

Stop Running Agents as Service Accounts

User-Scoped Access for Enterprise Tool Calls

Masato Kozuka
IAM Specialist, Hitachi, Ltd.

Date
October 22, 2026

About me



Masato Kozuka

IAM Specialist, Hitachi, Ltd. in Japan



Current AI Focus

- IAM architecture for enterprise AI platforms

Core Technical Background

- Enterprise IAM/SSO architecture, including Keycloak implementations
- Security consulting on API hardening & Zero Trust access
- Platform Engineering advisory for enterprise adoption

Community & Speaking

- Keycloak contributor & Backstage organization member
- Maintainer of Backstage Auth Plugin (for Keycloak)
- Speaker at Open Source Summit Japan 2025, MuleSoft Meetup Tokyo, etc.
- Upcoming speaker at KubeCon NA 2026 & BackstageCon NA 2026

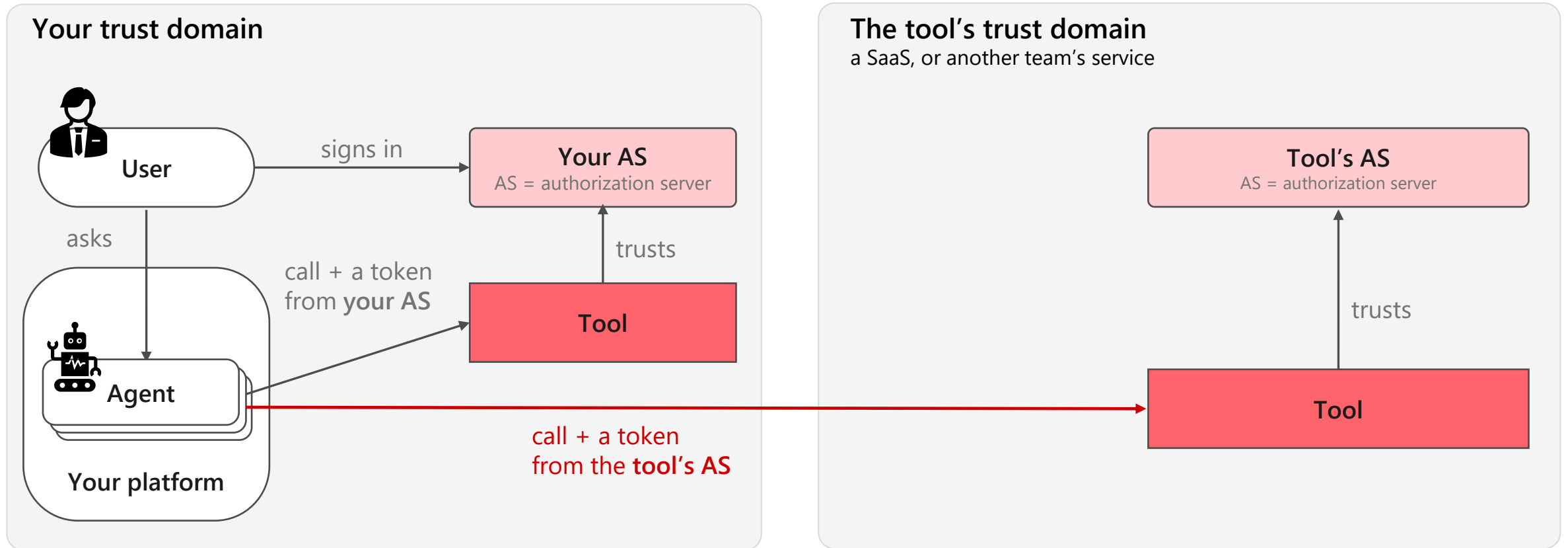
Can your tool know...

which user asked?
and which agent acted?

The tool trusts its own AS, not yours

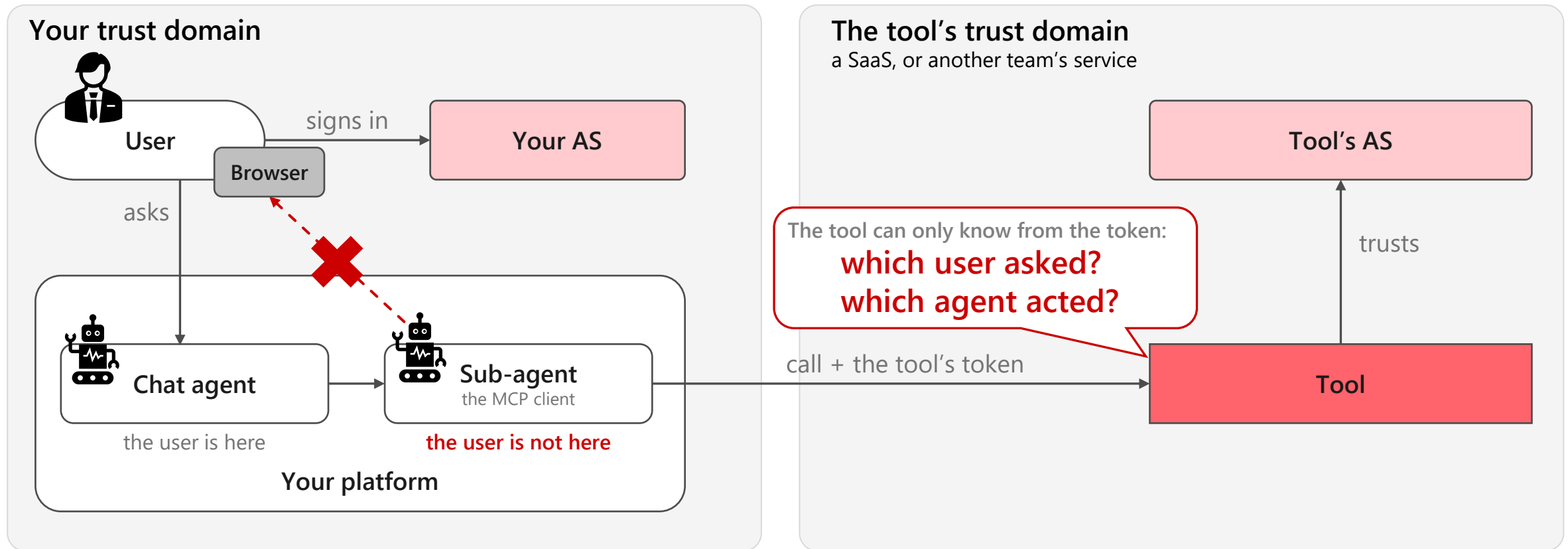
A trust domain is everything one AS protects.

This talk is about agents calling tools across trust domains.



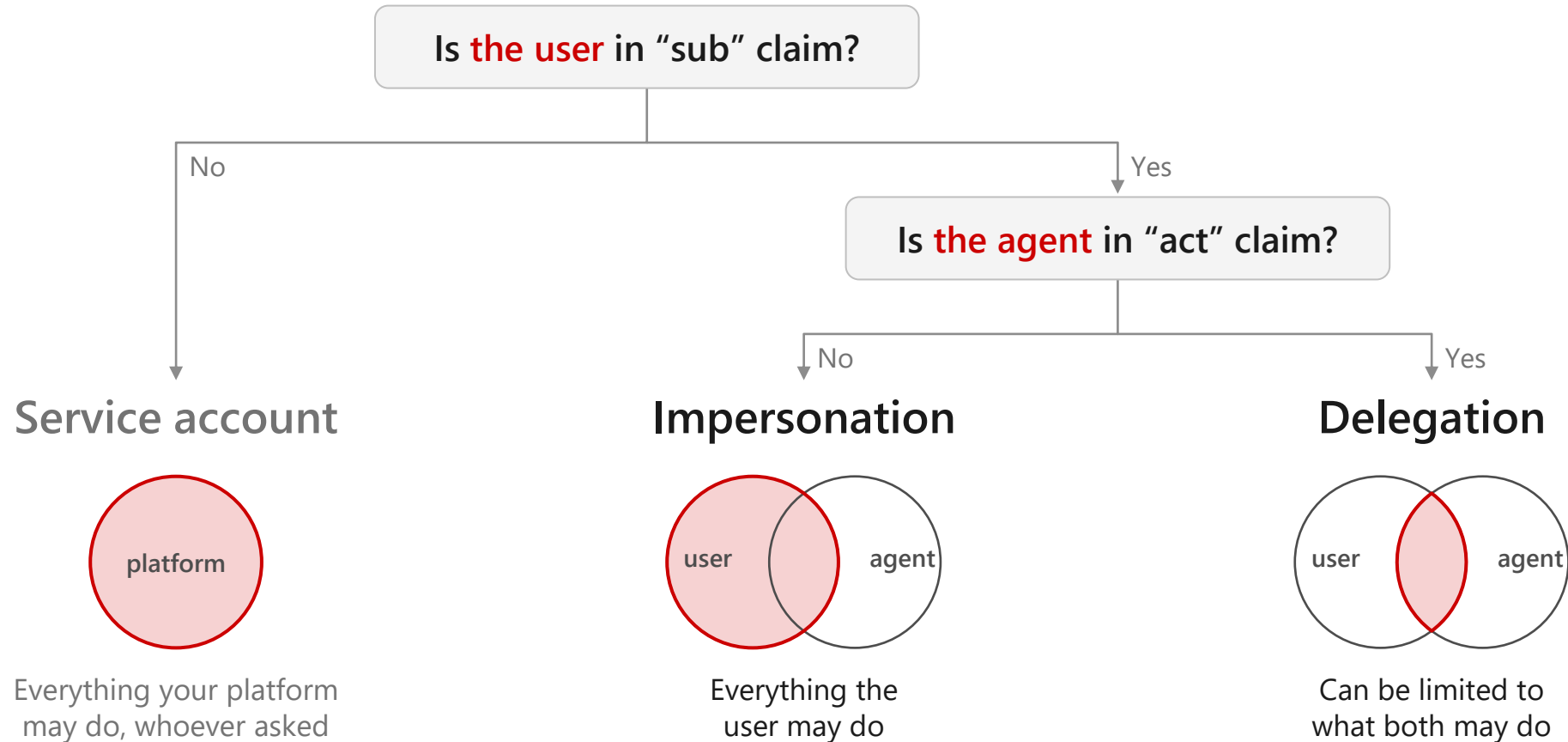
The agent that calls the tool can't reach the user

The call comes from a sub-agent, which can't open the user's browser.
So the tool's token has to come without the user signing in.



Impersonation or delegation: can the tool tell who acted?

Impersonation and delegation, as "RFC 8693 (OAuth 2.0 Token Exchange)" defines them.



One line apart: the same call at Levels 0, 1 and 2

Access tokens the tool's AS issued for that same call.

Level 0

Service account

iss = https://as.tools.example.com

exp = 1792678037

aud = tools.example.com/hr

scope = hr:read

client_id = platform-client

sub = **svc-agent-platform**

act = (none)

-
- ✗ which user
 - ✗ which agent

Level 1 (reachable today)

Impersonation

iss = https://as.tools.example.com

exp = 1792678037

aud = tools.example.com/hr

scope = hr:read

client_id = platform-client

sub = **usr_51c9...**

act = (none)

-
- ✓ which user
 - ✗ which agent

Level 2

Delegation

iss = https://as.tools.example.com

exp = 1792678037

aud = tools.example.com/hr

Scope = hr:read

client_id = platform-client

sub = usr_51c9...

act = **expense-bot**

-
- ✓ which user
 - ✓ which agent

Agenda

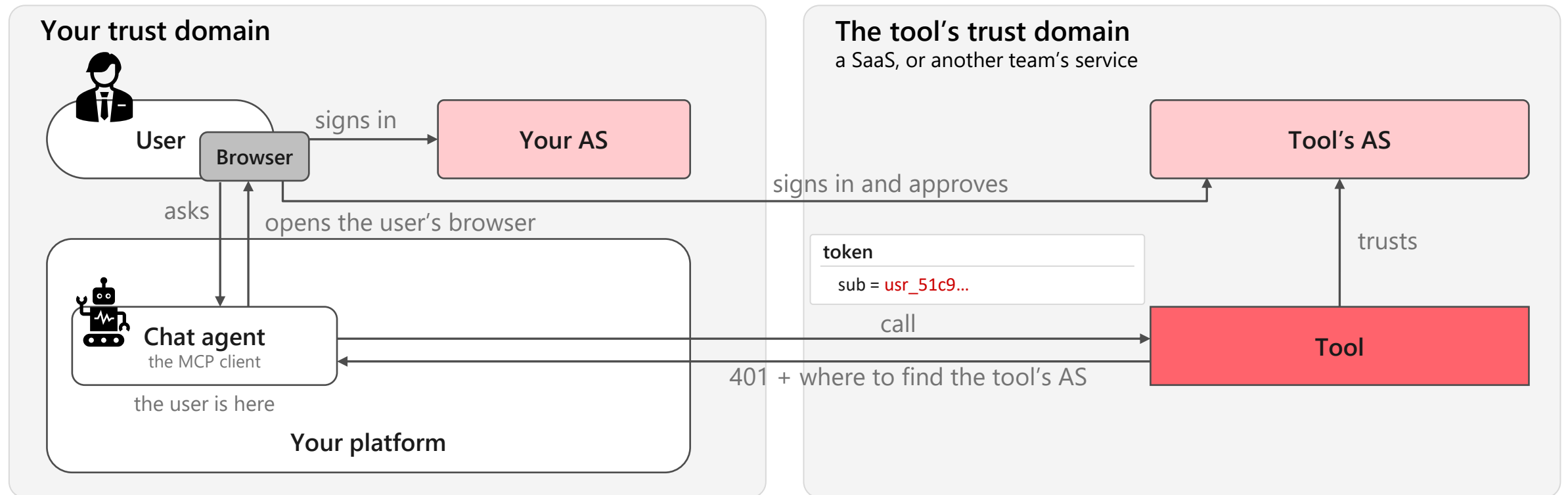
- Level 0 → 1 : How do you get the user into the token?
- Level 1 : How do you check the agent?
- Level 1 → 2 : How do you get the agent into the token?
- Takeaways : What can you do on Monday?

Level 0 → 1:

How do you get the user into the token?

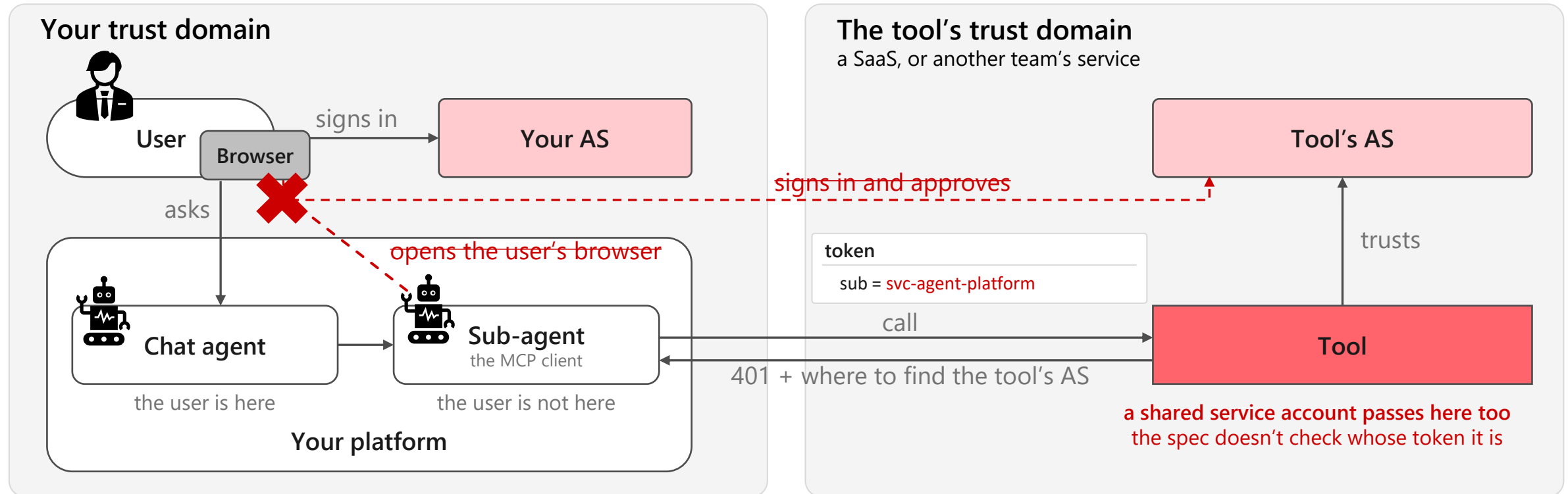
In core MCP, the user gets into the token by signing in

OAuth 2.1 authorization code grant: the MCP client opens the user's browser.



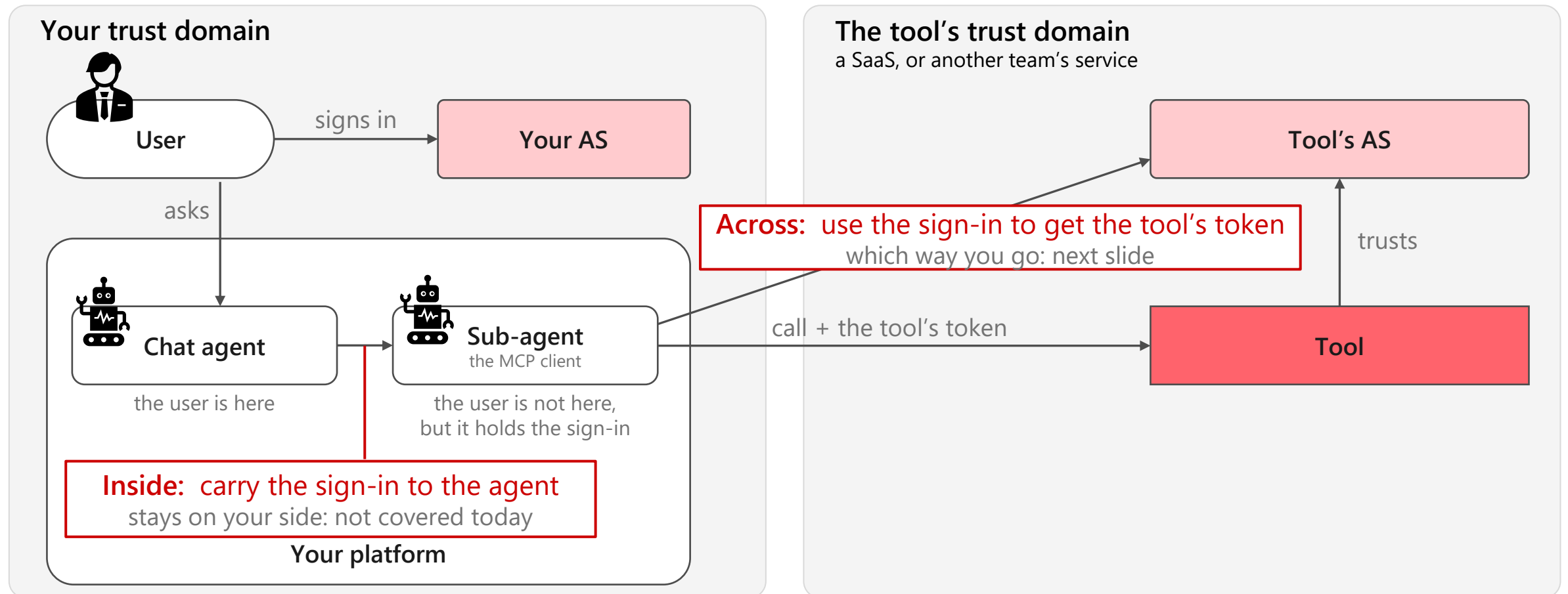
The flow breaks at the browser, but the user already signed in

OAuth 2.1 authorization code grant: the MCP client has to open the user's browser.



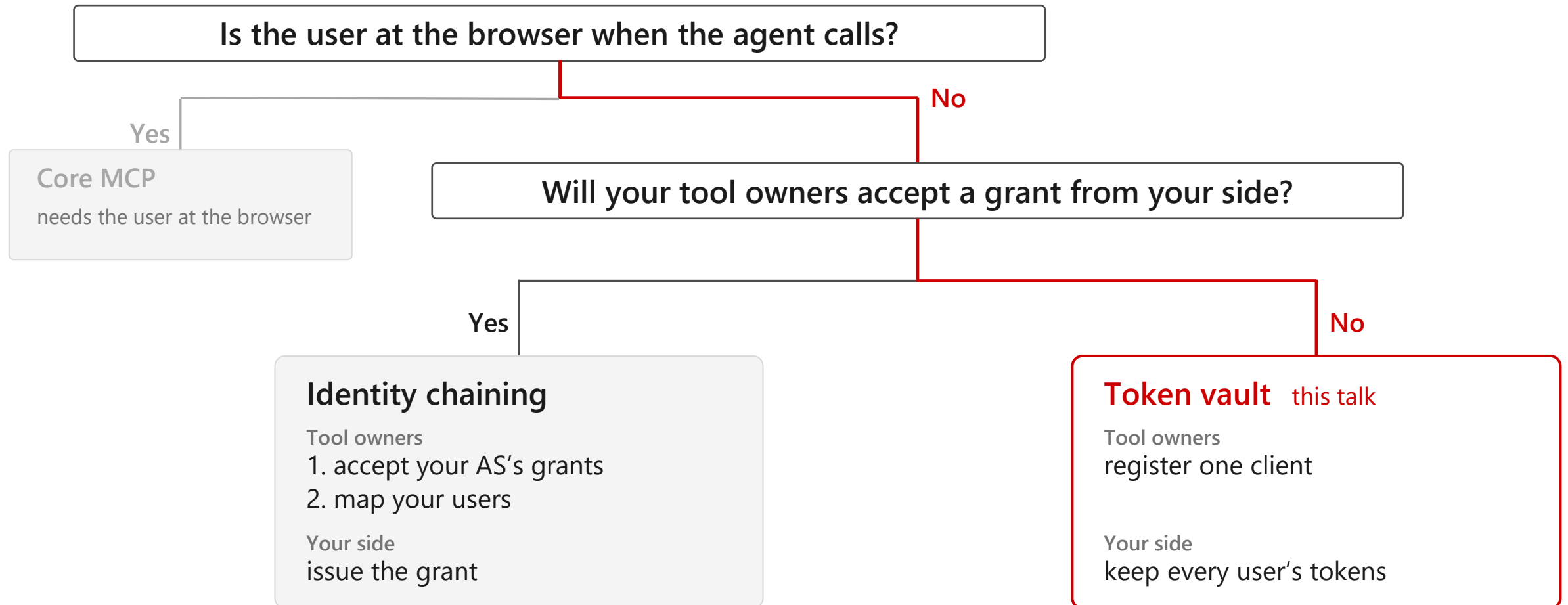
Two parts: get the sign-in to the agent, then get the tool's token

Given: the sub-agent can't open the user's browser.



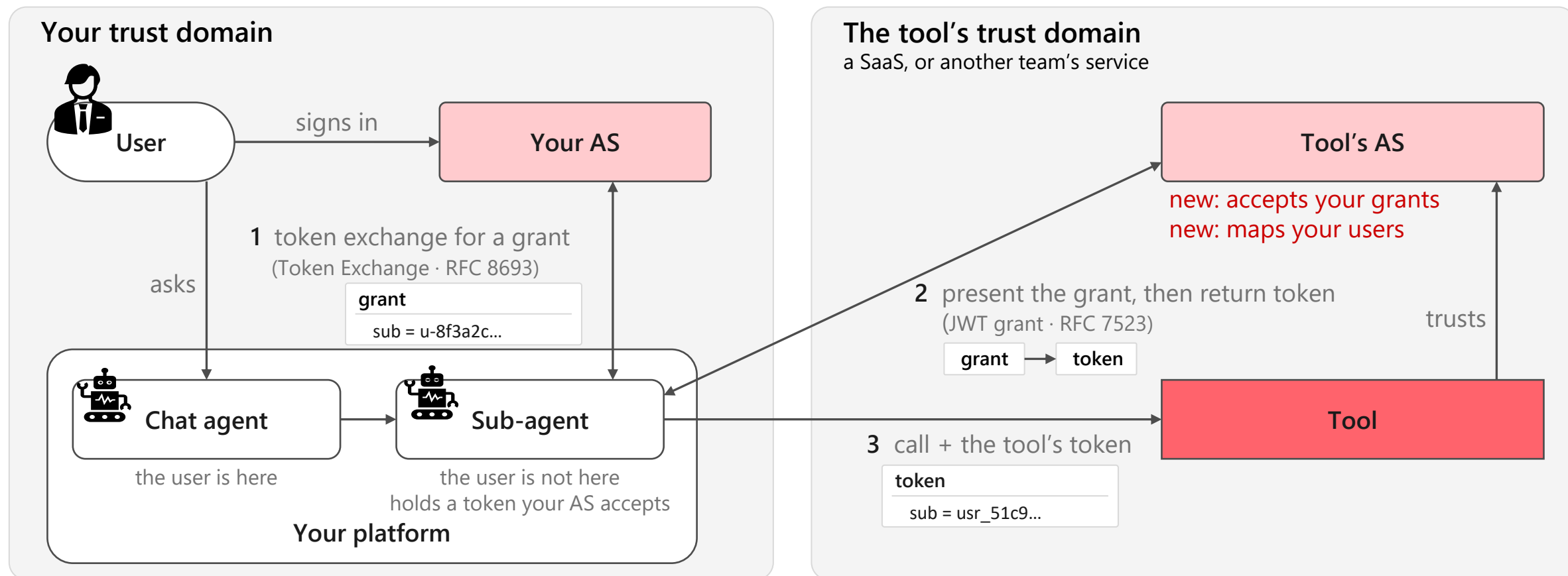
Across: who has to move decides which way you take

Grant: a token your AS issues that the tool's AS accepts instead of a sign-in.



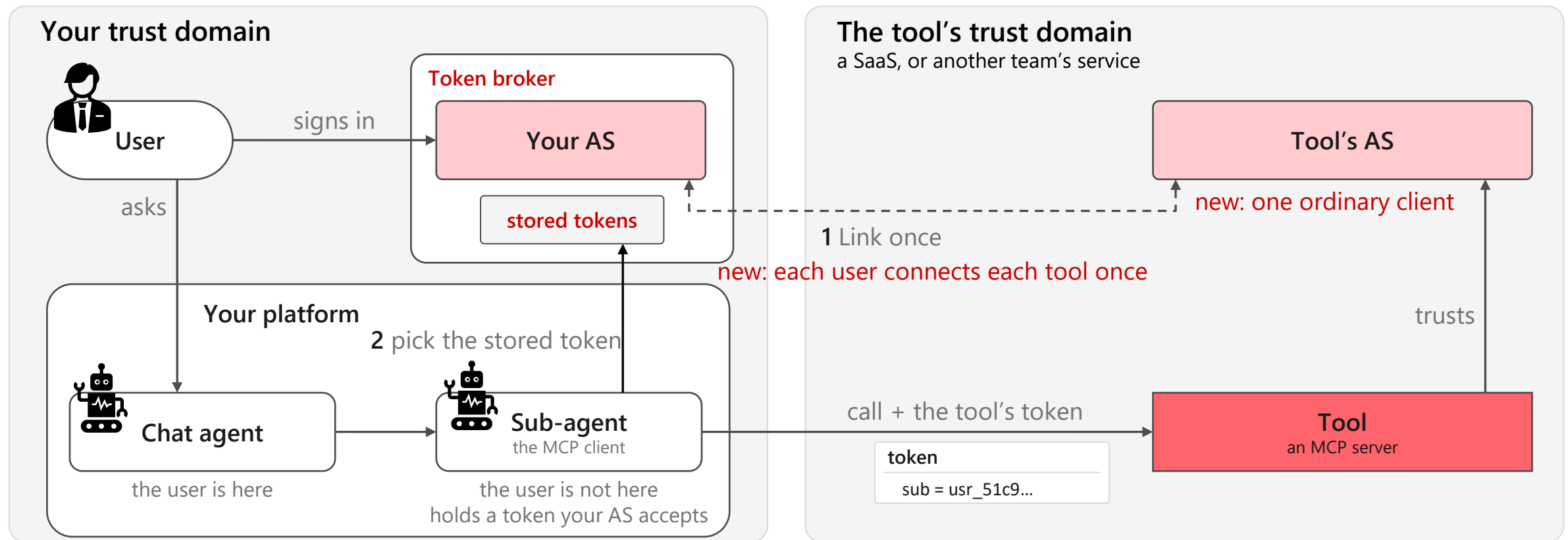
OAuth carries that sign-in, if the tool's AS accepts your grant

Identity chaining: token exchange (RFC 8693), then JWT grant (RFC 7523).

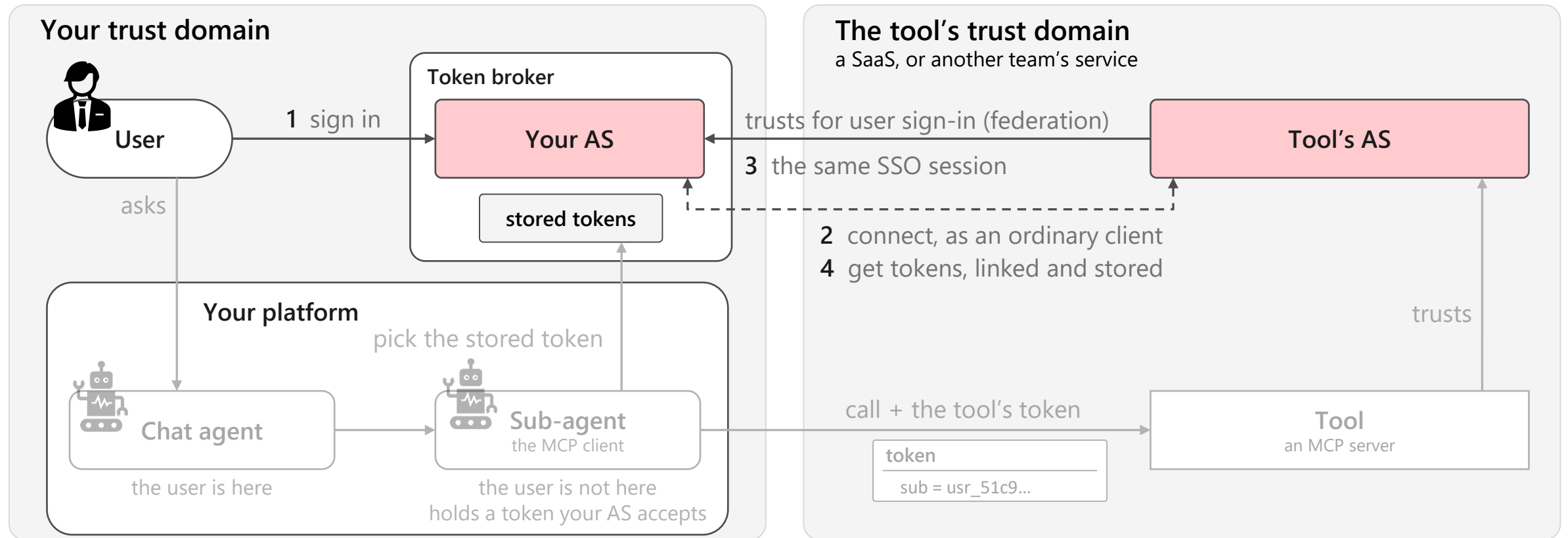


When tool owners only register a client, get tokens ahead of time

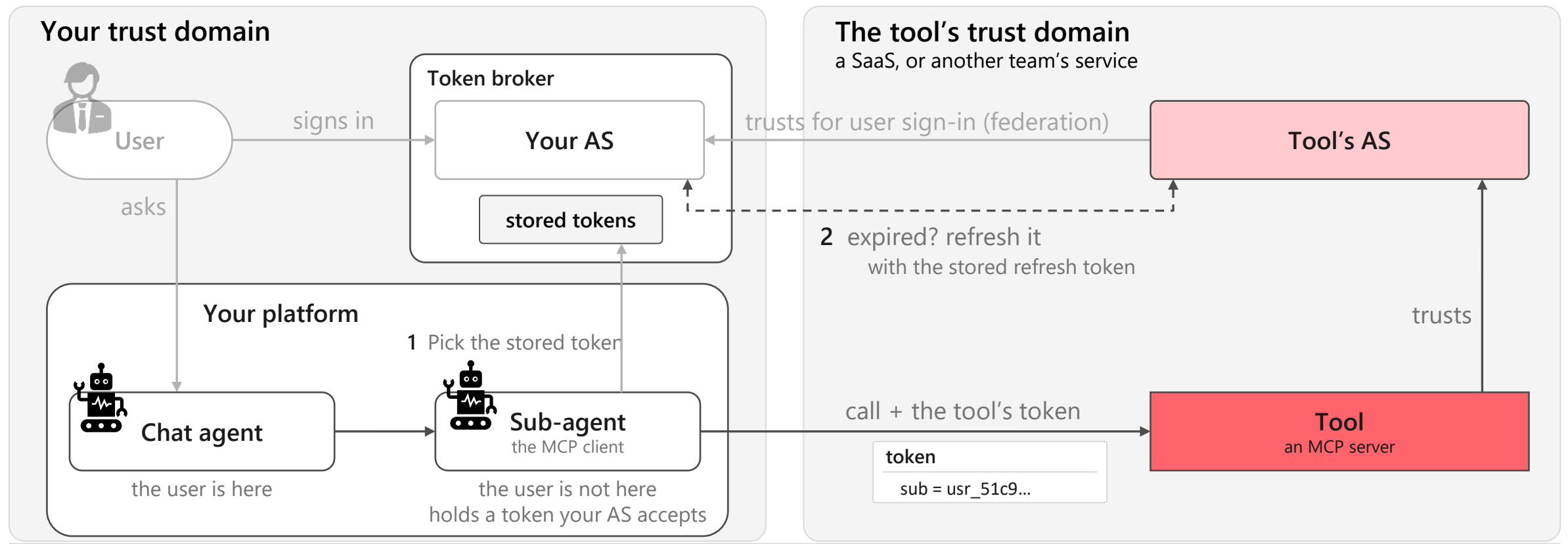
Token vault: your AS keeps the tool's tokens.



Step 1: link once, through the sign-in both sides trust



Step 2: on every call, pick the stored token



With the user in the token, the tool can finally authorize

Tool owners only register a client, but you hold the tokens

This is the price of the vault.

1 Stored tokens

What it costs

A usable credential at rest, per user and tool.

What limits it

A stolen refresh token is useless without the broker's client credentials.

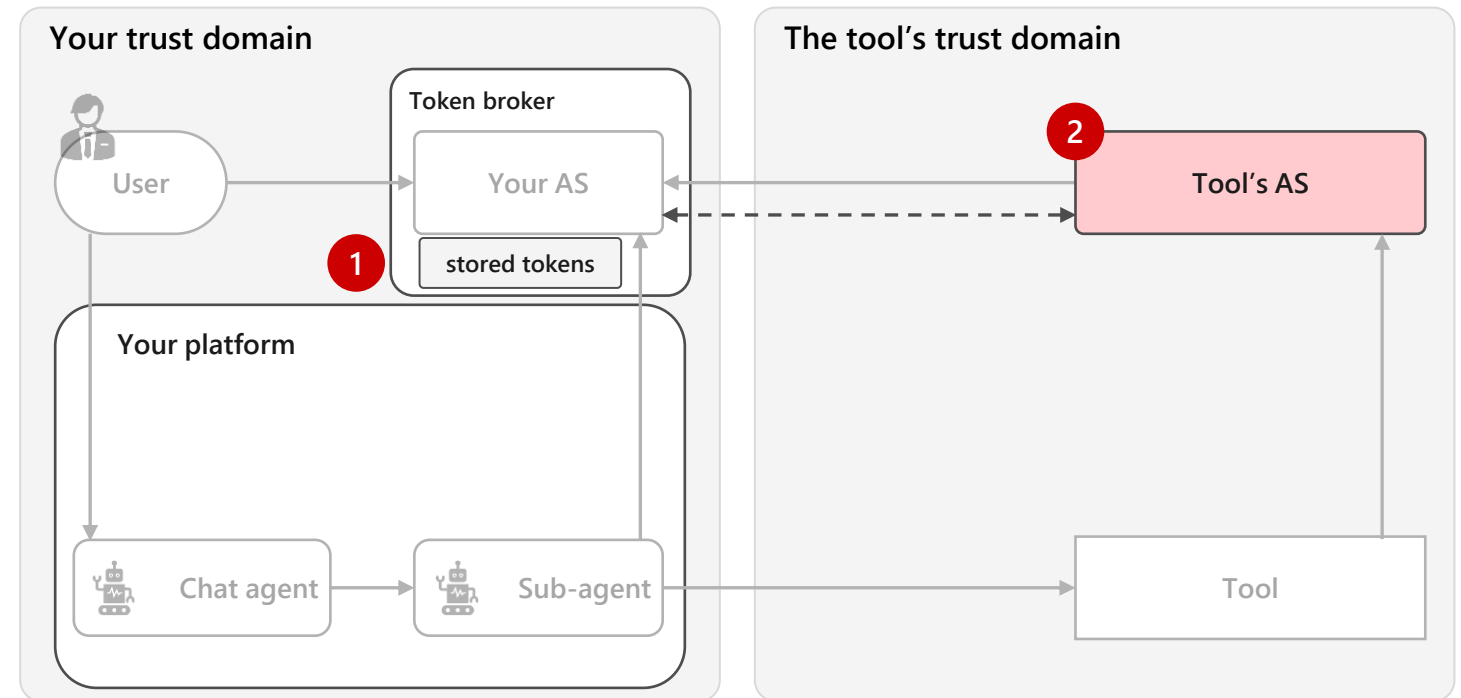
2 Revocation

What it costs

Disabling a user stops sign-in, not the stored tokens.

What limits it

Delete that user's stored tokens when you offboard, and renewal stops.

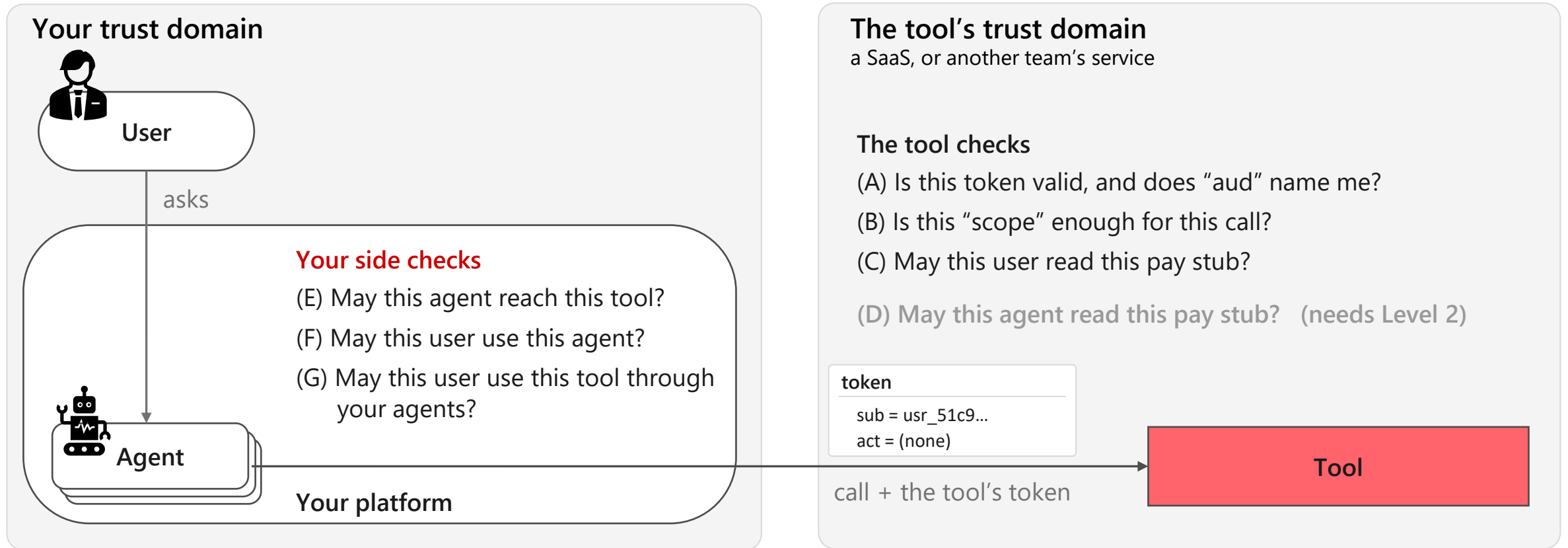


Level 1:

How do you check the agent?

At Level 1, only your side can check the agent

Per-agent checks on the tool's own data wait for Level 2.



Check the agent where every token is picked up

One way to meet all three: a routing layer, the broker and one catalog

The three things it needs

1 Sits in every token's path

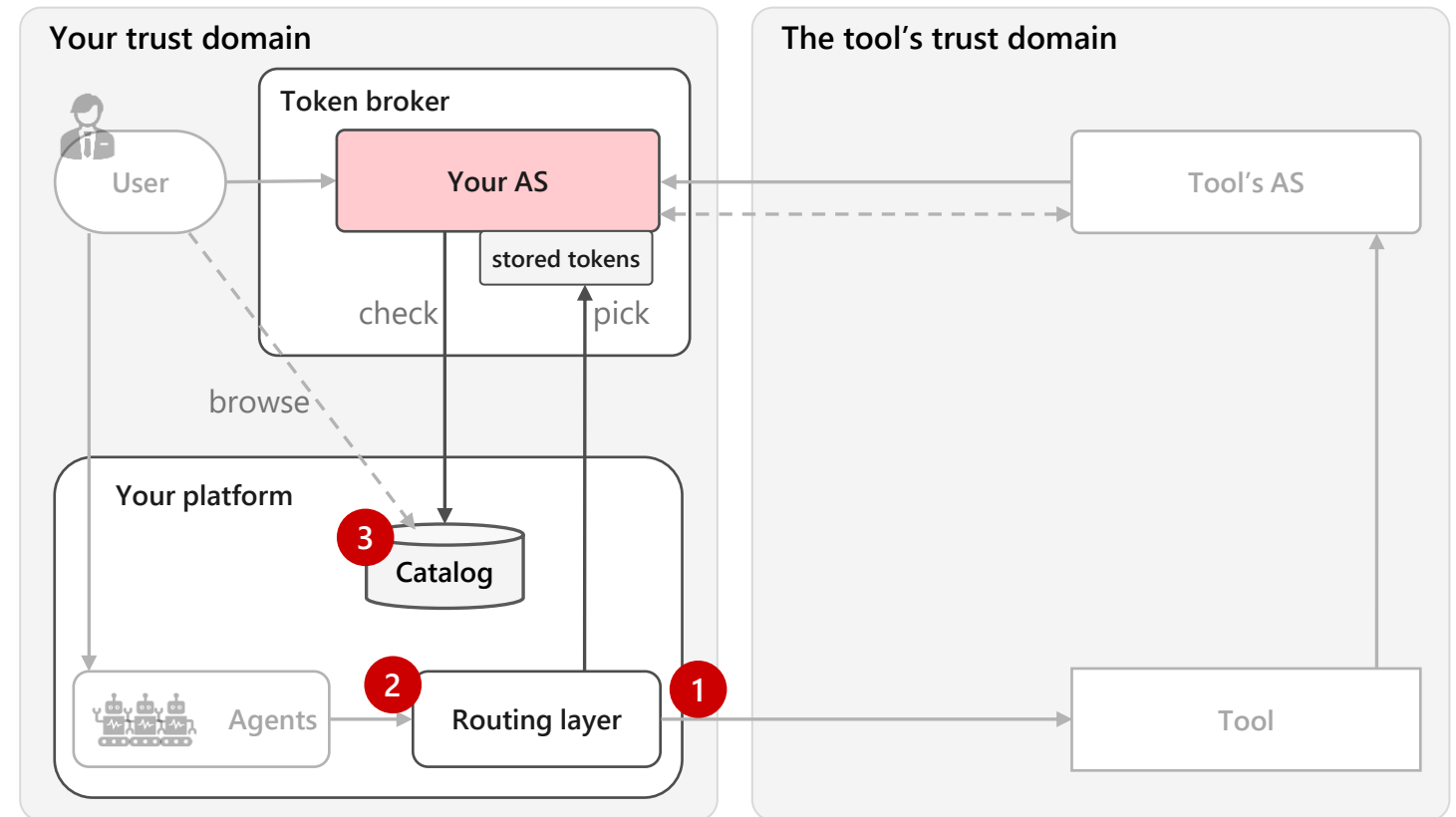
Agents carry only tokens the broker accepts, and every call passes the routing layer.

2 Knows the real agent

The routing layer knows which agent sent each call.

3 Uses one policy

One catalog for users, agents and tools.

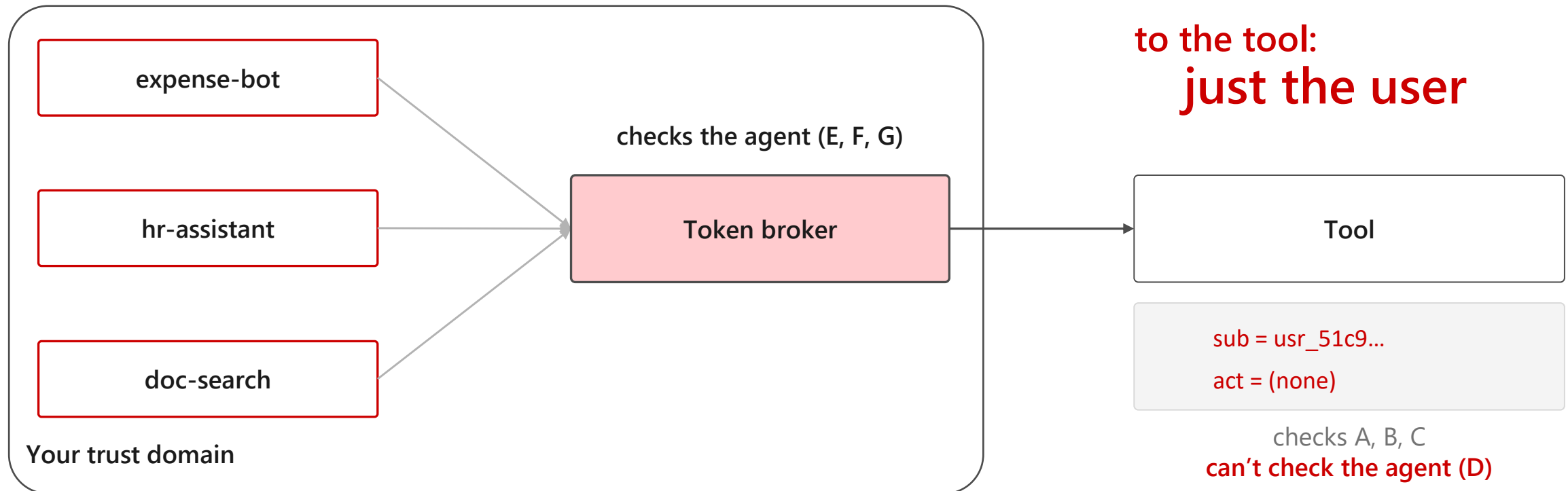


Level 1 → 2:

How do you get the agent into the token?

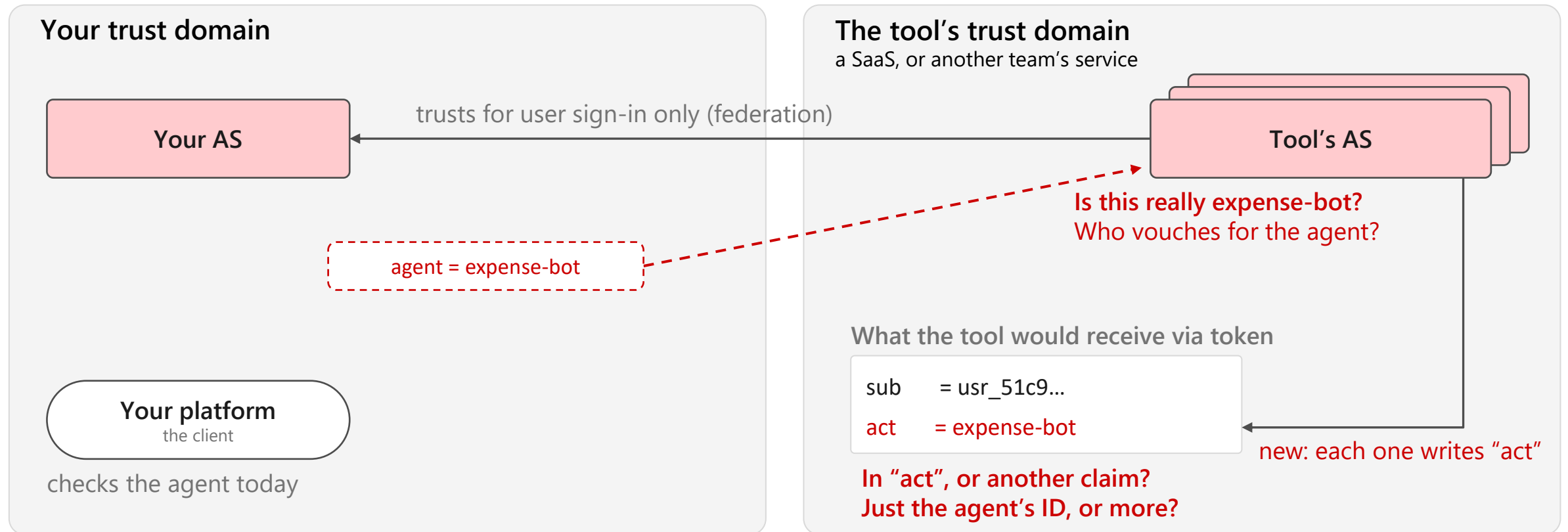
Level 1 works. Level 1 is impersonation.

To the tool, all three agents are the user.



Level 2 asks every tool owner to move again

Only the tool's AS can write "act".



Of Level 2's four questions, only one is settled

The other three are still open. Drafts differ, and most are individual drafts.

Who writes the agent in?

Settled

The tool's AS. The tool trusts nothing else.

Which claim names the agent?

Open

In this talk: "act", RFC 8693's actor claim.

AIMS, a working-group draft, makes the agent the client.
Other drafts add claims of their own.

What else goes in with the ID?

Open

In this talk: only the agent's ID.

Some drafts add the task and its context, in claims of their own.

Who vouches for the agent?

Open

Your AS, through the grant?

Your platform, through an actor token?

No adopted profile for either yet: only individual drafts.

Takeaways:

What can you do on Monday?

Five things for Monday

01

Look at what your tool knows

If the answer is a service account, that is Level 0, and the fix is the user in the token.

02

Use a JWT grant where your tool owners will accept one

Identity chaining, or in MCP, the Enterprise-Managed Authorization extension.

03

Use a token vault where they will not

Keep the tool's tokens on your side. Link each user to each tool once, through the sign-in both sides trust.

04

Check the agent on your side

Put it in every token's path, with one policy and an agent ID it cannot fake. Leave the rest to each tool.

05

Ask your tool owners for "act"

Only their AS can write it. Follow how the three open questions get settled.

Thank you

HITACHI

AGNTCon + MCPCon North America 2026
@San Jose, California

Stop Running Agents as Service Accounts User-Scoped Access for Enterprise Tool Calls

Masato Kozuka
IAM Specialist, Hitachi, Ltd.

Date
October 22, 2026



LinkedIn



GitHub

Trademarks

- Model Context Protocol (MCP) is a trademark of LF Projects, LLC.
- OpenID is a trademark (registered in numerous countries) of the OpenID Foundation.
- Other product names, company names, and logos mentioned in this presentation are trademarks or registered trademarks of their respective owners.