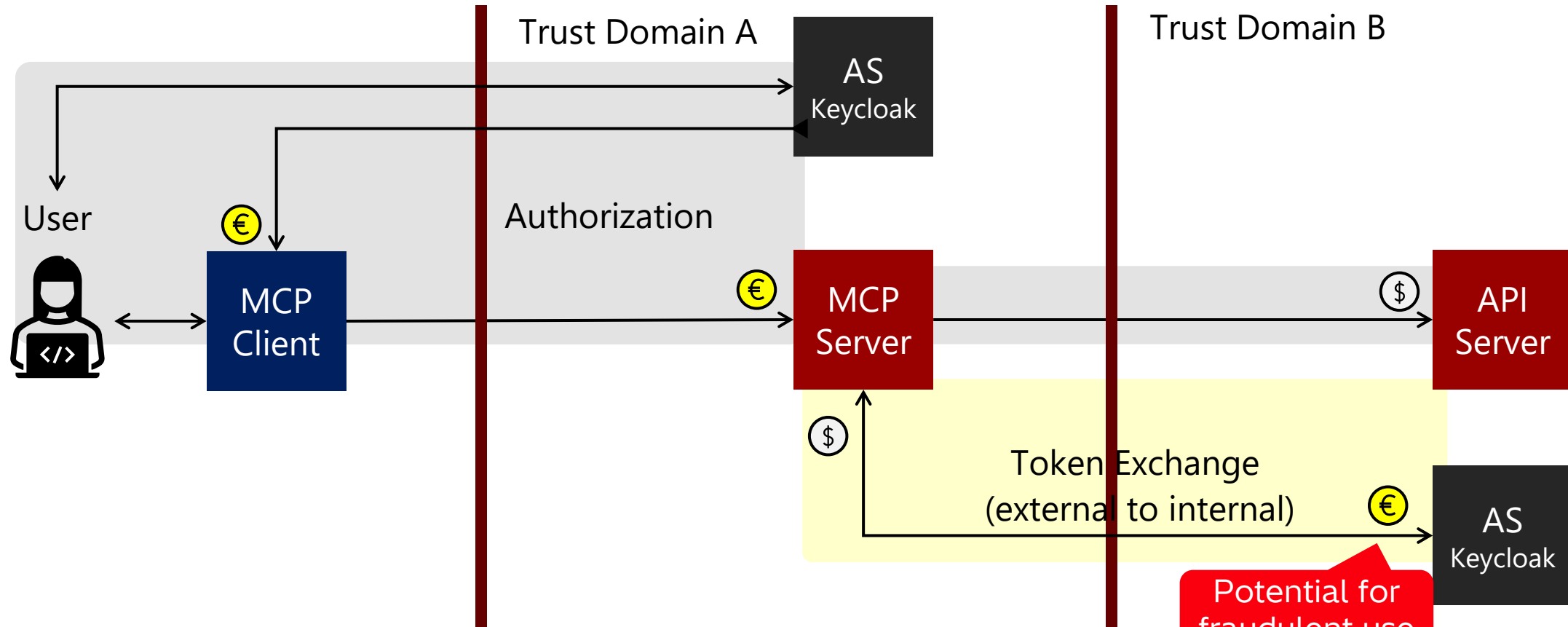
Forwarding



*: <https://modelcontextprotocol.io/specification/2026-07-28/basic/authorization/security-considerations#access-token-privilege-restriction>

€ : Access Token for MCP Server in Domain A

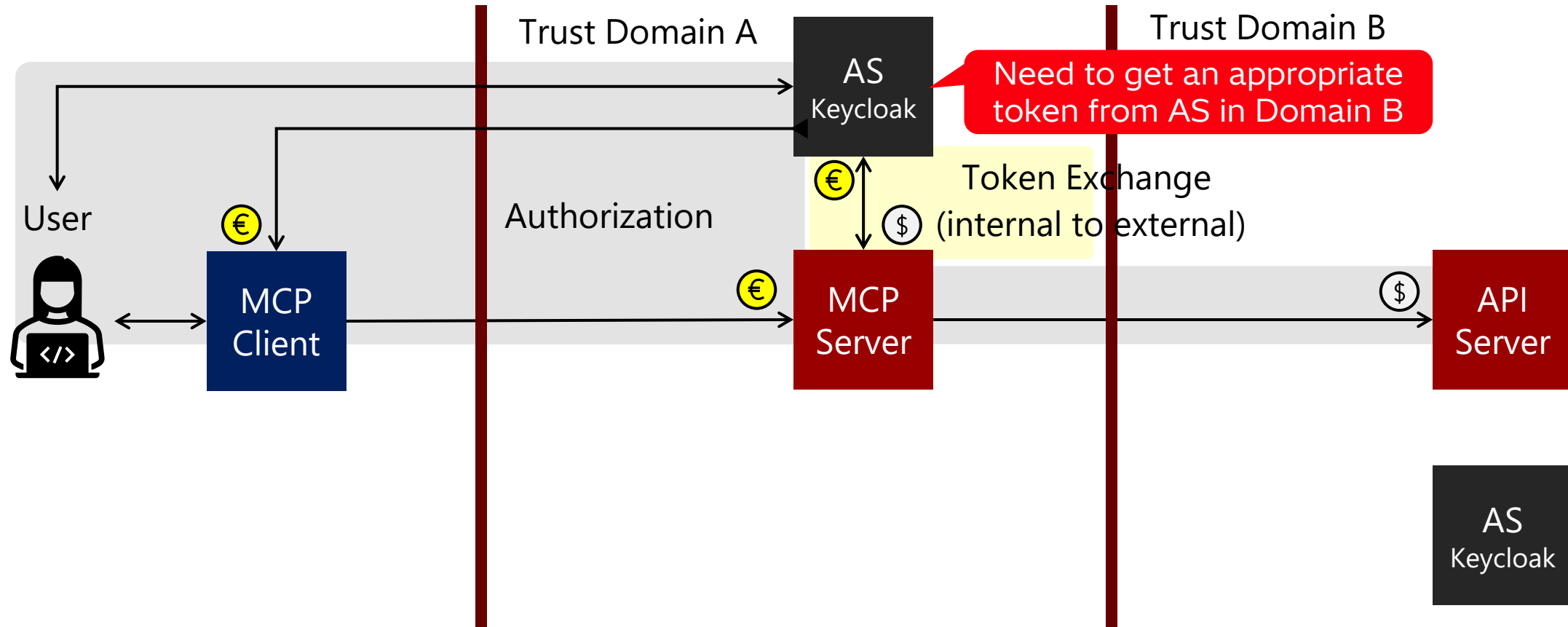
Token Exchange (External to Internal)



€ : Access Token for MCP Server in Domain A

\$: Access Token for API Server in Domain B

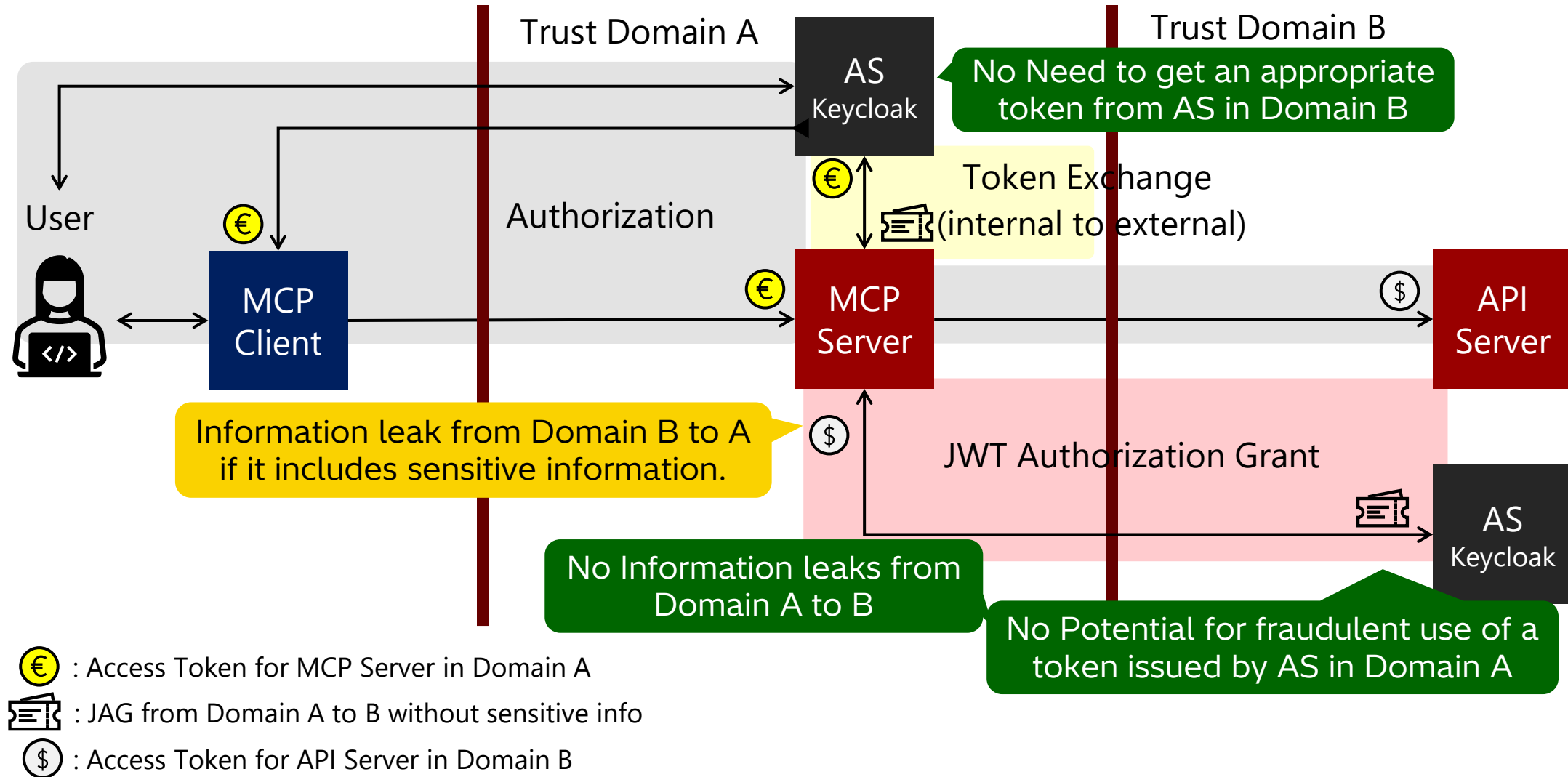
Token Exchange (Internal to External)



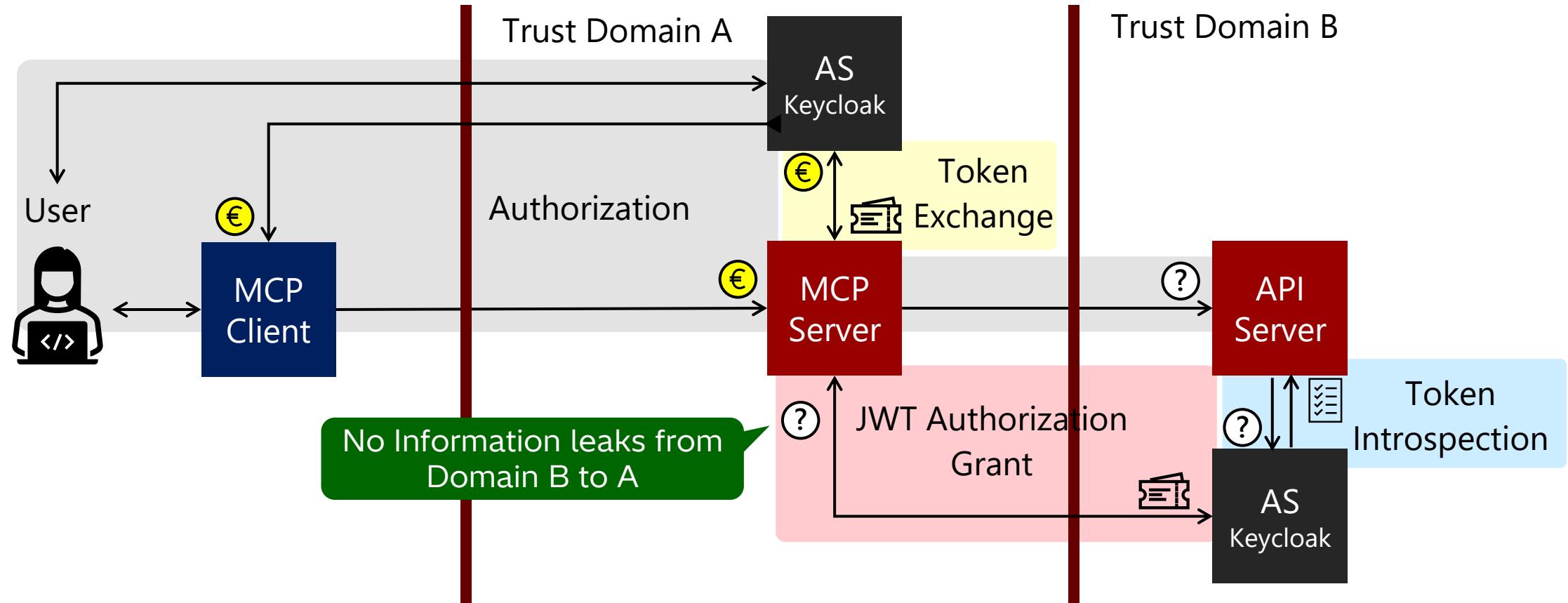
€ : Access Token for MCP Server in Domain A

\$: Access Token for API Server in Domain B

OAuth Identity and Authorization Chaining Across Domains



+ Lightweight Access Token



€ : Access Token for MCP Server in Domain A

📄 : JAG from Domain A to B without sensitive info 📋 : Identity and authorization sensitive info in Domain B

Ⓜ : Lightweight Access Token for API Server in Domain B with only token metadata

Contents

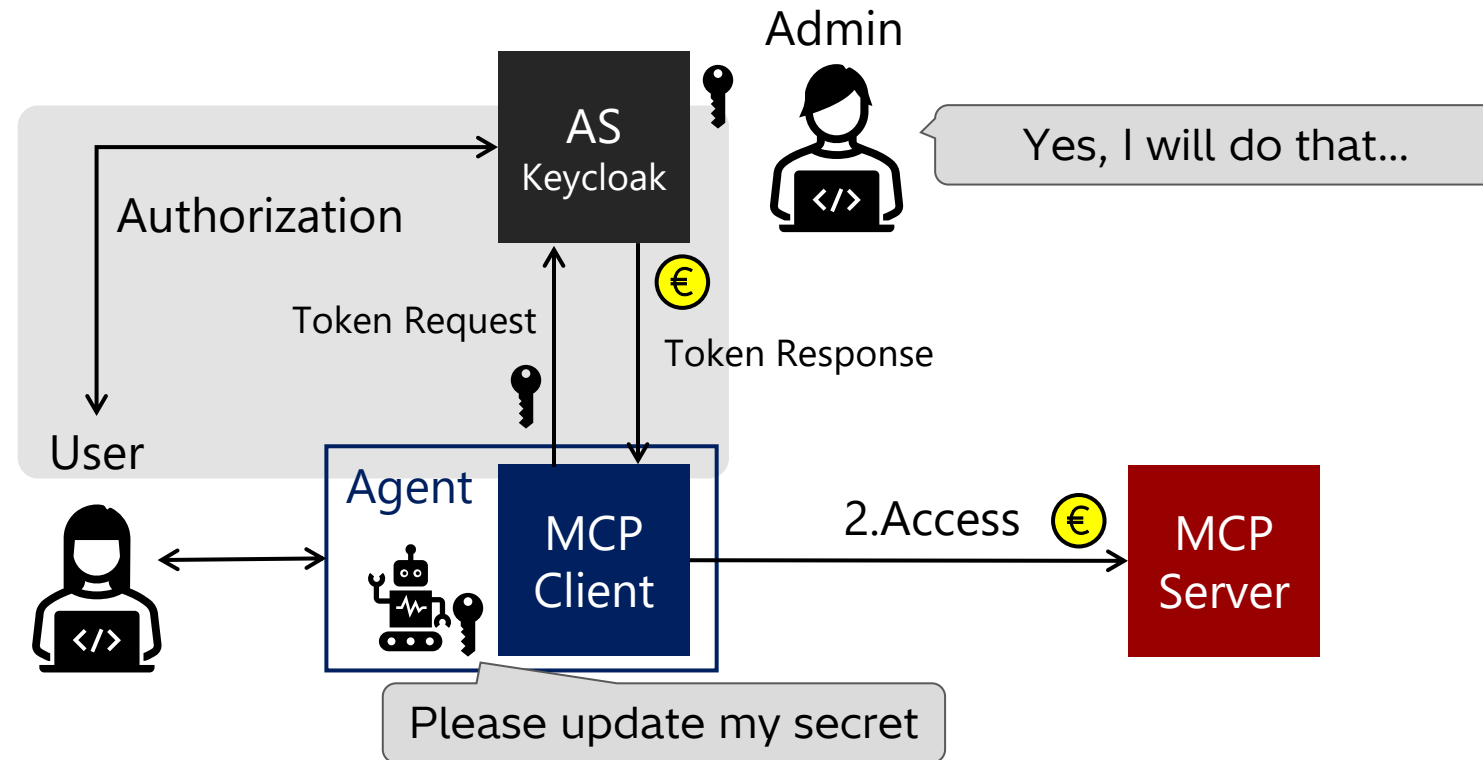
1. What features are beneficial for Agentic AI Authentication and Authorization
2. MCP Authorization
3. Cross-Domain API Call
- 4. Secret-less Workload Authentication**
5. Demonstration

Secret-less Workload Authentication

Standard / Feature		Keycloak 26.8
RFC 7521 Assertion Framework for OAuth 2.0 Client Authentication and Authorization Grants* ¹	Using Assertions for Client Authentication	Supported
RFC 7523 JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants* ²	Using JWTs for Client Authentication	Supported
Feature: Federated Client Authentication* ³	OpenID Connect* ⁴	Supported
	Kubernetes Service Accounts* ⁵	Supported
	SPIFFE JWT SVID* ⁶ - Internet Draft: OAuth SPIFFE Client Authentication* ⁷	Preview

*1: <https://datatracker.ietf.org/doc/html/rfc7521>
*2: <https://datatracker.ietf.org/doc/html/rfc7523>
*3: <https://www.keycloak.org/2026/01/federated-client-authentication>
*4: https://www.keycloak.org/docs/latest/server_admin/index.html#_identity_broker_oidc
*5: https://www.keycloak.org/docs/latest/server_admin/index.html#_identity_broker_kubernetes
*6: https://www.keycloak.org/docs/latest/server_admin/index.html#_identity_broker_spiffe
*7: <https://datatracker.ietf.org/doc/draft-ietf-oauth-spiffe-client-auth/>

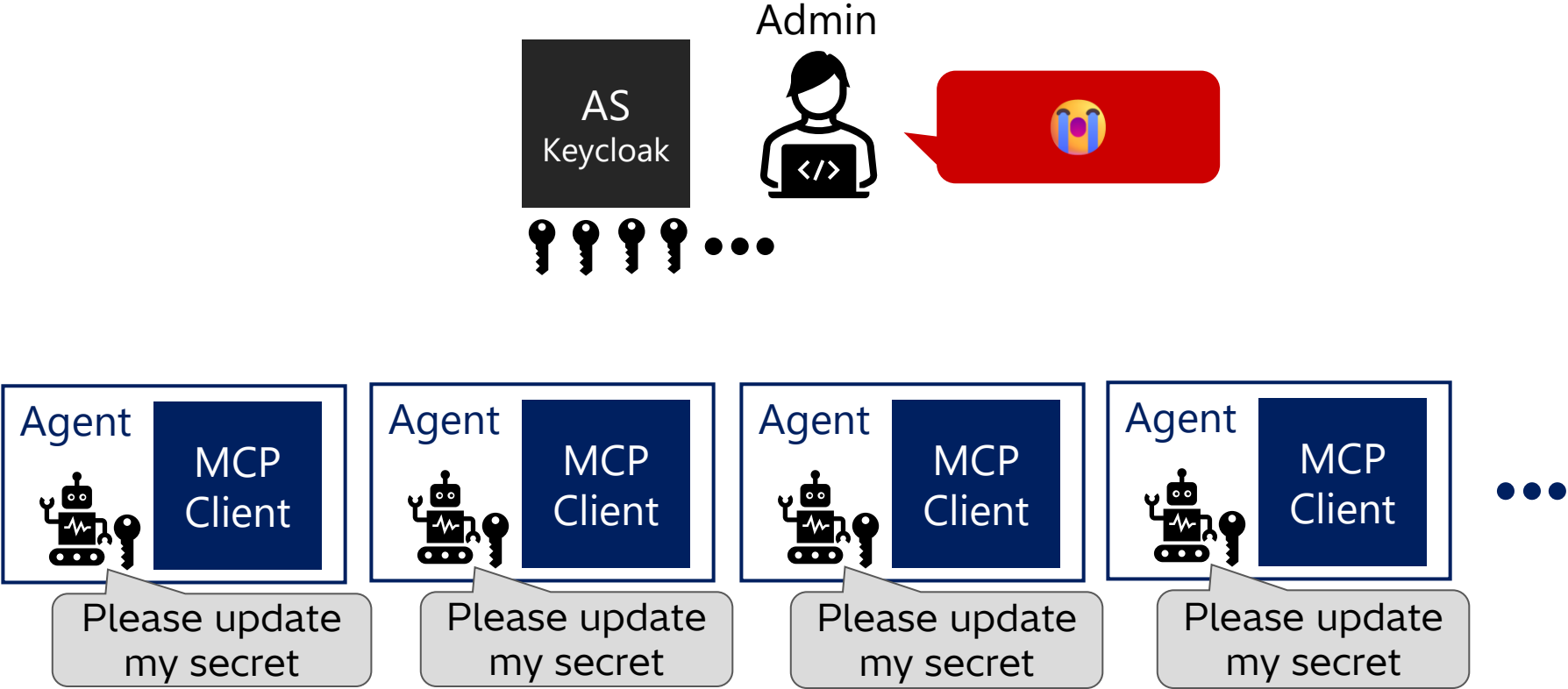
Secret-less Workload Authentication



🔑 : Client secret for client authentication

€ : Access token for MCP Server

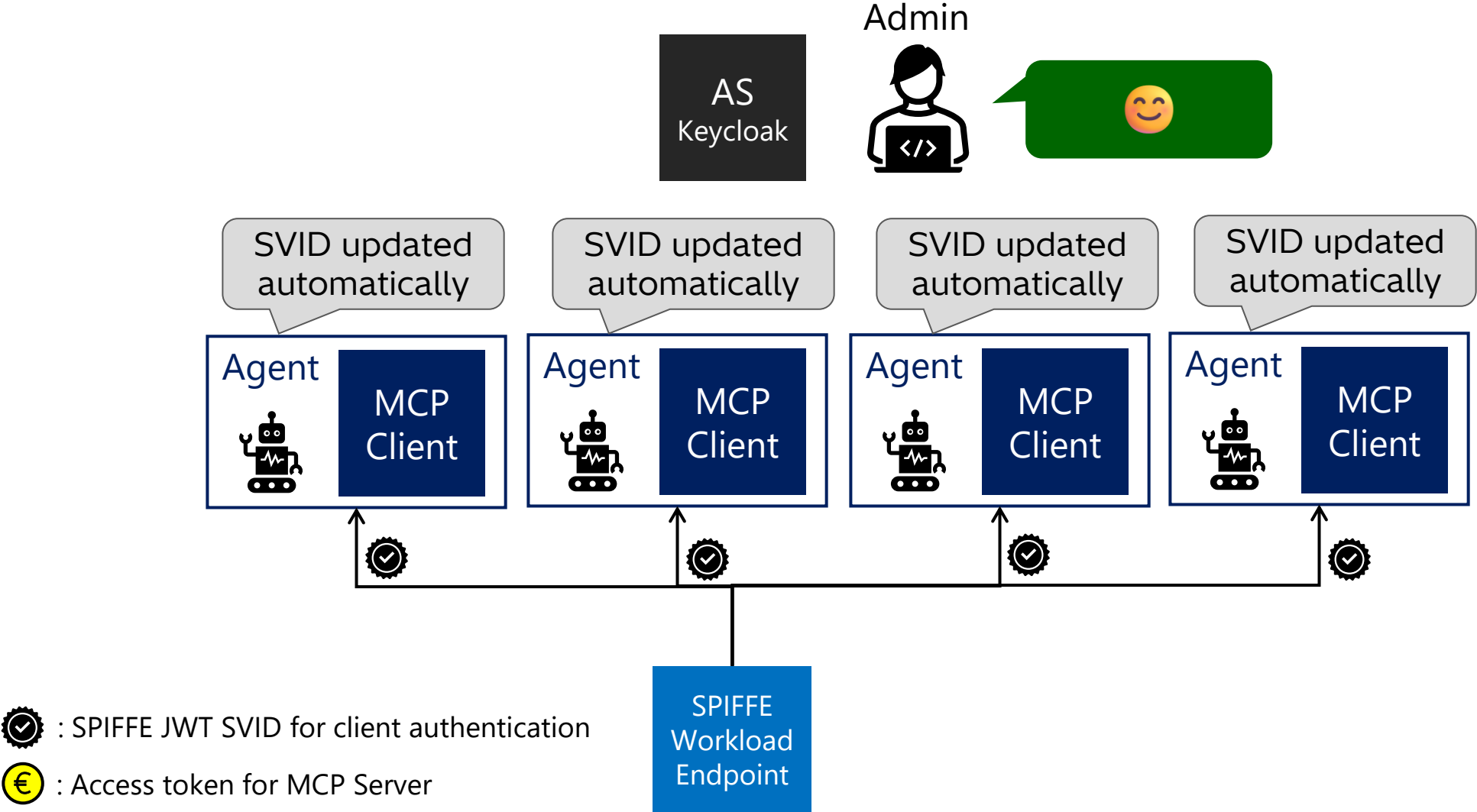
Secret-less Workload Authentication



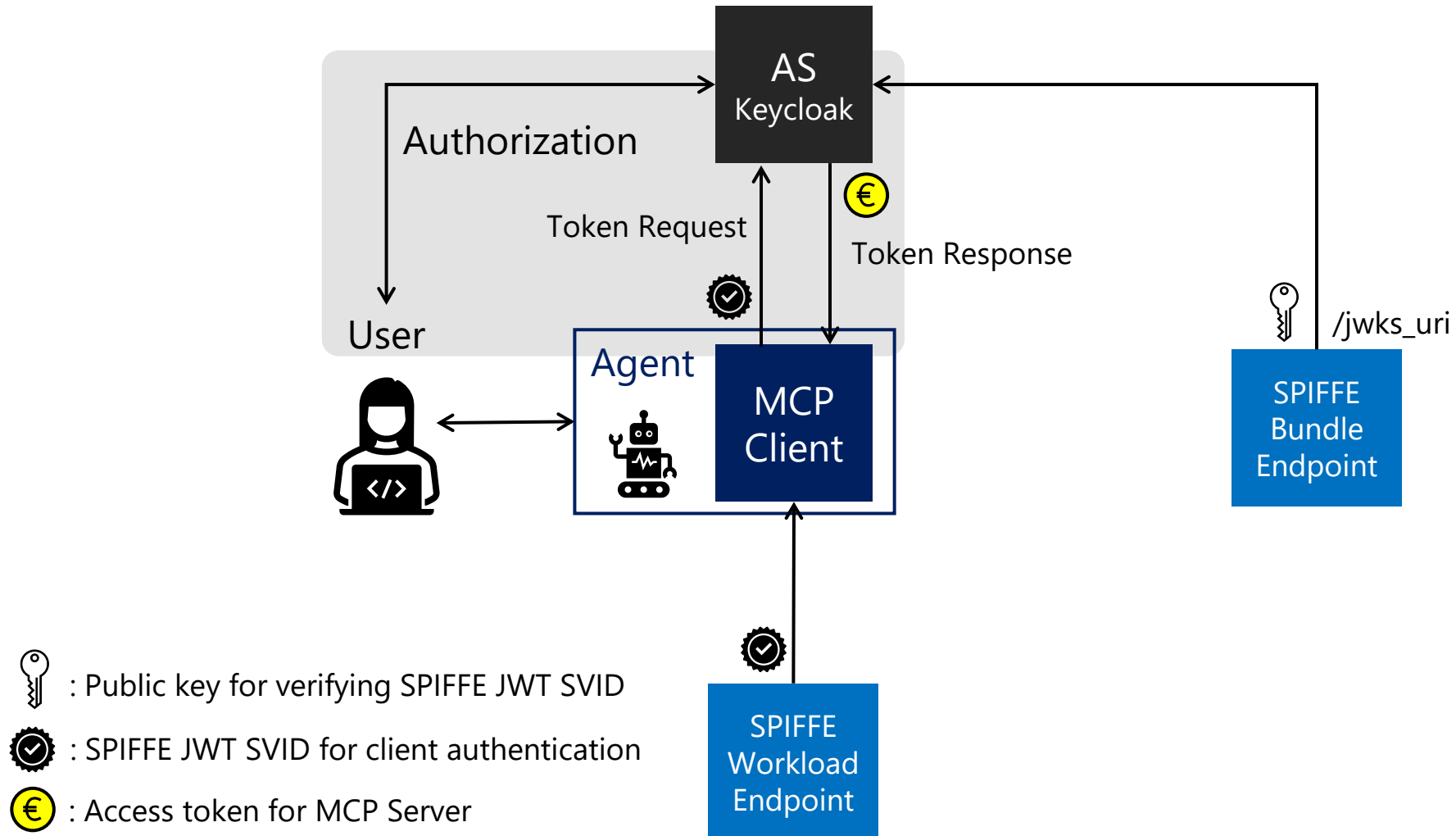
🔑 : Client secret for client authentication

🪙 : Access token for MCP Server

Secret-less Workload Authentication



Secret-less Workload Authentication



Contents

1. What features are beneficial for Agentic AI Authentication and Authorization
2. MCP Authorization
3. Cross-Domain API Call
4. Secret-less Workload Authentication
5. **Demonstration**

Turning it on

1

--features=cimd

Advertises `client_id_metadata_document_supported: true`

2

Client policy profile — what to enforce

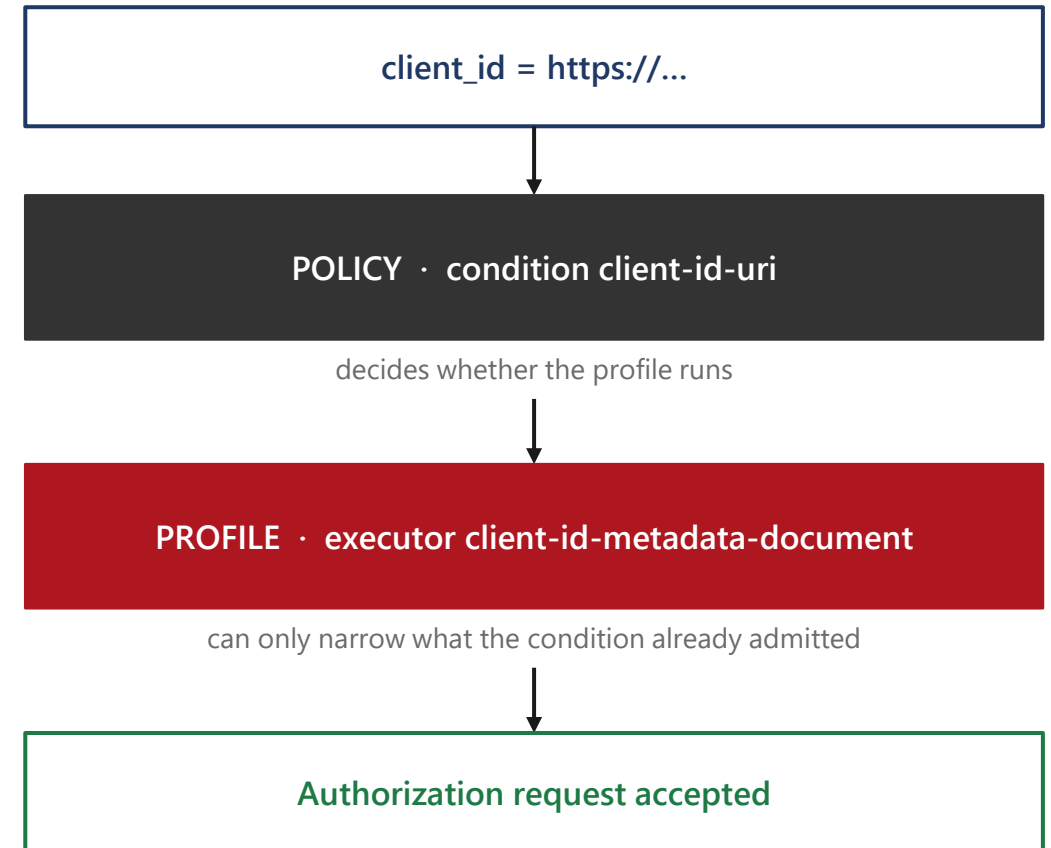
executor: `client-id-metadata-document`. Permitted domains, http scheme, same-domain restriction, required properties.

3

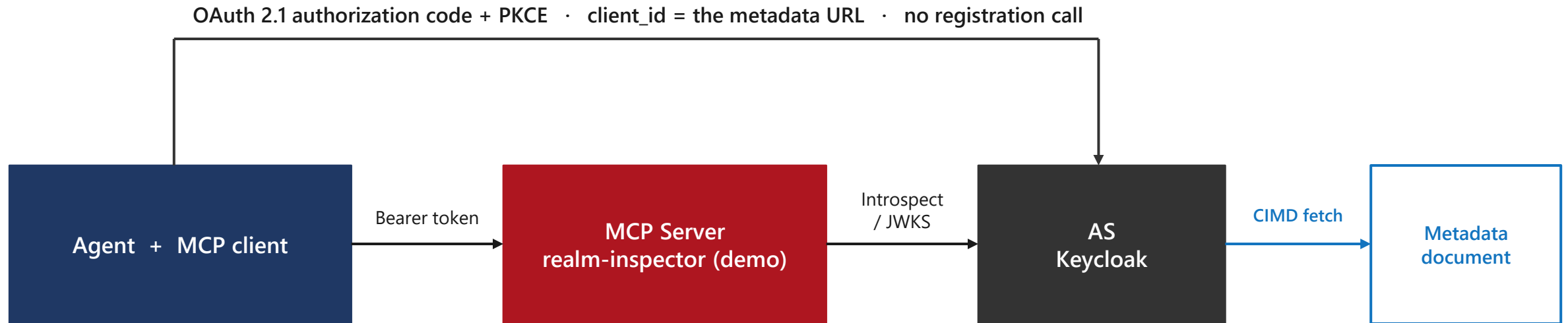
Client policy — when to enforce it

condition: `client-id-uri`. Which schemes and which hosts in `client_id` make the profile run at all.

Neither is useful alone. Profile without policy never runs; policy without profile enforces nothing.



Where CIMD sits in the flow



The MCP client reads `client_id_metadata_document_supported` and picks CIMD over DCR

azp in the access token is the metadata URL

Takeaways

1

The flag is the start, not the switch

CIMD needs an explicit client policy profile and policy.

2

What it removed for us

No registration call, and no need to open anonymous registration. The client's metadata lives at a URL it controls.

3

Where this stands

Experimental behind `--features=cimd` in 26.7.x, and the specification is still an active IETF draft. Worth building against, worth pinning versions.

Demo code: github.com/atulram/keycloak-realm-inspector · admin/admin and relaxed TLS settings in the repo are demo-only

Trademarks

- Keycloak is a trademark of The Linux Foundation in the United States and other countries.
- VS Code is a registered trademark of Microsoft Corporation in the United States and other countries.
- Claude Code is a registered trademark of Anthropic, PBC 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

KeyConf 26
@Holiday Inn Prague, Prague, Czech Republic

Keycloak Becomes Familiar with AI: The Advancement of Integrating Keycloak with AI

Takashi Norimatsu
Open Source Program Office (OSPO)
Hitachi, Ltd.

Atul Ram
Software Development Engineer – Specialist
Hitachi Vantara LLC

Date
October 8, 2026

HITACHI