Agent Gateway: Secure Interoperability and Delegated Authority with Third-Party Identity Providers

Author: Sumeet Mittal, Google Cloud

The Agentic Transformation and the Identity Frontier

The **Gemini Enterprise Agent Platform** provides a comprehensive, vertically integrated stack to build, scale, govern, and optimize autonomous AI agents at scale. The enterprise landscape is undergoing a fundamental "agentic transformation," shifting from static, siloed applications to autonomous AI ecosystems capable of complex reasoning and execution. Market data confirms this shift: Gartner predicts 40% of enterprise apps will feature task-specific AI agents by 2026, up from less than 5% in 2025, and ~52% of organizations have already deployed agents in production. The Gemini Enterprise Agent Platform establishes the necessary infrastructure to mediate the expanding identity and trust boundary.

A cornerstone of this agent platform is the Google Agent Gateway (AGW), which functions as a unified Policy Enforcement Point. Agent Gateway allows you to enforce governance on all agentic traffic while freeing developers to focus on business logic in their agents. This architecture eliminates the risk of fragmented, embedded security logic and ensures that every AI-driven action is governed by centralized enterprise intent.

Foundations of Agentic Identity & Identity Sovereignty

The core security narrative of modern enterprise architectures centers on a vital principle: **agents should never have raw, unscoped, direct access to tokens/credentials**. Without a governing layer like the Agent Gateway, agents receiving raw access tokens pose a significant security risk, as any compromise of the agent could lead to full unauthorized access to the

user's resources. By acting as the Policy Enforcement Point (PEP), the gateway ensures agents operate on delegated, scoped assertions rather than raw user tokens, effectively mitigating the risk of credential leakage. In addition, to ensure maximum security, all delegated tokens—particularly those carrying sensitive authorization scopes—should remain encrypted at rest and in transit. This aligns with the principle of least privilege.

Agents represent a distinct identity class requiring specialized access controls tailored to autonomous execution rather than standard human or machine credentials. Hence, Agent Gateway plays a critical role in preventing raw token exposure.

Google Cloud's Agent Gateway adopts an **identity-first philosophy** that completely decouples routing and credential management from the agent's application logic to enforce governance and access policies across all agent interactions. This ensures that even when external user identities are brought into this ecosystem, agents can securely act on behalf of humans without exposing sensitive credentials to the agent's code.

Identity Core Concepts

Before we go further, let's review a few key concepts that provide the structural scaffolding for the high-level architectures discussed in this document.

Concept	Description
Identity vs. Principal	**Identity** is the persistent "who" (e.g., email or service name). A **Principal** is an identity that is authenticated and recognized by Identity and Access Management (IAM) to make requests.
Agent Identity	A standardized, ephemeral principal uniquely identified across the external identity provider and the Google Cloud trust boundary.

Identity Federation	A trust relationship where Google Cloud accepts assertions (Security Assertion Markup Language (SAML)/OpenID Connect (OIDC) tokens) from a **3P IdP** and maps them to local principals.
Identity Delegation	A model where an external provider serves as the authoritative Policy Decision Point (PDP) while Google Cloud's Agent Gateway acts as the Policy Enforcement Point (PEP). Rather than just mapping identities, authority is explicitly delegated to an agent to act on an identity's behalf (utilizing flexible mechanisms like OAuth 2.0 tokens with On-Behalf-Of (OBO) flows, Transaction Token (Txn-Token), or Identity Assertion JSON Web Token (JWT) authorization grant (ID-JAG)).
OAuth/OIDC Flows	Interactive flows optimized for Workforce Federation (human users) using browser-based or app-based sessions.
mTLS/SPIFFE Flows	Non-interactive flows optimized for Workload Federation (autonomous agents) for secure peer-to-peer communication.
SPIFFE	SPIFFE (Secure Production Identity Framework for Everyone) acts as a "universal passport" for workloads. It allows an agent to be uniquely identified regardless of whether it is running on-premises or in a multi-cloud environment.
A2A	Agent-to-Agent communication, typically involving secure peer-to-peer interactions between autonomous agents.
Trust Boundary	A logical boundary outside of which the system does not trust an identity unless it is verified.

Identity Types

During runtime operations, three distinct types of identities interact seamlessly within the trust boundary:

Identity Type	Purpose	Issuing System
ID-User User Identity	Identity used by the user to access agent or Software as a Service (SaaS) (e.g. Gemini Enterprise)	Human IdP, e.g., Okta, Ping Identity, Auth0, Microsoft Entra when user signs up

ID-AgentWorkload Agent Identity (Using its own authority)	Identity used by the agent to access resources under its own authority	GCP, when agent is created
ID-OBO Agent Identity (Using delegated authority from end-user)	Identity used by the agent to access resources using the user's authority	OAuth server (1P or 3P), when user delegates access using OAuth dance

Framework for Identity Extensibility

To achieve enterprise readiness, a security architecture must respect identity sovereignty, meeting organizations where their "**source of truth**" currently resides. While Google Cloud provides a native, turnkey identity and security stack (inc. Identity and Access Management (IAM), Identity Aware Proxy (IAP), Agent Identity, Auth Manager, and so on), global enterprises might want to continue using their existing authorities (such as**,** Okta, Ping Identity, Microsoft Entra etc**.**) and Identity and Governance Administration platforms (such as Saviynt, Sailpoint and so on) for workforce and workload identity access management across their hybrid and multi-cloud environments.

Google's perspective on identity extensibility is highly collaborative, framing external (non-Google Cloud) identity solutions—generally referred to in this paper as Third-Party Identity Providers (3P IdPs)—as complementary components of an open, secure agentic ecosystem. This vendor-neutral approach ensures that the technical architectures described remain applicable irrespective of the specific identity provider chosen to anchor the trust domain. To learn more about the Agent Gateway Independent Software Vendor (ISV) ecosystem, read here.

To support the agentic enterprise in today's multi-cloud and multi-AI world, Google is partnering with leading identity and AI security providers to integrate with Agent Gateway and help ensure that the customer's security posture remains as flexible as the agents you're building.

This paper focuses specifically on the **"Bring Your Own Identity Provider" (BYOIdP)** model. We will examine how 3P IdPs **can** integrate with the Agent Gateway to support mutual customers through Authentication and Authorization (AuthN/AuthZ) delegation. In this context, we distinguish between identity federation and delegation: while federation typically involves mapping external identities to Google Cloud principals, this document focuses on a delegation model where the 3P IdP serves as the Policy Decision Point (PDP) and the Agent Gateway functions as the Policy Enforcement Point (PEP).

Note: *Some capabilities described in this paper may be under development at the time of publication. For the most current information on available features, please reach out to your Google Cloud representative.*

Core Architecture: The Envoy Proxy

The centerpiece of this architecture is the Agent Gateway, which utilizes a high-performance architecture built on the **Envoy proxy**. Agent Gateway intercepts all incoming traffic—including Model Context Protocol (MCP), Agent-to-Agent (A2A), Representational State Transfer (REST), and Remote Procedure Call (gRPC)—ensuring that no request reaches an agent without cryptographic validation.

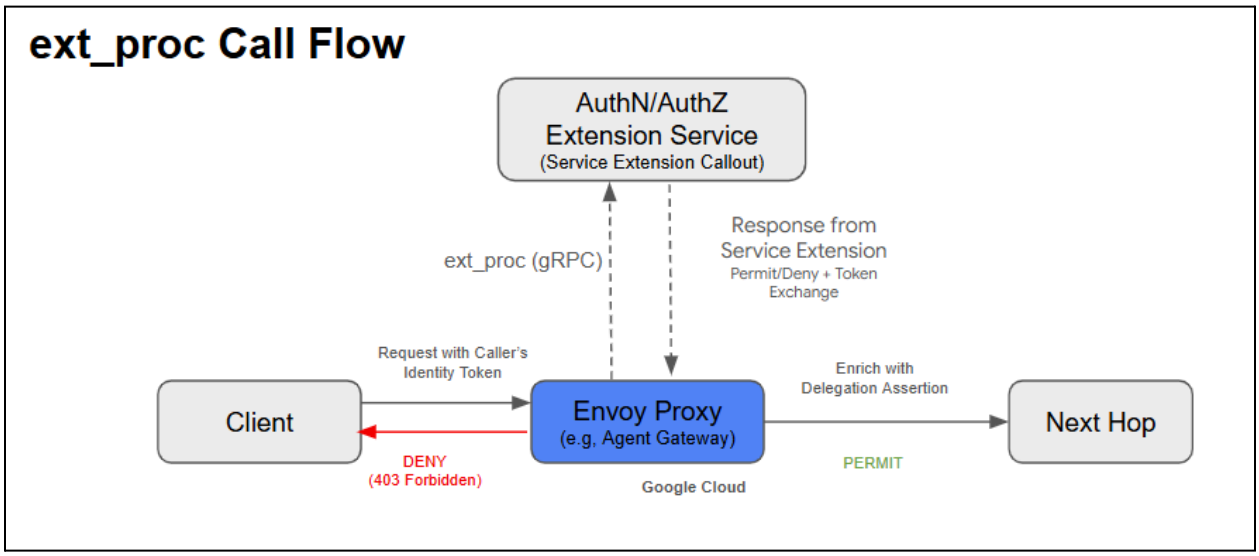
The BYOIdP model is enabled by Google Cloud Service Extensions, which leverage the industry-standard ext_proc gRPC callout mechanism to integrate 3P IdPs directly into the request/response path, serving as the external Policy Decision Point (PDP).

This architecture also allows flexibility for partners to decouple the Policy Decision Point (PDP) for runtime policy enforcement from the Token Issuance and Exchange mechanisms. The extension service can decompose the gRPC stream into multiple distinct operations (Authorization, then Token Issuance), effectively decoupling the two phases. Partners can choose to implement these as separate service endpoints—one for runtime fine-grained tool

authorization and another for OAuth token issuance/exchange—ensuring that token lifespans and scopes are managed appropriately at each boundary.

Through this delegation, the Agent Gateway can offload complex authorization logic to the enterprise's chosen identity authority while maintaining its role as the authoritative Policy Enforcement Point (PEP). Note that Agent Gateway can support both Ingress and Egress deployment modes.

Here is a rough data flow using GCP service extensions and BYOIdP.



- ***Request Interception:*** *The Envoy Proxy (e.g., Agent gateway) captures the incoming request (MCP/REST/gRPC).*
- **`ext_proc`** ***Callout:*** *A streaming gRPC call is initiated to the AuthN/AuthZ extension service. It carries the identity context for the caller client.*
- ***AuthN/AuthZ Decisioning:*** *The extension service validates the caller's identity and evaluates access rules against a centralized policy store (such as* ***Open Policy Agent****), decoupling authorization logic from runtime application code, achieving protocol mediation at scale.*
- ***Token Exchange & Delegation Headers:*** *If authorized, the extension service mints and injects an appropriate delegation assertion into the request header—such as On-Behalf-Of (OBO) standards implemented via RFC 8693, or draft specifications such as Identity Assertion JWT authorization grant (ID-JAG) or Transaction Tokens (txn-token). This short-lived assertion*

(Time To Live 5–10 minutes) contains critical claims to maintain a chain of trust, or the gateway can enforce pure PDP-based permissions without requiring token issuance.

- ***Permit/Deny Enforcement:*** *The service returns a ProcessingResponse to the Gateway.*
    - *For "**Permit**" decisions, the gateway forwards the request enriched with the appropriate delegation token (e.g.,OAuth 2.0 tokens with OBO flows, Txn-Tokens, or ID-JAG, if applicable) to the next hop target agent/tool.*
    - *For "**Deny**" decisions, the gateway issues an **ImmediateResponse**, returning an **HTTP 401/403** directly at the edge. This prevents unauthorized traffic from ever reaching backend agents, effectively reducing the computational and security blast radius.*

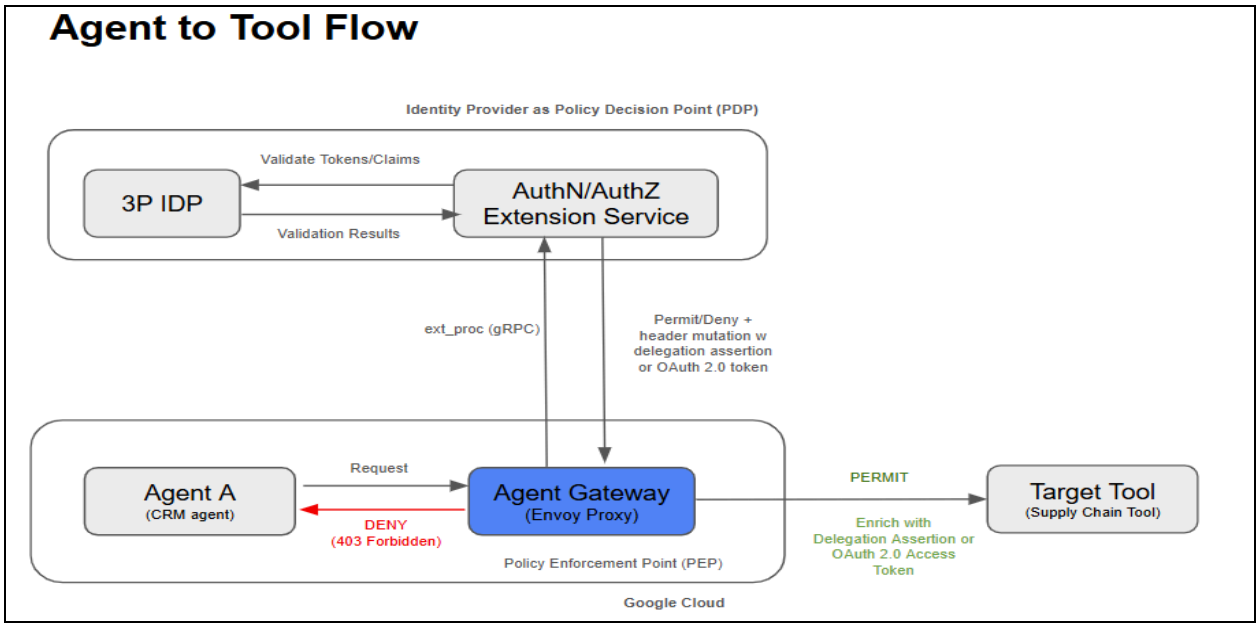
Cumulative Customer Journeys

Now that we have established the mechanism of how the Agent Gateway Integration works as a policy enforcement point with 3rd party IdP providers, let's explore the most **common customer journeys** for agentic workloads. Each journey builds chronologically and cumulatively upon the technical mechanics established in the previous scenario.

Customer Journey 1: Baseline Autonomous Agent-to-Tool

Consider a Google Cloud-hosted CRM agent (Agent A) that identifies a critical depletion in regional inventory and must securely update an external, third-party supply chain tool.

Technical Resolution:



1. ***Agent Authentication (mTLS):*** *The connection is inherently secured via* ***mutual TLS (mTLS)*** *as both the CRM agent and the Agent Gateway reside within Google Cloud. The Gateway terminates the mTLS session, validates the client certificate, and extracts the agent's* ***SPIFFE ID*** *to establish a verified workload identity.*
2. ***AuthN/AuthZ Callout and Authorization:*** *The Gateway triggers the* `ext_proc` *call to the AuthN/AuthZ Extension Service. The Extension Service authenticates the Google Agent's SPIFFE ID and authorizes the request based on predefined policies (Role Based Access Control (RBAC)/ Attribute Based Access Control (ABAC)) for accessing the specific target tool.*
3. ***Token Exchange & Tool Access:*** *To cross the trust boundary to access the external supply chain tool, the Extension Service performs a Token Exchange (RFC 8693 or standard OBO exchange) with the enterprise's 3P IdP, trading the verified agent context with an external tool-specific credential, ensuring least-privilege access.*
4. ***Secure Egress:*** *The Agent Gateway enables secure egress communication, allowing the CRM agent to execute an inventory update with full authorization while maintaining a strictly audited chain of trust and generating an unalterable audit trail recording the Trace ID, x-agent-id, and certificate fingerprint.*

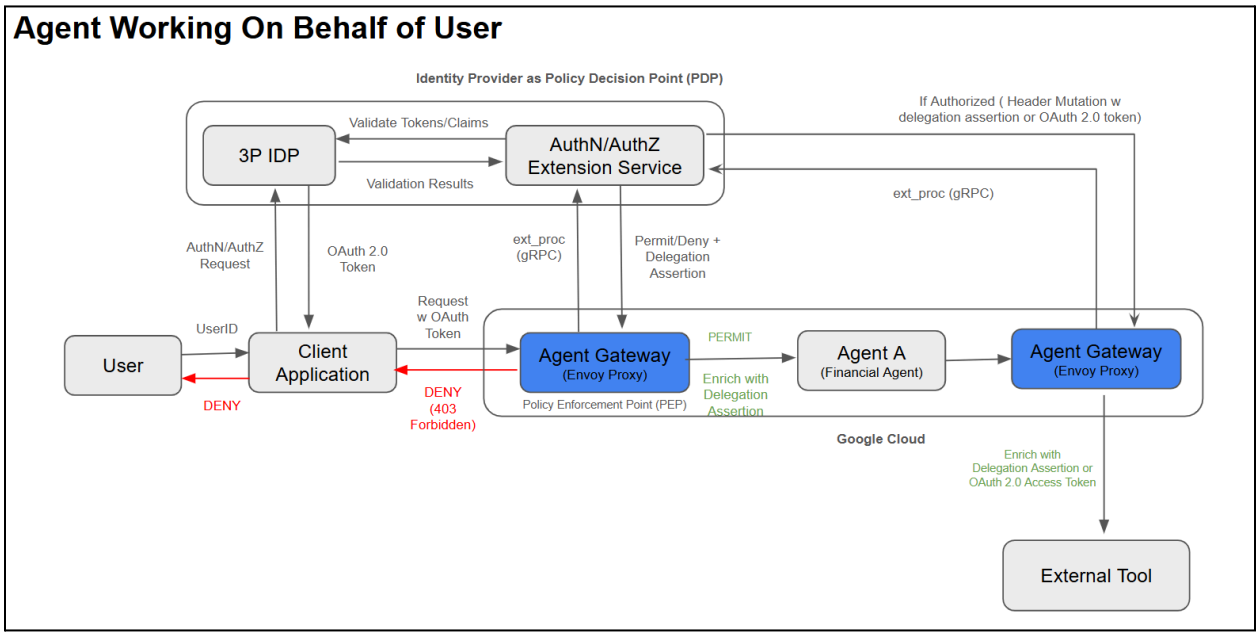
Customer Journey 2: Agent Working "On-Behalf-Of" (OBO) User

While autonomous tasks provide baseline efficiency, true enterprise value is unlocked when these agents act with the delegated authority of a human user. This "**On-Behalf-Of" (OBO)** journey creates a trust bridge between a human and their digital representative.

Consider a Google Cloud-hosted financial agent (Agent A) that must execute a high-value transaction through an external SaaS tool on behalf of an employee (human user) who has authenticated via an enterprise 3P IdP.

To ensure the above scenario works securely, the Agent Gateway must enforce "compound authorization." This authorization proves that the agent is trusted and that the person has granted explicit permission. The Agent Gateway sits between the agent and the target SaaS, intercepting the outbound call and utilizing the user's delegated token to enforce authentication and authorization.

Technical Resolution:



1. ***User Authentication:*** *The user initiates a request to a Google Cloud hosted financial agent through a client application. The client application obtains an OAuth 2.0 token (access token)*

from the 3P IdP after the user is authenticated. The client sends a request to the Agent Gateway, including an access token issued by their 3P IdP.

2. ***IdP Token Validation:*** *The Agent Gateway at the agent ingress calls the AuthN/AuthZ Extension Service. This service validates the token with the 3P IdP's endpoints or public keys and authorizes the request based on predefined policies (RBAC/ABAC).*
3. ***Delegation Token Issuance:*** *If authorized, the extension service mints and injects an appropriate delegation token (e.g., OAuth 2.0 tokens with OBO flows, Txn-Tokens, or ID-JAG, if applicable) into the request header. This token carries claims including sub, azp, scp, and actor/delegation context, allowing downstream policy decisions to be decoupled from runtime application code.*
4. ***Egress Interception:*** *The Google Financial Agent receives the request with the delegated token header. When the Agent needs to call the external tool (e.g., a SaaS API), it sends an egress request back through the Agent Gateway at the egress.*
5. ***Egress AuthN/AuthZ:*** *The Agent Gateway calls the ext_proc extension service. The service validates the Google Agent's identity and inspects the delegation token. This step enforces runtime policy (PDP) independently of token issuance.*
    - *The Google Agent is allowed to egress to the specific external tool.*
    - *The scopes in the token header permit this specific operation.*
6. ***Credential Injection & Scoping:*** *If authorized, the Extension Service performs a Token Exchange (such as OAuth 2.0 Token Exchange, standard OBO, or downscoping) to obtain a tool-specific credential, ensuring least-privilege access.*
7. ***Step-Up Authentication (Optional Human-in-the-loop (HITL)):*** *For high-risk operations, the Extension Service can trigger dynamic challenges via the 3P IdP (e.g., a push notification to the user's phone) before allowing the request.*

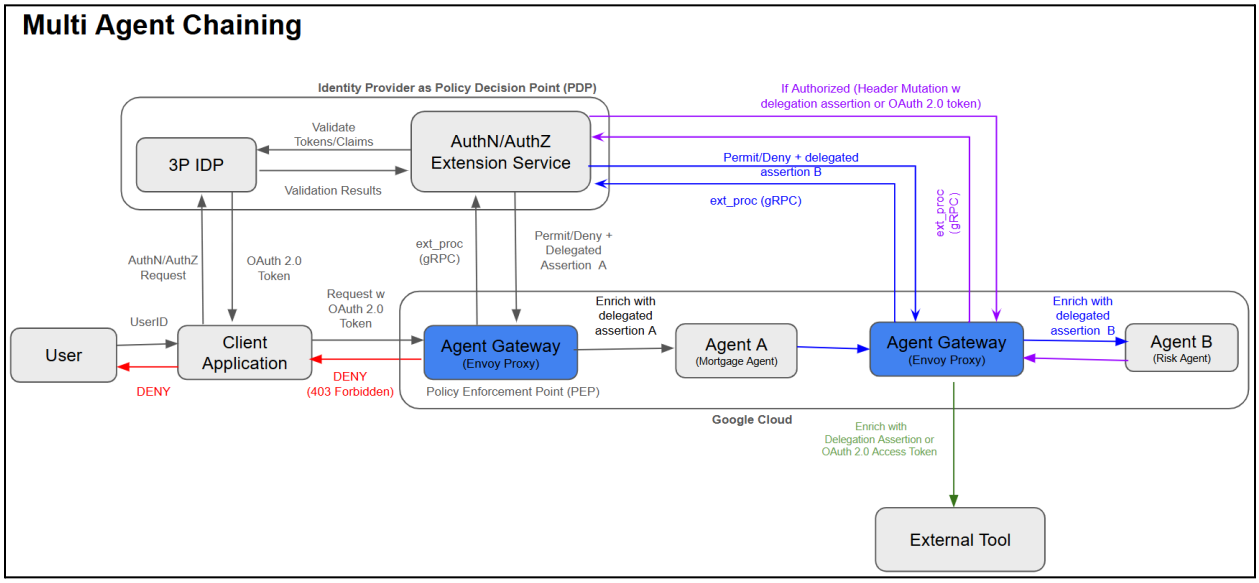
Customer Journey 3: Multi-Agent Chaining (Agent A -> B -> C)

Our final scenario layers multi-agent coordination on top of Journey 2's compound delegation framework. In complex ecosystems, tasks are decomposed across a chain of specialized agents.

Consider a mortgage processing flow: a loan officer (authenticated via their trusted 3P IdP) signing into the **Gemini Enterprise** mortgage processing agent (Agent A), which must invoke a high-code risk assessment agent (Agent B) on the **Agent Platform** to query underlying

databases (external tools). The Agent Gateway helps enforce strict governance at each hop, preventing the **"Confused Deputy" problem** where a compromised agent might attempt to exceed its authority.

Technical Resolution:



1. ***User to Agent A:*** *The loan officer authenticates with the 3P IdP and requests access to Agent A. The user's delegated authority is captured in an initial delegation token (such as OAuth 2.0 tokens with OBO flows, Txn-Tokens, or ID-JAG, if applicable) after validation.*
2. ***Agent A to Agent B:*** *When Agent A makes a request to Agent B, the request is again mediated by the Agent Gateway.*
    - *The Gateway verifies Agent A's service identity (e.g., via mTLS).*
    - *The Extension Service verifies Agent A is allowed to call Agent B.*
    - ***Delegation Token Propagation:*** *A new delegation token is minted/exchanged—nesting Agent A's identity while retaining the delegated user identity claims to maintain the full chain of trust—and propagated to Agent B. Crucially, the extension service can scope the claims per-hop (using standards such as OAuth 2.0 tokens with OBO flows, Txn-Tokens, or ID-JAG, if applicable), or enforce pure PDP checks at each hop.*
3. ***Agent B to Tool (Egress Authorization):*** *When Agent B calls the Tool, the request is routed via the Agent Gateway (egress) again.*

    - *The Gateway authenticates Agent B (e.g., via mTLS) to establish its specific workload identity.*
    - *The Extension Service checks if Agent B is authorized to call the Tool and if the scopes in the current delegation token permit this action. It ensures the egress token is downscoped to the **intersection of the chain's authority**, meaning Agent B cannot access data that Agent A or the original user were restricted from.*
4. ***Egress with Token Exchange:** If authorized and intersection checks pass, database tool credentials are obtained via Token Exchange that are scoped exactly to the validated, scoped context, thus closing the entire loop securely.*

Closing the Trust Loop: Scaling with Identity Sovereignty

These diverse architectural paths—ranging from foundational Agent-to-Agent (A2A) handshakes secured by mTLS to sophisticated OBO token exchanges and multi-agent chaining—all converge at the Agent Gateway. Ultimately, the gateway serves as more than a proxy; it is an indispensable bridge to enterprise identity sovereignty.

By integrating seamlessly with your chosen 3P IdP, the Agent Gateway reconciles varied agentic journeys under a unified security umbrella. This ensures that as the agentic transformation scales, every autonomous action remains grounded in centralized, cryptographic intent, empowering enterprises to innovate with confidence in a secure, interoperable, and audited cloud environment.

Additional Resources

Introducing Agent Gateway ISV ecosystem for security and governance

Agent Identity Overview

Agent Gateway Overview