

START HERE

Every agent gets its own identity. Every hop proves who it is for.



The problem today

Most agents run on a person's API key. They get that person's full access with no task scope, and the audit log cannot tell the agent from the person.



1 Give every agent its own identity

A first-class principal in the platform, with an owner, scopes and a kill switch. Native to DataRobot, or federated to your IdP (Okta, Entra) if you choose.



2 Trade the token at every hop

Nothing is forwarded. The authorization server swaps each token for a new one bound to the next hop only, valid for minutes (RFC 8693 token exchange).



3 Carry the subject and the actor

sub is who the request is for: alice. act is who is acting: agent-2, then agent-1. The API at the end sees both, so no agent can become a confused deputy.



4 Let policy decide each hop

effective = caller n agent n task n resource

RBAC for who may call which agent or MCP. ABAC for tenant, data class and time. **Fine-grained** scopes per tool. **Sensitive** actions wait for a person to approve.

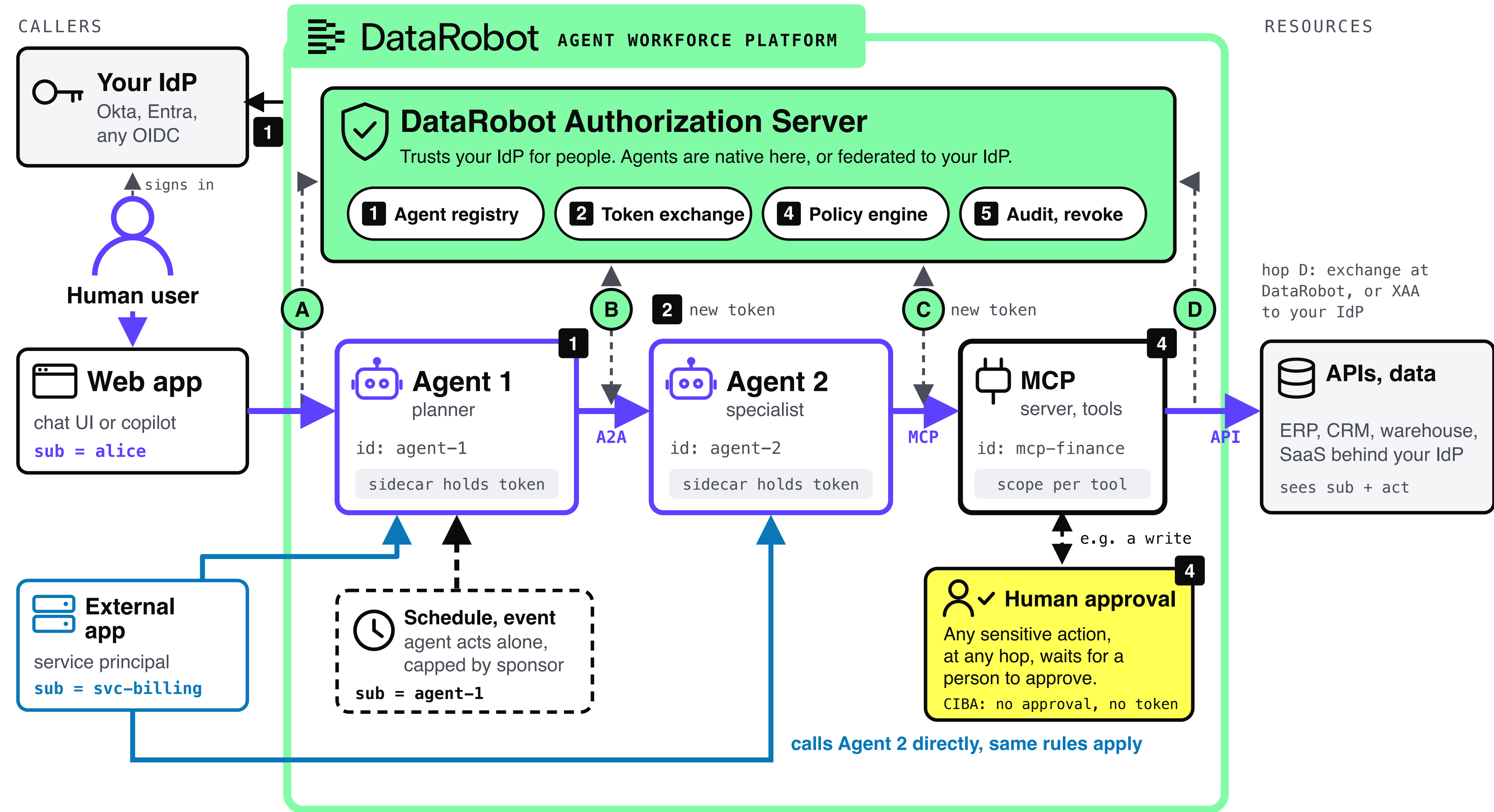


5 Revoke one agent, audit the chain

Suspend an agent and new tokens stop at once. Tokens it holds expire in minutes. Every call is logged with its full chain, and no person's key is rotated.

REFERENCE ARCHITECTURE

One request, three kinds of caller. **Every hop gets its own token.**



3 Follow alice's request through hops A to D

sub never changes. act grows by one. aud names one resource. Scope only narrows.

A App → Agent 1

sub **alice**
act **web-app**
aud **agent-1**
scope **workloads.invoke**

B Agent 1 → Agent 2

sub **alice**
act **agent-1** ← web-app
aud **agent-2**
scope **workloads.invoke**

C Agent 2 → MCP

sub **alice**
act **agent-2** ← agent-1
aud **mcp-finance**
scope **expenses.read**

D MCP → API

sub **alice**
act **mcp** ← agent-2 ← agent-1
aud **erp-api**
scope **expenses.read**

