A Protocol-Agnostic Threat Classification for Cross-Boundary AI Agent Authorization

Artificial Intelligence Identity Management Community Group

12 October 2026

A Protocol-Agnostic Threat Classification for Cross-Boundary AI Agent Authorization

Audience: Who is this report for?

This report is primarily intended for *Architects and Security Practitioners* responsible for designing and securing cross-boundary agentic systems. While the strategic implications are relevant to the C-suite, the technical depth regarding token exchange, trust anchors, and protocol-agnostic threats is scoped for those implementing these frameworks. Rather than aiming for a universal audience, this document provides the technical foundation required to address the gaps between current identity infrastructure and agentic requirements.

It should be noted that this classification serves as a baseline for a proposed multi-document structure. Future iterations will split these concerns into three distinct areas: (1) Authentication standards by use case, (2) Authorization standards by use case, and (3) A comprehensive threat model with explicit mitigation guidance.

Introduction

Organizations are deploying AI agents faster than the identity and authorization infrastructure those agents need, leaving a gap between known good practice—strong agent identities, delegated authorization, least or zero standing privilege—and what actually ships.

This paper does not try to close that gap with a solution. It threat-models one specific problem: what goes wrong when agents act across security boundaries. We call this agentic federation and define it as enabling agents in one trust domain (say, hosted on AWS) to be reliably identified, authorized, and held to a consistent permission set when they call tools or services in another (say, Azure).

It is tempting to imagine that this problem will be solved by consolidation: a single universal gateway through which all of an organization's agentic traffic is mediated. It will not. Each major provider operates its own gateway on its own assumptions about identity, trust, and scope, and none will delegate that function to a competitor—Anthropic will not route its agents through Microsoft's gateway, nor Microsoft through Google's.

Nor does the history of identity federation rescue the idea: vendor-neutral intermediaries did emerge there, but an agentic gateway sits in the request path of every tool call rather than passively translating assertions. Where an enterprise interposes its own, the provider's gateway remains behind it, so consolidation yields chained mediation under inconsistent assumptions

rather than a single point of control. What providers can realistically become is policy engines that interoperate, not a single point that rules them all.

Cross-boundary federation is therefore not a transitional inconvenience to be engineered away, but the permanent condition under which agentic systems will operate. Any real enterprise will stitch many platforms together, and the expedient choices engineers make in the gaps between them are where security boundaries erode.

That permanence is precisely why these threats are worth modeling: they are not artifacts of an immature market that will disappear as it matures, but the durable consequences of a landscape that will always have more than one gateway. Articulating them clearly is also the most effective pressure toward interoperability, giving providers and standards bodies a concrete, shared reason to converge.

This paper is not about any one technology. It does not focus on MCP, Transaction Tokens, SPIFFE, or any single specification, because the threats come from the act of crossing a boundary, no matter which mechanism is used. It takes as given the questions that apply even within a single boundary—what an agent's identity is and how delegated authorization works—and focuses only on what changes when a boundary is crossed. And it is a threat model: its purpose is to identify what can go wrong, not to prescribe how to fix it.

Terms

Agentic and federation terms are used inconsistently across the industry, with different practitioners often assigning them different meanings. This section defines them as they are used in this paper. The intent is not to propose a standardized usage, but to establish a common vocabulary for the following sections.

- **Gateway.** The control point through which an agent's calls to tools, services (including LLMs), or agents pass - whether on an originating side or terminating side. A gateway mediates requests rather than letting an agent reach a resource directly, and is the place where authorization is checked and enforced. A gateway is a control only insofar as it cannot be bypassed; the gateway discussed here is agentic—it sits in front of an agent's tool and service calls.
- **Broker.** The component responsible for translating an agent's intent into a specific, context-bound authorization. Rather than returning a persistent credential, the broker issues an authorization that is cryptographically bound to the requesting instance and the specific transaction.
- **Access and Transaction Tokens.** This framework distinguishes between identity-based entry and context-bound action. Access tokens represent the durable identity used to enter a security domain, while transaction tokens convey specific, ephemeral authority within it.
- **Agent, Instance, and Context.** An *agent* is the durable actor. An *instance* is a specific running occurrence. A *context* defines the specific access and circumstances—including intent and environment—under which an instance acts.

Threat Categories

Key Takeaways

Permanent Condition: Agentic federation is a permanent architectural reality, not a transitional phase.

Security Gaps: Boundaries fail due to reinterpretation of intent and machine-speed risks that outpace human judgment.

Authorization Focus: Effective security requires authorization-time controls (scope discipline and automated cleanup) rather than relying on behavioral detection.

The threats below are organized around six questions: *what* has access, on what basis, *what* it acts on, *why*, and *when and for how long*. The framing deliberately substitutes "what" for the more familiar "who," because the principal in an agentic system is not a human. Credential exposure, provisioning, behavioral detection, and content-as-instruction are treated as cross-cutting concerns.

The Actor: Identity Resolution ("What")

The main problem is knowing if an agent is actually who it says it is. When agents work across different systems, it is easy for them to fake their identity or use keys in the wrong place to get permissions they shouldn't have. This creates a difficult choice: if you give all agents the same identity, you cannot tell which specific one is causing trouble. But if you give every single agent a unique identity, your system becomes too big and hard to manage. To solve this, organizations often give an 'agent type' a permanent identity and then use temporary 'tickets' or claims to identify the specific agent instance. This creates a new problem at the border: the receiving system must decide whether to treat all those different agents as one, or try to check temporary claims that it might not fully trust.

The Foundation: Trust Establishment ("On What Basis")

Before identity or scope can be questioned, the two domains must establish trust in one another's assertions—which keys sign them, how those keys are distributed and rotated, and what occurs when a trust anchor is compromised or replaced. This is where human federation's durable failures live: stale metadata, manual key exchange, rotation deferred because it breaks integrations.

Central to this trust is the provenance of domain identity and the lifecycle of cryptographic keys. Effective trust establishment requires clear visibility into how keys are generated, assigned, and protected. A critical security distinction exists between organizations that leverage established, verifiable decentralized tools and those that "roll their own" custom cryptography, assertions, or

attestations. The former benefits from standardized, peer-reviewed security properties that facilitate interoperability, while the latter often introduces opaque vulnerabilities and unique failure modes that are difficult for external partners to audit or trust during cross-boundary transactions.

The Protected Resource: The Direct Object ("What It Acts On")

Even a correctly identified agent acting with legitimate authority can act on the wrong resource once a boundary is crossed. An identifier that resolves to one principal in the originating directory may resolve to a different principal, or to nothing, in the target directory: an agent intending to act on a particular named entity in one domain may reach a differently named or entirely different entity in another. Because agents operate at machine speed and scale, the consequence of acting on a misresolved resource is amplified well beyond what a human operator working at human pace would cause before the error became apparent.

Access Versus Authorization: Boundary Reinterpretation ("Why")

The primary threat at the boundary is the reinterpretation of scope. Because authorization logic is rarely identical across trust domains, a token scope issued in the originating domain is translated into a local equivalent that is often broader for convenience. This remapping reinterprets original intent as broad access, causing authority to expand precisely where it should be contained. Without a shared mechanism to preserve intent across the boundary, the receiving domain conflates the agent's technical access with its original mission, creating "confused deputy" risks where an agent exceeds its intended authority simply because the structure permitted the expansion.

Lifecycle And Cleanup: Practical Governance at Scale

Correctly granted authority becomes a durable threat if it persists beyond its immediate mission. To move from abstract lifecycle stages to actionable guidance, organizations must treat agentic lifecycles as a machine-scale version of enterprise Joiner, Mover, and Leaver (JML) processes.

1. Registry-Based Creation (Join)

Agents must not be anonymous or self-asserted. Registration must be an immutable, cryptographically verifiable event.

1. **Attested Identity:** Require verifiable identities (e.g., SPIFFE, DIDs) rather than static service accounts.
2. **Mission Definition:** Registration must capture the agent "type," its intended mission, and the responsible owner (human or legal entity). This metadata is essential for intent-based auditing.
3. **Automated Child Registration:** In multi-agent flows, child agents must be registered automatically by the parent broker, inheriting audit trails and owner responsibility.

2. Configuration Sensitivity (Move)

Any change to an agent's configuration—such as model updates, added tool access, or hosting environment shifts—is a lifecycle "move" that silently expands scope.

4. **Re-authorization Gates:** Treat all configuration changes as re-authorization events. An agent’s authority must be re-validated against its mission statement whenever its functional capabilities change.
5. **Policy-as-Code:** Use registry data to automatically apply policy constraints. If an agent's registry entry changes (e.g., "model updated"), automated policy enforcement should re-evaluate its access to high-sensitivity domains.

3. Automated Decommissioning (Leave)

"Orphaned authority"—where derived credentials outlive the task—is the primary driver of revocation sprawl.

6. **Zero-Standing Privilege (ZSP):** Shift entirely to task-bound, ephemeral transaction tokens. Revocation should be an inherent, automated side effect of task completion or expiration.
7. **Cascading Revocation:** Implement strict dependency tracking. If a parent agent or its broker is decommissioned, all associated child tokens and active sessions must be invalidated automatically and globally.
8. **Graceful Exit Patterns:** For long-running agents or external partnerships, implement a "Transition Service Agreement" (TSA) pattern. Rather than immediate hard cut-offs which break processes, define a strict, time-bound sunset period for active credentials that is automatically enforced by the gateway.

These lifecycle controls transition the architecture from manual cleanup to an automated, policy-driven model, ensuring that identity and authorization are always aligned with the agent’s current, authorized mission.

Credential And Token Exposure (Cross-Cutting)

When tokens or their contents are exposed to the calling agent, whether through bloated headers or by returning more information than the agent requires, that material can be captured and reused. This enables both immediate misuse, in which access to a token causes damage within seconds, and deferred misuse, in which captured long-lived material—refresh tokens, signing keys, or credentials cached beyond need—is exploited after the exposing transaction has ended..

Conclusion

The threats in this paper share one structure: containment that holds within a trust domain and degrades the moment an agent crosses into another. Identity resolves cleanly until two directories disagree on what an identifier means; scope is well-defined until it is reinterpreted at a boundary.

Tokens are revocable until they have been scattered across systems that do not coordinate cleanup, and tokens can be reused. The recurring lesson is that agentic federation is partly user federation's threats run at machine speed—misresolution, scope drift, and revocation sprawl all have human-federation ancestors.

It is also a class user federation never modeled: a principal whose instructions arrive through its data and whose pursuit of a goal is unchecked by human judgment. It is the extension of trust to a non-human principal that operates at machine speed and is consistent only by accident—and treating it as a human user is the common root of the failures catalogued here.

This is a threat model, not a remediation guide, but the analysis points to concrete actions. Architects should enumerate the trust boundaries their agents cross and confirm what each identifier, scope, and token means on both sides of every one, refusing the expedient anti-patterns identified here even where existing infrastructure invites them. Security owners should treat the loss of behavioral detection as a real gap and invest in authorization-time controls—scope discipline, intent preservation, and cleanup—rather than post-hoc detection that agents defeat by design. Standards bodies and platform providers should regard interoperability as a security control, since every provider running its own gateway on its own assumptions widens these gaps.