

Federated Global Agentic Factory Reference Architecture and Qualification Programme

1. Architectural Foundation and Strategic Paradigm

The transition from a localized, single-region agentic execution environment to a federated, multi-region global factory demands a fundamental paradigm shift in how authority, operational state, and identity are governed. The existing internal research establishes a highly robust operational baseline. This baseline comprises an internal canonical task, handoff, tool-intent, tool-receipt, and event architecture. It explicitly defines external A2A protocols as gateway protocols that must undergo translation into the internal canonical task model, vehemently rejecting their use as the platform's authoritative state machine. Furthermore, the established program dictates a strict control-plane and execution-plane separation; maintains comprehensive project, fleet, provider, environment, and agent registries; and mandates workload identity, just-in-time credentials, and hardware-isolated execution for untrusted workloads. It also relies on policy-as-code, bitemporal state event-first recovery, and immutable ledgers, explicitly establishing human approval as the current production-deployment boundary while recognizing human review capacity as the ultimate factory bottleneck.

The new programme cites these reports as architectural assumptions rather than rebuilding their conclusions. However, integrating these foundational elements into a seamless, provider-neutral global architecture introduces novel friction points. As multi-agent systems evolve beyond simple single-model inference, the security discourse must shift from rudimentary prompt injection toward complex authorization propagation, lateral privilege escalation, and split-brain state inconsistencies¹. The central objective of this research is to define how the established internal architectures seamlessly combine into one coherent provider-neutral framework without creating contradictory authority models, duplicated protocols, or unmanageable operational complexity.

2. Programme-Wide Architectural Principles

2.1 The Absolute Supremacy of Internal State

A core vulnerability in federated multi-agent systems is the implicit trust frequently granted to remote agents reporting task completion or status updates. In the federated factory model, internal state remains entirely authoritative. A remote agent's task status, explanation, claimed completion, reputation score, or provider session must never directly mutate the authoritative internal state machine.

To enforce this, all external communications must be algorithmically translated into internal objects. These transitional primitives include ExternalTaskOffer, RemoteTaskContract, ExternalAgentIdentity, DelegationGrant, ExternalEvidenceReceipt, RemoteArtifactReference, FederationPolicyDecision, and ExternalCompletionClaim. When an external agent communicates a finalized state, this external completion claim must enter the internal platform exclusively as evidence awaiting validation, never as a finalized, completed task. This requires a bitemporal event sourcing architecture. An auditor or deterministic controller must be capable of rewinding the event ledger to a specific tick to prove exactly what the system knew and when it knew it, thus satisfying the rigorous traceability demands of enterprise governance⁴. External inputs merely append claims to the event stream; internal deterministic arbiters resolve these claims into authoritative canonical state.

2.2 External Agents as Untrusted Service Providers

The current Agent-to-Agent (A2A) 1.0 family, stewarded by the Linux Foundation, standardizes Agent Cards, task lifecycles, structured multi-modal content, and streaming notification patterns⁵. It intentionally relies on standard transport-layer authentication and server-side authorization. However, cryptographic protocol compliance is fundamentally distinct from behavioral trust.

The presence of a valid Signed Agent Card⁷ proves only domain ownership and protocol conformity. It does not constitute proof that an agent possesses its advertised competence, handles data according to claimed privacy policies, will remain available, is immune to subliminal prompt injections, generated authentic evidence, will respect budgetary ceilings, will avoid infinite recursive delegation, or holds any legal or financial accountability⁸.

The official A2A samples themselves warn that remote Agent Cards, messages, artefacts, and status records must be handled as strictly untrusted input. Consequently, the architecture mandates that all artefacts and outputs generated by external entities immediately enter cryptographic quarantine upon receipt, requiring independent internal verification before they can influence internal operational state or production environments.

2.3 Deterministic Human Governance at the Production Boundary

The automation of cognitive reasoning and task execution must not precipitate the automation of production deployment authority. The rule that stochastic models cannot become the root of production authority remains inviolable. For production deployment, the target model operates under a strict mandate: human-defined policy authorizes a bounded class of deployment, deterministic controllers evaluate each release against that policy, and while agents may propose and initiate actions, they cannot authorize themselves.

Humans retain absolute authority over policy creation, the definition of maximum blast radii, permitted repositories and environments, financial transaction ceilings, risk class definitions, rollback requirements, revocation triggers, emergency stops, and residual-risk acceptance. This deterministic boundary ensures that the emergent, often unpredictable dynamics of multi-agent negotiation, delegation, and reasoning do not bypass established safety constraints².

2.4 Regional Independence and Static Stability

To achieve resilience against catastrophic global outages and network partitions, regional independence must be structurally stronger than global coupling. Each region or cell must be capable of continuing safe, local operations even when global directory services or telemetry streams are impaired.

A global dependency is introduced only where globally consistent authority is genuinely necessary—such as the revocation of a compromised identity root or the initial provisioning of cross-region budgets. All other operational mechanics must utilize regional ownership, asynchronous replication, immutable event sourcing, deterministic reconciliation, bounded staleness, and read-only degraded modes. By adopting a cell-based architecture, the blast radius of any failure is contained, splitting traffic and state across independent platforms and ensuring that split-brain scenarios do not corrupt the global ledger¹⁰.

2.5 Ephemeral Execution and Logical Identity Scalability

The requirement to support thousands of concurrent projects must not necessitate thousands of permanently running agent processes, dedicated Kubernetes clusters, or isolated databases. Scale is achieved by decoupling logical identity from physical compute infrastructure.

The target architecture explicitly distinguishes between registered projects, enabled projects, projects with queued work, concurrently active projects, active workflows, active agent attempts, and active sandboxes. Persistent logical identity and state storage must remain highly inexpensive and decoupled. In contrast, heavy operational resources—compute nodes, large language model inference sessions, credential brokers, and execution sandboxes—must remain strictly ephemeral and pooled⁸. When a task concludes, the sandbox is destroyed, leaving zero residual state.

2.6 Expansion Regulated by Measurable Evidence

The platform's maturity model dictates that capability expansion is not arbitrary. Each new autonomy or scale tier requires a formal threat-model update, deterministic conformance tests, adversarial scenarios evaluating multi-agent collusion, capacity tests, failure-injection tests, recovery drills, economic modeling of API consumption, and sustained operational evidence. Scale follows verifiable safety.

3. Resolving Required Integration Decisions

The synthesis of Topics 30–33 into a coherent provider-neutral architecture requires resolving several complex integration decisions to prevent duplicated protocols and unmanageable operational complexity.

3.1 Global versus Regional Project Identity and Placement

Project identity exists on a global continuum but executes regionally. The global directory assigns a Universal Resource Name (URN) to a project, providing a singular logical anchor. However, project-to-cell and cell-to-region placement are deterministic routing decisions

governed by data residency policies and latency requirements. A project's operational state is bound to a specific regional cell's event stream. If a project operates globally, the architecture dictates asynchronous replication of read-only state, while write-authority remains pinned to the designated primary cell to prevent distributed lock contention.

3.2 External versus Internal Agent Identity

External agent identity is governed by the A2A protocol and is rooted in external domain ownership (e.g., verified via Signed Agent Cards)⁷. When an external agent interacts with the platform, this identity is ingested and mapped to an internal `ExternalAgentIdentity` object. Internal agents, conversely, derive their identity from the platform's SPIFFE/SPIRE infrastructure, receiving hardware-attested SPIFFE Verifiable Identity Documents (SVIDs)¹². The two identities never merge; the internal system tracks the external identity as an untrusted actor proposing changes, while the internal SPIFFE identity is used to securely invoke internal tools and databases.

3.3 A2A Task ID versus Canonical Task ID

When a remote task contract is negotiated over A2A, an external A2A task ID is generated. This ID serves as the correlation key for horizontal communication over the external gateway. However, it is never used as the primary key for the platform's internal state machine. The system generates an internal Canonical Task ID, mapping the A2A Task ID as a secondary, foreign-key reference. All bitemporal state transitions, credential broker requests, and ledger entries utilize the Canonical Task ID, insulating the internal platform from external ID collision or spoofing attacks.

3.4 External Delegation and Regional Ownership

When an internal regional cell delegates a task to an external provider, the regional cell retains absolute ownership of the workflow's state. The remote agent may perform recursive sub-delegations within its own infrastructure, provided it adheres to the `max_depth` parameter defined in the delegation grant. However, the regional cell views this externally delegated tree as a single, opaque asynchronous operation. The external network's complexity is abstracted away, and the regional cell simply monitors the gateway for an `ExternalCompletionClaim`.

3.5 Global versus Cell-Local Budgets and Policy

Global policies (e.g., "No deployment without SLSA Level 3 provenance") and global budgets are defined in the Global Governance Plane. To ensure static stability, these are projected into Cell-Local budgets and deterministic rulesets. If a project has a global budget of \$10,000, it may be divided into \$5,000 quotas distributed to local cells. This allows regional enforcement of financial ceilings without requiring synchronous, high-latency communication with the global control plane for every API call¹¹.

3.6 Autonomous Release Authority and Emergency Control

Autonomous release authority operates strictly within regional boundaries based on standing

authorization policies. If telemetry detects anomalies, rollbacks are executed regionally. The Global Governance Plane retains emergency control authority—the ability to inject a high-priority revocation event into all regional streams to halt specific agent identities globally if a systemic compromise is detected².

3.7 Public Federation During Regional Isolation

In the event of regional isolation (a network partition separating a cell from the global plane), the cell relies on its cached trust roots and policy projections. It can continue to federate publicly via its local gateway, processing incoming remote task contracts based on its last known good state. If a task requires global consensus or access to unreachable cross-region data, the task transitions to a paused auth-required state, demonstrating a read-only degraded mode rather than a systemic failure⁶.

4. Target Reference Architecture: The Layered Model

The architecture utilizes a rigidly defined layered model. Conceptually, this structure operates as a vertical hierarchy. The Global Governance Plane sits at the apex, distributing policy and identity roots downwards. Beneath it, separated by strict isolation boundaries, reside the Regional Control Cells which manage workflow state and execution pools. Flanking these regional cells are the External Federation Gateways, which act as horizontal perimeters terminating incoming A2A traffic from external organizations. At the foundation lies the Deterministic Deployment Authority, which enforces the final barrier before artefacts alter production reality.

4.1 Global Governance Plane

The Global Governance Plane is the central administrative authority, decoupled from the high-throughput execution paths. It owns the overarching organization directory, the project directory, and the cell and region directory, ensuring global namespace consistency. Furthermore, it houses the schema registry, ensuring that all regional cells and external translation layers utilize unified data structures.

Critically, this plane owns the global policy roots and the trust roots. It serves as the ultimate Certificate Authority (CA) for the internal infrastructure, governing the issuance of identities and the extension and agent certification registries¹². It manages global budgets, monitoring aggregate burn rates across all federated regions. Lastly, it holds emergency authority, possessing the cryptographic keys necessary to broadcast global revocation lists and halt factory operations universally in the event of a critical security breach.

4.2 Regional Control Cells

The Regional Control Cells represent the decentralized engines of the agentic factory. Designed around the principles of cell-based architecture, they isolate failures and prevent regional outages from cascading globally¹⁰. They own the project operational state and are responsible for workflow execution.

Each cell houses its own localized scheduler, a projection of the global project policy, and a

sophisticated model routing layer that directs prompts to the optimal inference endpoints. The cell contains a credential broker that facilitates just-in-time access, interacting with the ephemeral execution pools where the actual compute occurs. To guarantee auditability and recovery, the cell maintains its own local event streams and observability infrastructure, logging every state transition in a bitemporal ledger⁴. The regional cell also handles the localized release execution once an artefact has cleared the deterministic deployment boundary.

4.3 External Federation Gateways

The External Federation Gateway is the primary defensive perimeter, managing all interactions with untrusted service providers. It owns protocol termination, ensuring that the external A2A v1.0 traffic—comprising HTTP transports, JSON-RPC messages, and Server-Sent Events (SSE)—never penetrates the internal network⁵.

When an interaction initiates, the gateway performs remote discovery, fetching and caching external agent capabilities. It conducts trust evaluation by cryptographically verifying Signed Agent Cards against the global certification registry⁷. It negotiates remote task contracts and performs external identity translation, mapping external entities to internal representations. Most importantly, it governs artefact quarantine; any data or code returned by an external agent is securely isolated here until internal deterministic processes can independently verify its safety and utility. The gateway also handles billing and dispute receipts, leveraging extensions like the Agent Payments Protocol (AP2) to coordinate economic transactions without exposing internal financial systems¹⁵.

Protocol Demarcation: A2A vs. MCP Boundaries

Dimension	A2A Protocol (External)	MCP Protocol (Internal)
Primary axis	Horizontal (agent ↔ agent)	Vertical (agent ↔ tools/data)
Creator	Google (April 2025)	Anthropic (November 2024)
Governance	Linux Foundation (June 2025)	Linux Foundation / AAIF (Dec 2025)
Interaction model	Peer-to-peer, opaque agents negotiating tasks	Client-server, model invokes tools
Discovery	Agent Card (/well-known/agent-card.json)	Capability handshake at connect time
State model	Explicit state machine per Task	Mostly stateless tool calls
Execution location	On another network	Inside your process or a local server
Long-running ops	First-class (input-required, streaming, push)	Limited, ad-hoc
Execution duration	Minutes to hours, pauses included	Milliseconds
Typical use	Procurement agent ↔ supplier agent, multi-agent orchestration	Agent ↔ GitHub, Agent ↔ Postgres

The External Federation Gateway terminates the **horizontal** A2A protocol, translating external state into internal tasks. Internal agents then utilize the **vertical** Model Context Protocol (MCP) to interact securely with internal tools and databases.

Data sources: [Stellagent.ai](#), [Linux Foundation](#), [Mastra.ai](#)

4.4 Deterministic Deployment Authority

This layer enforces the absolute rule that stochastic AI models cannot authorize production

changes. It acts as a rigid, algorithmic gatekeeper. It owns release-evidence validation, inspecting cryptographic signatures and software bills of materials (SBOMs). It conducts standing-authorization validation, ensuring that a proposed deployment explicitly matches the human-defined parameters for risk and blast radius.

The authority manages progressive rollouts, carefully expanding the reach of a new artefact while conducting runtime analysis. If regressions or security anomalies are detected, it owns the automated rollback sequence, immediately reverting the system to the last known good state. It also possesses revocation capabilities, able to invalidate specific artefacts or agent access grants on demand. Finally, it coordinates post-release completion, finalizing the canonical task status once the deployment has been verifiably successful in production¹⁶.

4.5 Ephemeral Execution Plane

The execution plane is the highly scalable, untrusted environment where the actual computational work of agents takes place. It owns absolutely no durable authority. It receives only exactly what it needs for the duration of a single task.

Upon instantiation, a sandbox receives a task-bound identity, tying its actions strictly to a specific internal canonical task. It is provided with scoped context and temporary credentials brokered via identity token exchange mechanisms¹⁷. The environment enforces hard execution limits (compute time, memory constraints, API quotas) and operates under an explicit network policy, preventing lateral movement or unauthorized egress⁸. Once the task concludes, the execution sandbox is cryptographically wiped and destroyed, eliminating the risk of persistent advanced threats.

5. Agentic Zero Trust and Identity Governance

Classical Zero Trust Architecture (ZTA)—as codified in NIST SP 800-207—provides a rigorous foundation but was designed for a world of human users and static services. It is necessary but fundamentally insufficient for agentic AI¹⁹. As autonomous agents retrieve data, dynamically delegate tasks across organizational boundaries, and synthesize results, traditional identity perimeters dissolve. A multi-agent system creates a distinct authorization propagation problem: ensuring that authority does not blindly accumulate through delegation chains, resulting in privilege escalation¹.

5.1 SPIFFE/SPIRE for Hardware-Attested Workload Identity

To combat these vulnerabilities, the architecture treats identity governance as foundational infrastructure. Every internal agent and ephemeral execution sandbox is issued a cryptographically verifiable identity using SPIFFE (Secure Production Identity Framework for Everyone), managed by a SPIRE control plane¹².

When a sandbox boots within the Ephemeral Execution Plane, it does not rely on static API keys or long-lived service accounts, which are highly susceptible to extraction via prompt injection. Instead, an embedded SPIRE agent attests the workload to the SPIRE server using X.509 Proof of Possession. The workload receives a short-lived SPIFFE Verifiable Identity Document (SVID)—either an X.509 certificate or a JWT—tied to a unique identifier (e.g.,

spiffe://global-factory.internal/region-eu/agent/researcher-01)¹². This anchors the identity of the agent to the actual cryptographic and hardware state of the node, establishing a verifiable root of trust without human-touched secrets.

5.2 The Agent Authorization Profile (AAP) and Token Exchange

While SPIFFE establishes *who* a workload is, it does not govern *what* it is permitted to do across varied external and internal APIs¹². For authorization, the platform implements the Agent Authorization Profile (AAP), an advanced profile built on OAuth 2.0 and JWTs specifically designed for autonomous AI agents¹⁷.

When an agent needs to execute a task, it exchanges its SVID for an AAP-compliant access token via RFC 8693 Token Exchange¹². This token encapsulates structured, context-aware claims that resource servers explicitly evaluate. These claims include:

- `aap_task`: Cryptographically binds the token to a specific internal canonical task and purpose.
- `aap_capabilities`: Explicitly defines authorized actions and constraints (e.g., `domains_allowed`, `max_requests`).
- `aap_delegation`: Tracks the precise depth and chain of delegation.
- `aap_oversight`: Specifies human approval requirements for certain high-risk actions¹⁷.

Crucially, this mechanism solves the transitive delegation problem. When Agent A delegates a sub-task to Agent B, the system enforces scope attenuation. The token exchange mechanism ensures the downstream agent receives a token with equal or fewer privileges than its parent⁸. Authority strictly decays across hops, preventing a compromised sub-agent from escalating privileges back up the chain².

5.3 Provenance and Supply Chain Integrity (SLSA v1.0)

The governance of agent identity must extend to the artifacts they produce. A signature on an artifact without provenance is merely an unverified "trust-me" claim, while provenance without a signature offers no tamper evidence²¹. The architecture strictly adheres to the Supply-chain Levels for Software Artifacts (SLSA) v1.0 framework²².

Any code or configuration generated by an agent must be processed by an isolated, ephemeral build platform. This builder generates an in-toto attestation (predicateType: "https://slsa.dev/provenance/v1") detailing exactly how the artifact was produced²³. This provenance document records the `externalParameters` (the inputs or prompts provided to the agent), the `resolvedDependencies` (the specific models or external data accessed), and the `builder.id`²⁴.

The Deterministic Deployment Authority leverages this data to enforce Policy-as-Code. Before any artifact can be promoted to a production canary, the controller programmatically verifies the SLSA attestation, ensuring the artifact was generated within a secure, authenticated sandbox and that all `aap_oversight` requirements were met²¹.

6. Integrated Qualification Scenario: End-to-End

Execution

To validate the integration of external federation, zero-trust identity, SLSA provenance, and deterministic deployment, the platform must successfully execute a rigorous end-to-end qualification scenario spanning all architectural planes.

The following table summarizes the chronological lifecycle of this qualification process, mapping the theoretical architecture to a concrete sequence of events, from external delegation through to automated rollback.

Phase	Stage	Architectural Action and State Transition
Phase 1: Federation & Delegation	1. External Delegation	A project in the EU delegates a task. The Regional Control Cell initiates the request. The External Federation Gateway verifies the remote agent's Signed Agent Card and establishes a RemoteTaskContract.
	2. Constrained Subdelegation	The remote agent subdelegates the task within its own network. The Gateway enforces the max_depth parameter via A2A protocol tracking, ensuring the agent operates within permitted boundaries.
	3. Cryptographic Quarantine	The remote agent claims completion and returns the result. The Gateway translates this into an ExternalCompletionClaim. The artefact is immediately placed in an immutable quarantine bucket; internal canonical state remains

		unchanged.
Phase 2: Provenance & Release	4. SLSA Build and Sign	An internal ephemeral builder (authenticated via SPIRE) inspects the quarantined artefact. It executes a secure build, generating a cryptographically signed SLSA v1.0 in-toto attestation detailing the provenance of the untrusted code.
	5. Policy Check and Canary	The Deterministic Deployment Authority evaluates the SLSA attestation against standing policy-as-code. Verifying the provenance and human oversight claims, it authorizes a progressive canary rollout scoped only to the EU cell.
Phase 3: Resilience & Rollback	6. Regional Network Partition	During the rollout, a network failure isolates the EU region. Leveraging static stability, cell-based architecture, and cached policies, the EU cell continues the deployment autonomously. Global routing is unaffected.
	7. State Transfer	Task ownership transfers smoothly. Local bitemporal ledgers ensure the exact deployment state is known, preventing split-brain double-deployments upon

		network reconnection.
	8. Regression Detection	The Deterministic Deployment Authority's local telemetry detects a seeded business regression (e.g., elevated error rates) within the 1% canary traffic.
	9. Automated Rollback	Because rollback gates were defined prior to rollout, the regression triggers an immediate, automated rollback, reverting the deployment to the last known good immutable state.
Phase 4: Audit & Revocation	10. Audit Reconstruction	An auditor reconstructs the entire event timeline from a single correlation chain in the bitemporal ledger, tracing from the initial SPIFFE identity to the SLSA provenance and the automated rollback trigger.
	11. Agent Revocation	Telemetry confirms the external agent provided a malicious payload. The Global Governance Plane revokes the agent's Public Key, propagating the ban to all regional gateways.
	12. System Preservation	Due to strict cell isolation and memory boundaries, the compromise is entirely contained. Thousands of other federated projects continue operating

		seamlessly without invalidation.
--	--	----------------------------------

This scenario proves that the architecture successfully integrates untrusted external agents while maintaining absolute deterministic control over production state and system resilience.

7. Deliverables and Migration Strategy

The deployment of the Federated Global Agentic Factory requires a structured, phased implementation roadmap to safely migrate from the existing internal single-region Agent OS without disrupting active workloads. The research program mandates the delivery of several key artifacts to formalize this transition.

7.1 Unified Trust Boundary and Authority Matrix

The programme must finalize a Unified Trust Boundary Diagram detailing the exact network and identity perimeters between the five architectural planes. Accompanying this is the Unified Authority Matrix, which serves as the definitive ruleset for the Deterministic Deployment Authority. This matrix explicitly maps the intersection of Agent Authorization Profile (AAP) claims, SLSA Provenance levels, and required human governance overrides.

For example, the matrix will dictate that an "Internal Tool Use (MCP)" action requires a SPIFFE SVID and an AAP JWT, while a "Canary Deployment" action strictly requires SLSA Level 3 provenance (Isolated and Signed) and Bitemporal Consensus from the deployment controller.

7.2 Catalogues and Threat Models

The integration necessitates comprehensive updates to existing platform registries. The programme will deliver Global, Regional, and External Schema Catalogue Additions, defining the exact JSON-RPC schemas and Protocol Buffer definitions required for gateway translation. An Integrated Threat-Model Update must be published, specifically addressing the novel attack vectors of multi-agent architectures, including transitive delegation abuse, aggregation inference, and asynchronous memory poisoning¹. Finally, an Integrated NFR, SLI, and SLO catalogue will define the performance baselines required for latency, event ledger consistency, and gateway throughput.

7.3 Platform Maturity Model and Migration Roadmap

Expansion must follow measurable evidence. The migration roadmap is structured across defined maturity tiers:

- **Tier 1: Foundation (Internal Zero Trust):** Deployment of the Global Governance Plane and the underlying SPIFFE/SPIRE PKI infrastructure. Existing internal agents are migrated to ephemeral sandboxes utilizing SVIDs, establishing the baseline workload attestation layer.
- **Tier 2: Cell Partitioning (Resilience):** The single-region architecture is fractured into

independent Regional Control Cells. Bitemporal event sourcing ledgers are activated. Static stability is empirically validated through chaos engineering, intentionally severing the global control plane to prove local cells survive.

- **Tier 3: External Federation (A2A Integration):** The External Federation Gateways are deployed. The platform begins accepting ExternalTaskOffers from external organizations via A2A v1.0. The artefact quarantine and SLSA v1.0 provenance pipelines are brought online to handle untrusted incoming code.
- **Tier 4: Autonomous Production (Deterministic Authority):** The Deterministic Deployment Authority assumes full control. Human-in-the-loop approvals for routine deployments are deprecated in favor of policy-as-code evaluations of SLSA attestations. Automated progressive rollout and rollback mechanisms become the sole authoritative deployment boundary.

7.4 Explicit List of Deferred Capabilities

To maintain project focus and manage operational complexity, specific advanced capabilities are explicitly deferred until defined trigger conditions are met.

1. **Cross-Domain Identity Federation:** Allowing external platforms to assert granular internal identity beyond standard A2A task tracking is deferred. This capability will only be explored once the Agent Network Protocol (ANP) matures sufficiently to complement the A2A standard⁶.
2. **Decentralized Offline Attenuation:** The use of capability-based token formats that allow offline attenuation without contacting a central authorization server (e.g., Macaroons or Biscuits) is deferred¹⁸. The platform will rely on centralized RFC 8693 token exchange for scope down-scoping until the latency overhead of centralized exchange breaches the established SLOs.

8. Conclusion

The Federated Global Agentic Factory Reference Architecture provides a comprehensive, provider-neutral blueprint for scaling autonomous operations while retaining absolute deterministic control over enterprise reality. By combining cell-based regional isolation, bitemporal event sourcing, and SLSA-verified supply chain integrity, the architecture mitigates the profound risks inherent in multi-agent systems.

The implementation of Agentic Zero Trust—anchoring identity in hardware-attested SPIFFE documents and enforcing capability constraints via the Agent Authorization Profile—ensures that authority cannot blindly accumulate across opaque delegation chains. By treating all external multi-agent collaboration via the A2A protocol as inherently untrusted input subject to strict cryptographic quarantine, the platform protects its internal canonical state from external manipulation. Ultimately, this architecture guarantees that while stochastic AI models may generate vast quantities of highly valuable work, the ultimate authority over production environments, financial risk, and system integrity remains deterministically tethered to human intent.

This is for informational purposes only. For medical advice or diagnosis, consult a professional.

Works cited

1. Authorization Propagation in Multi-Agent AI Systems: Identity Governance as Infrastructure - arXiv, <https://arxiv.org/pdf/2605.05440>
2. AgenticCyOps: Securing Multi-Agent AI Integration in Enterprise Cyber Operations - arXiv, <https://arxiv.org/pdf/2603.09134>
3. Architecture Matters for Multi-Agent Security - arXiv, <https://arxiv.org/pdf/2604.23459>
4. (PDF) Bounded LLM Reasoning: A Neuro-Symbolic Blackboard Architecture for Production Document Analysis - ResearchGate, https://www.researchgate.net/publication/410497174_Bounded_LLM_Reasoning_A_Neuro-Symbolic_Blackboard_Architecture_for_Production_Document_Analysis
5. Agent-to-Agent (A2A) Protocol: Implementation and Trade-offs - n8n Blog, <https://blog.n8n.io/agent-to-agent-protocol/>
6. What is the Agent2Agent (A2A) protocol? How AI agents delegate work - Mastra, <https://mastra.ai/blog/what-is-agent-to-agent-protocol>
7. A2A Protocol Explained: How Google's Agent-to-Agent Standard Grew to 150+ Organizations in One Year | Stellagent, <https://stellagent.ai/insights/a2a-protocol-google-agent-to-agent>
8. Authorization Propagation in Multi-Agent AI Systems: Identity Governance as Infrastructure, <https://arxiv.org/html/2605.05440v1>
9. AINTMA: Agentic AI Architecture for Autonomous Test Management with Generative Intelligence, Secure Cloud Communication and Adap - arXiv, <https://arxiv.org/pdf/2607.20452>
10. Building Streaming Infrastructure That Scales: Because Viewers Won't Wait until Tomorrow, <https://www.infoq.com/articles/evolution-backend-streaming-application/>
11. Cell-Based Architecture (Part 5): Multi-Platform Cells — Resilience Across Clouds and On-Prem - CloudToAILearn, <https://cloudtoailearn.dev/posts/resiliency/cell-based-architecture-part-5-multi-platform-cells/>
12. SPIFFE vs. OAuth: Access Control for Nonhuman Identities - Aembit, <https://aembit.io/blog/spiffe-vs-oauth-access-control-nonhuman-identities/>
13. Control plane and data plane - Reducing the Scope of Impact with Cell-Based Architecture, <https://docs.aws.amazon.com/wellarchitected/latest/reducing-scope-of-impact-with-cell-based-architecture/control-plane-and-data-plane.html>
14. Stuff The Internet Says On Scalability For December 19th, 2020, <https://highscalability.com/stuff-the-internet-says-on-scalability-for-december-19th-202/>
15. A2A Protocol Surpasses 150 Organizations, Lands in Major Cloud Platforms, and Sees Enterprise Production Use in First Year - Linux Foundation,

<https://www.linuxfoundation.org/press/a2a-protocol-surpasses-150-organizations-lands-in-major-cloud-platforms-and-sees-enterprise-production-use-in-first-year>

16. Canary Deployment Pattern: Progressive Delivery Guarded by SLOs - Abstract Algorithms,
<https://www.abstractalgorithms.dev/canary-deployment-pattern-progressive-delivery-with-slos>
17. Agent Authorization Profile (AAP) for OAuth 2.0 - IETF Datatracker,
<https://datatracker.ietf.org/doc/draft-aap-oauth-profile/>
18. The Enterprise Blueprint for Agentic Identity: Workload Attestation and Lifecycle Governance,
<https://www.indigoconsulting.ca/blog/the-enterprise-blueprint-for-agentic-identity-workload-attestation-and-lifecycle-governance/>
19. Agentic Zero Trust: Extending the Zero Trust Security Paradigm to Autonomous AI Systems - Cequence Security,
<https://www.cequence.ai/wp-content/uploads/2026/05/Agentic-Zero-Trust-Research-Paper-v3.pdf>
20. Agent Authorization Profile (AAP) for OAuth 2.0 - IETF,
<https://www.ietf.org/archive/id/draft-aap-oauth-profile-00.html>
21. Artifact Provenance and Attestations: From SLSA to in-toto | Secure Pipelines,
<https://secure-pipelines.com/ci-cd-security/artifact-provenance-attestations-slsa-in-toto/>
22. What is the SLSA Framework? - JFrog,
<https://jfrog.com/learn/grc/slsa-framework/>
23. Provenance - SLSA.dev, <https://slsa.dev/spec/v1.0-rc1/provenance>
24. Provenance - SLSA.dev, <https://slsa.dev/spec/v1.0/provenance>