International Journal of Leading Research Publication (IJLRP)

E-ISSN: 2582-8010 ● Website: www.ijlrp.com ● Email: editor@ijlrp.com

A Reference Architecture for Governing Autonomous, Non-Human Identities in the Regulated Enterprise

Mr. Madhusudan Kousik Kappaganthula

IAM Practice Leader
Cyber Security Services, IBM

Abstract

Autonomous software agents built on large language models — commonly described as agentic AI — are entering enterprise production at a scale and velocity that existing identity and access management (IAM) programs were never designed to absorb. Unlike traditional automation, these agents reason, plan, invoke tools, spawn sub-agents, and act on behalf of human users, frequently executing thousands of privileged operations per minute across multiple cloud environments. They are, in effect, a new and rapidly proliferating class of non-human identity (NHI) whose behaviour is probabilistic, whose lifespan can be measured in seconds, and whose permissions — if left unmanaged — create an unbounded and largely invisible attack surface.

This paper presents a comprehensive, vendor-neutral reference architecture for securing agentic AI through the lens of identity: how agents are discovered, issued verifiable identities, authenticated, delegated authority, authorized, granted ephemeral credentials, governed across their lifecycle, and continuously observed. We treat agent identity as a first-class identity category, distinct from but interoperable with human and conventional machine identities, and we anchor the architecture on established open standards for federation and token exchange, on zero-standing-privilege access, and on continuous, policy-based authorization.

Our central contribution is an agentic identity control plane — a layered set of governance, registry, runtime-enforcement, and build-time controls positioned between agent runtimes and the enterprise resources they act upon. Around this control plane we develop a maturity model for just-in-time credential issuance, a mandate-based delegation framework for on-behalf-of (OBO) execution, a closed-loop model for behavioural risk scoring and automated response, and a discovery approach that renders the full population of agents — including unsanctioned "shadow" agents — continuously visible in a unified identity graph. We map the architecture to widely adopted risk and assurance frameworks, illustrate it with two anonymized enterprise case studies, and close with a discussion of open problems including cross-organizational agent gateways, financial-grade agent identity, consent, decentralized credentials, and post-quantum readiness.

1. Introduction

1.1 Motivation

For three decades, enterprise security has organized itself around a comparatively stable premise: identities are mostly human, they change slowly, and the systems they use are relatively static. Access management matured against that premise — joiner-mover-leaver processes, periodic access certifications, role-based entitlements, privileged access vaults, and audit trails keyed to a named person. Even the first wave of machine identities (service accounts, API keys, workload certificates) was, in practice, managed as a slow-moving extension of the human model.

Agentic AI breaks that premise decisively. An agent is not a static service account: it interprets natural-language intent, decomposes goals into steps, chooses which tools and APIs to call, and can create additional agents to parallelize its work. It may be instantiated to service a single request and destroyed seconds later, or it may persist as a long-running autonomous process. It can be redirected by adversarial input embedded in the very data it was asked to read. And it typically acts with credentials — often broad, long-lived credentials — that were provisioned for convenience rather than for least privilege. The combination of autonomy, speed, scale, and delegated authority means that a single compromised or misdirected agent can inflict damage far faster and far wider than a compromised human account.

The strategic value of agentic AI is nonetheless substantial, which is precisely why it is being adopted so aggressively. The security question is therefore not whether to permit agents, but how to permit them safely: how to grant an agent exactly the access it needs, for exactly the purpose it was invoked, for exactly as long as it needs it, on behalf of an accountable human, under a policy that can be evaluated and revoked in real time — and how to prove, after the fact, precisely what every agent did and why it was allowed to.

1.2 The identity gap

Industry analysis through 2025 and into 2026 has converged on a common diagnosis: non-human identities now vastly outnumber human ones in most enterprises, and agentic AI is accelerating that ratio further; yet the governance controls applied to NHIs remain immature relative to those applied to humans. Machine and agent identities are frequently over-permissioned, rarely certified, seldom owned by a clearly accountable individual, and often authenticated with static secrets that never expire. Where human identity has decades of governance scaffolding, the agent population sits in a governance vacuum.

This gap is not merely a matter of tooling. It reflects a conceptual lag: organizations have not yet agreed on what an agent identity is, who owns it, what it means for an agent to act "on behalf of" a person, how an agent's authority should be bounded and expressed, or how an agent's actions should be attributed in an audit. Closing the gap requires both a conceptual model and a concrete architecture. This paper offers both.

1.3 Contributions

This paper makes the following contributions:

- **A threat model for agentic identity** that articulates why the autonomy, velocity, delegation, and proliferation of AI agents defeat traditional IAM assumptions, and derives a set of design requirements from it.

- **An agentic identity control plane** — a layered reference architecture comprising a governance layer, an identity registry, a runtime-enforcement pipeline, build-time enablement, and a pervasive traceability layer — positioned between heterogeneous agent runtimes and enterprise resources.
- **A just-in-time (JIT) access maturity model** (levels L0–L4) that classifies credential patterns from static embedded keys to per-invocation, purpose-bound, policy-validated ephemeral credentials, and prescribes targets by agent risk tier.
- **A mandate-based delegation framework** for the on-behalf-of pattern, capturing human intent, permitted actions, scope, delegation depth, and expiry as verifiable claims carried in a cryptographically bound token chain.
- **A closed-loop observability and response model** that unifies agent telemetry, builds behavioural baselines, computes dynamic risk scores, and orchestrates automated multi-vector remediation at machine speed.
- **A discovery approach and unified identity graph** for continuous, multi-vector detection of agents — including shadow agents — and mapping of the human-to-agent-to-entitlement-to-data access paths that reveal where sensitive data is exposed.
- **A mapping to recognized assurance frameworks**, two anonymized enterprise case studies, and an implementation methodology grounded in a validate-before-scaling philosophy.

1.4 Scope, terminology, and structure

This paper is deliberately technology-neutral. It refers to capability categories — an identity provider, an identity governance and administration (IGA) platform, a secrets-management/credential broker, a policy decision point, a security information and event management (SIEM) system, a cloud agent runtime — rather than to any specific commercial product. Where we reference technology by name, we reference only open, vendor-neutral standards and frameworks: authorization and federation protocols such as OAuth 2.0, OpenID Connect (OIDC), JSON Web Tokens (JWT), mutual TLS (mTLS), and token exchange; workload-identity constructs such as short-lived, cryptographically verifiable identity documents; open telemetry conventions; and public risk frameworks. This makes the architecture portable across vendors and cloud providers, which is itself a design goal.

The remainder of the paper is organized as follows. Section 2 surveys background and related work. Section 3 develops the problem statement and threat model. Section 4 states design principles. Section 5 presents the reference architecture. Sections 6 through 13 detail each capability domain — discovery, identity issuance and lifecycle, authentication, delegation, authorization, just-in-time access, cross-domain interactions, and observability. Section 14 describes an implementation methodology; Section 15 presents anonymized case studies; Section 16 discusses risks and limitations; Section 17 maps the architecture to assurance frameworks; Section 18 discusses future directions; and Section 19 concludes.

2. Background and Related Work

2.1 From human identity to machine and non-human identity

Enterprise IAM has historically been structured into a small number of well-understood disciplines. Identity governance and administration (IGA) manages the lifecycle of identities and their entitlements, runs access certification campaigns, enforces separation-of-duties (SoD) policy, and produces audit evidence. Access management handles authentication and single sign-on, increasingly via federation protocols. Privileged access management (PAM) secures and brokers the most sensitive credentials. Secrets management stores and issues machine credentials such as database passwords, API keys, and certificates. Each discipline grew up primarily to serve human users, with machine accounts treated as a secondary concern.

Non-human identity (NHI) refers collectively to the service accounts, workloads, automation scripts, bots, API keys, tokens, and certificates that authenticate without a human at the keyboard. In most enterprises, NHIs already outnumber human identities by a wide margin, and they are disproportionately implicated in breaches because they are over-permissioned, long-lived, and weakly governed. The agentic-AI wave is best understood as a dramatic acceleration and complication of the NHI problem rather than as an entirely new one — but the complication is severe enough to warrant a distinct treatment.

2.2 The emergence of agentic AI

An AI agent, for the purposes of this paper, is an autonomous or semi-autonomous software entity that (a) receives a goal or intent expressed in natural language, (b) plans a sequence of actions to achieve it, (c) invokes external tools, APIs, data stores, or other agents to carry out those actions, and (d) observes results and adapts its plan. Agents may run inside enterprise-built applications, be embedded within third-party software-as-a-service products, execute on managed cloud agent-runtime platforms, run locally in browsers or on developer endpoints, or be operated by external partners. They increasingly interact with one another — agent-to-agent (A2A) — and invoke standardized tool and context services to reach enterprise systems.

Two properties distinguish agents from prior automation. First, non-determinism: an agent's behaviour is a function of a probabilistic model and its context window, so the same input can yield different action sequences, and adversarial content injected into that context can hijack the agent's behaviour. Second, emergent scale: agents create sub-agents, run in parallel, and are instantiated and destroyed continuously, so the identity population is both large and highly dynamic. These properties are what make static, periodic, human-oriented controls inadequate.

2.3 Existing building blocks and open standards

The architecture in this paper does not discard existing IAM investments; it extends them. The relevant building blocks and standards include:

- **Federation and token protocols:** OAuth 2.0 and OIDC for delegated authorization and federated authentication; JWT as a signed, verifiable token format; and OAuth 2.0 Token Exchange (the pattern standardized in RFC 8693) for minting narrowly-scoped, short-lived tokens that carry a delegation ("actor") chain.

- **Rich authorization requests:** mechanisms for pushing and signing authorization requests and for requesting fine-grained, resource-level authorization detail, allowing intent and constraints to be bound into the request itself.
- **Workload identity:** an open framework for issuing short-lived, cryptographically verifiable workload identity documents (generically, SVIDs) via platform attestation, so that a running agent can prove it is a genuine, expected workload without any embedded static secret; and mutual TLS for authenticated, encrypted service-to-service channels bound to those identities.
- **Policy-as-code:** externalized policy decision points (PDPs) evaluated at policy enforcement points (PEPs), expressing authorization as versioned, testable code rather than as scattered application logic.
- **Zero Trust:** the architectural principle that no identity — human or machine — is implicitly trusted by virtue of network location, and that every access is authenticated, authorized, and continuously evaluated against policy.

2.4 Risk and assurance frameworks

Several public frameworks provide the governance context for agentic identity. A widely used AI risk management framework organizes controls into four core functions — Govern, Map, Measure, and Manage — providing a lifecycle discipline for identifying and treating AI-specific risks. An international management-system standard for AI provides certifiable requirements for an AI management system, including accountability, risk treatment, and operational controls. Industry consortia have begun to publish agent- and NHI-specific governance guidance, including an agentic identity governance framework and analyses of the non-human identity governance vacuum. Financial-services supervisory expectations add further requirements around operational resilience, model risk management, third-party risk, and auditability. Section 17 maps the architecture to these frameworks explicitly.

2.5 Open problems this paper addresses

Despite the maturity of individual building blocks, several problems remain unsolved in combination for agentic AI: there is no agreed canonical identity for an agent that spans platforms and clouds; delegation semantics for "agent acting on behalf of a user, who in turn invokes another agent" are not consistently expressed or enforced; standing privilege for agents is pervasive and dangerous; discovery of the full agent population (especially shadow agents) is largely manual and immediately stale; and audit trails rarely correlate an agent's identity, its delegated authority, its actions, and the resulting outcomes into a single forensic record. The architecture that follows targets exactly these gaps.

3. Problem Statement and Threat Model

3.1 Why agents defeat traditional IAM assumptions

Traditional IAM rests on assumptions that agentic AI violates. It assumes identities are enumerable and slow-changing; agents are created and destroyed continuously. It assumes access can be certified periodically; an agent may hold an entitlement for its entire operational life of a few seconds, long expired before any quarterly review. It assumes a human is ultimately behind each action; an agent may act autonomously, or on behalf of a user, or on behalf of another agent acting on behalf of a user. It assumes credentials are relatively stable secrets; agents frequently carry broad, long-lived keys embedded in configuration. And it assumes the identity behaves deterministically; an agent's behaviour is probabilistic and can be steered by adversarial input.

3.2 Characteristics of agentic identities

We characterize agent identities along the dimensions that most affect security posture:

Dimension	Human identity	Conventional NHI	Agentic identity
Lifespan	Years	Months to years	Seconds to months; highly variable
Behaviour	Deterministic-ish, habitual	Deterministic	Probabilistic; steerable by input
Action rate	Human-paced	Scripted, bounded	Thousands of actions per minute
Delegation	Explicit, rare	Static service binding	Dynamic, nested (OBO and A2A)
Credentials	Password + MFA	Static key/secret	Should be ephemeral, per-invocation
Ownership	The person	A team (often unclear)	Must be an accountable human owner
Proliferation	Managed onboarding	Moderate	Self-spawning sub-agents; shadow AI

Table 1. Comparative characteristics of identity classes. Agentic identities combine the worst governance properties of prior classes with new ones.

3.3 Threat model

We assume a capable adversary who may submit crafted input to an agent, compromise a dependency or tool the agent invokes, or gain a foothold on an endpoint where agents run. Within this model, the salient threats are:

- **Prompt injection and coercion.** Adversarial instructions embedded in data the agent reads (documents, web pages, tool outputs) hijack the agent into misusing its legitimate permissions — exfiltrating data, invoking destructive tools, or escalating scope. Because the credentials are valid, traditional controls see authorized activity.

- **Excessive blast radius from standing privilege.** An agent holding broad, long-lived credentials that is compromised or misdirected can act across every resource those credentials reach, at machine speed, before a human can intervene.
- **The confused-deputy and proxy problem.** A powerful agent can be induced to perform actions on behalf of an unauthorized party, becoming an uncontrolled proxy. Without a mandate that binds the action to a specific authorized user and purpose, the agent's own privileges become everyone's privileges.
- **Delegation and privilege escalation across agents.** In multi-agent orchestration, a downstream agent may inherit or accrete privileges beyond those of the initiating user, or a delegation chain may be forged or replayed, escalating authority as it propagates.
- **Credential sprawl and inheritance.** Sub-agents inherit credentials; API keys are copied into configuration and logs; secrets outlive the agents that used them. Each copy is an independent exposure.
- **Shadow AI.** Agents created outside sanctioned processes — in browsers, on endpoints, embedded in SaaS — operate without registration, ownership, or oversight, and are invisible to periodic discovery.
- **Orphaned high-privilege agents.** When the human owner changes role or leaves, an agent may continue operating with high privilege and no accountable owner, a latent and dangerous condition.
- **Audit and attribution failure.** Without identity-correlated telemetry, it is impossible to reconstruct which agent, acting for which user, under which mandate, took which action — defeating both incident response and regulatory reporting.

3.4 Derived design requirements

From the threat model we derive requirements that the architecture must satisfy:

R1. **Every agent must have a discoverable, canonical, verifiable identity** with an accountable human owner.

R2. **Authentication must not depend on static embedded secrets**; agents must prove they are genuine, expected workloads cryptographically.

R3. **Authority must be delegated as a bounded, verifiable mandate** — purpose, scope, on-behalf-of subject, delegation depth, and expiry — not as ambient privilege.

R4. **Access must default to zero standing privilege**, issued just-in-time, scoped to the invocation, and automatically revoked.

R5. **Authorization must be evaluated continuously against policy and live risk**, not granted once and forgotten.

R6. **The full agent population must be continuously discoverable**, including shadow agents, in a unified, always-current registry.

R7. **Every prompt, decision, tool call, and credential use must be recorded** and correlated by agent identity into a tamper-evident, regulator-ready audit trail.

R8. **The secure path must be the easy path** for developers, or controls will be bypassed under delivery pressure.

4. Design Principles and Tenets

The architecture is guided by seven tenets that recur throughout the paper.

4.1 Treat agent identity as a first-class identity category

Agents are neither humans nor ordinary service accounts. They warrant their own identity class with its own lifecycle, ownership model, risk tiering, and governance — while remaining interoperable with existing human and machine identity systems so that a single, consistent policy fabric applies to all identity types.

4.2 Leverage existing controls where possible

Most enterprises already operate an identity provider, an IGA platform, a secrets broker, a SIEM, and CI/CD security gates. The agentic identity control plane extends these rather than replacing them, reducing cost and risk and accelerating time-to-value. A gap analysis separates risks already addressed by existing NHI controls from those requiring purpose-built agent capabilities.

4.3 Open standards first

Wherever a mature open standard exists — for federation, token exchange, workload identity, telemetry, or policy — the architecture adopts it in preference to proprietary mechanisms. This preserves portability across vendors and clouds and avoids lock-in in a fast-moving market where products are still maturing.

4.4 Zero standing privilege and least privilege by default

No agent should hold persistent, broad access. Access is granted just-in-time, scoped to purpose, and revoked automatically. Least privilege is enforced down to the level of individual sensitive resources, and tool access is minimized to only what the agent's role requires.

4.5 Continuous, policy-based authorization

Authorization is not a one-time gate at login; it is a continuous evaluation that considers the agent's intent, its live risk score, its owner's status, and governance context (is it certified, is it in scope) on every consequential action.

4.6 Validate before scaling

Because the domain is immature, capabilities are proven on a thin infrastructure slice and against measurable acceptance criteria before being scaled. Architecture decisions follow proven capability, not vendor documentation.

4.7 The secure path is the easy path

Security controls succeed only if developers adopt them. Templates, runbooks, policy stubs, signed build pipelines, and instrumentation defaults make the compliant way to build and deploy an agent also the simplest way, so security is embedded at build time rather than bolted on afterwards.

5. Reference Architecture: The Agentic Identity Control Plane

5.1 Overview

The centerpiece of this paper is an agentic identity control plane (also termed an agentic trust control plane): a layered set of controls that sits between the heterogeneous runtimes where agents execute and the enterprise resources they act upon. Every agent action flows from an expressed intent, through governance and identity controls, into runtime enforcement, and finally into a governed invocation of enterprise services and resources — with a traceability layer capturing evidence at every step. The control plane is intentionally runtime-agnostic: it governs agents built in-house, embedded in third-party software, running on managed cloud agent platforms, or operated by partners, through one consistent set of controls.

Conceptually the control plane is organized into the following horizontal layers, described top to bottom in the flow of a request:

Layer	Purpose	Representative controls
Agent actors	The entities that originate intent	Enterprise-built, platform-embedded, internal custom, external/partner agents
Governance layer	Define who is accountable and what is permitted	Policy definition, entity accountability, risk tiering, certification, lifecycle governance
Identity registry	Know every agent as a first-class identity	Discovery, registration, canonical ID, capability profile, trust & lifecycle status
Runtime enforcement	Enforce identity and policy on every action	Identity resolution, authentication, delegation/mandates, authorization, capability discovery, credential issuance
Build-time enablement	Make the secure path the easy path	CI/CD security gates, policy-as-code stubs, templates, telemetry instrumentation
Agentic services	Governed shared services agents use	Tool/context servers, guardrails, memory stores, gateways
Enterprise resources	The protected targets of agent actions	APIs, data, applications, knowledge bases, infrastructure
Traceability layer	Evidence at every step	Policy/ownership changes, registry events, enforcement decisions, telemetry & immutable logs

Table 2. Layers of the agentic identity control plane. Intent flows downward through governance, identity, and enforcement into governed action; evidence flows outward to the traceability layer.

5.2 Agent actors

The architecture recognizes four broad classes of agent actor, each with a distinct trust posture. Enterprise-built agents are developed and operated internally and are the most controllable. Platform-embedded agents ship inside third-party applications and expose limited configurability. Internal custom agents are

user- or developer-created, often locally, and are prone to shadow-AI dynamics. External and partner agents originate outside the organization's trust boundary and require the strictest federation and gateway controls. Treating these classes explicitly allows policy to differentiate trust and to apply proportionate controls.

5.3 Governance layer

The governance layer establishes the rules of the road before any agent runs. It defines the processes and requirements that agents must satisfy (policy), assigns an accountable owner and liable party for each agent (entity accountability), classifies agents by risk and business impact (risk tiering), gates approval to deploy and operate (certification), and governs state changes and revocation across the agent's life (lifecycle governance). Crucially, it governs two distinct questions: what the agent itself is permitted to do, and who is permitted to use the agent to do it — the second being essential to prevent a powerful agent from becoming an uncontrolled proxy for unauthorized activity.

5.4 Identity registry

The identity registry is the system of record for agents as first-class identities. It performs continuous discovery to find agents, services, and shadow activity; governs the creation of an enterprise identity record through registration; issues one canonical identity per agent and maps that identity to platform-specific and external identifiers; records a capability profile of approved tools, functions, and peer agents; and maintains each agent's risk tier and operating status through trust and lifecycle tracking. The registry is the single source of truth against which all runtime enforcement resolves identity.

5.5 Runtime-enforcement pipeline

At runtime, every consequential action passes through a six-stage enforcement pipeline. The stages are summarized below and detailed in Sections 8–11.

Stage	Function	Mechanisms
1. Identity resolution	Map the calling entity to its canonical identity	Registry lookup; workload identity documents
2. Authentication	Prove the running actor is genuine	OIDC, mTLS, platform attestation, short-lived SVIDs
3. Delegation & mandates	Validate user intent, OBO authority, and delegation bounds	Signed mandates; pushed/rich authorization requests; token-exchange act chain
4. Authorization	Decide allow/deny under layered policy	Runtime PDP; policy-as-code; resource-level policy; risk/certification context
5. Capability discovery	Constrain to approved capabilities within scope	Capability profiles; tool/context manifests; agent-interaction cards
6. Credential issuance	Issue short-lived, scoped, revocable access	Dynamic secrets; per-invocation tokens; brokered credentials

Table 3. The runtime-enforcement pipeline. Each stage is independently observable and contributes evidence to the traceability layer.

5.6 Build-time and enablement

Security is shifted left into the software delivery lifecycle. CI/CD security gates enforce policy-as-code, signed builds, dependency and configuration scanning, and quality gates before an agent can be promoted. A deliberate developer experience — templates, runbooks, policy stubs, and telemetry instrumentation provided by default — ensures the compliant path is the lowest-friction path. This layer is what prevents the control plane from being circumvented under delivery pressure.

5.7 Traceability layer

Running alongside every other layer, the traceability layer captures evidence continuously: policy, ownership, and definition changes from governance; registry and identity-creation events; mandate enforcement and authentication/authorization decisions from runtime enforcement; scan results and deployment outcomes from build-time; service invocation, input/output, and hand-offs from agentic services; and telemetry, alerts, and immutable logs from enterprise resources. Because evidence is captured at each layer and correlated by canonical agent identity, the architecture yields a continuous, reconstructable account of agent behaviour rather than a fragmented set of logs.

6. Discovery and Unified Visibility

6.1 The dynamic agent landscape

The agentic landscape is dynamic: new agents are created and decommissioned in seconds, and sub-agents spawn without human intervention. Any manual or periodic discovery process is therefore obsolete on the day it is produced. Effective discovery must be continuous and automated, spanning every vector through which an agent can come into existence, and it must feed a registry that is never out of date.

6.2 Continuous, multi-vector discovery

Discovery must cover, at minimum, the following vectors, each blind to the others and therefore necessary in combination:

- **Agent platforms** — autonomous systems operating on managed cloud agent-runtime platforms, discovered through platform connectors and control-plane APIs.
- **Browsers and endpoints** — user-initiated agents and scripts running locally, discovered through endpoint instrumentation, which is also the primary channel for detecting shadow AI.
- **SaaS applications** — agents embedded within critical third-party applications and the automated workflows they drive.
- **Gateways** — traffic and actions executed through API gateways, revealing agents by the calls they make even where the agent itself is opaque.

A fine-grained connector library provides automated, continuous discovery across these vectors, ensuring the registry reflects a constant, current picture of the organization's position.

6.3 The unified identity graph and access-path mapping

Discovery yields more than a list; it yields a unified identity graph that contextualizes each agent within its relationships. The graph links human owners and human users to agents, agents to the sub-agents they

spawn, agents to the entitlements and credentials they hold, and those entitlements to the service accounts, API keys, sensitive repositories, and data they can reach. Reviews and ownership are edges in this graph, as are the credentials an agent holds and the data it can access.

The critical analytical capability the graph enables is data-access-path mapping: tracing every human → agent → entitlement → data path to discover precisely where sensitive data is exposed and through which agent. This transforms discovery from an inventory exercise into a risk-prioritization instrument, surfacing the specific paths that most warrant tightening.

6.4 The unified agent registry as single source of truth

The output of continuous discovery is a unified agent registry: a real-time inventory of every agent across all vectors — automated, continuous, and never out of date. Every agent in a connected system is discovered and registered regardless of where it operates, yielding a single source of truth that anchors ownership assignment, lifecycle governance, and every runtime-enforcement decision. Where discovery detects an unregistered or unsanctioned agent, that detection is itself an early-warning signal routed to governance for triage.

7. Agent Identity Issuance and Lifecycle Governance

7.1 Agents as first-class identities

The governance controls that apply to human users — access reviews, lifecycle management, policy enforcement — are extended, universally and consistently, to the non-human workforce. An agent identity is not a second-class annotation on a service account; it is a governed identity with the same rigor applied to human identities and the same policy fabric enforced against it. This universality is what closes the governance vacuum: agents cannot escape controls simply by being non-human.

7.2 Human ownership and accountability

Every agent has an accountable human owner, and governance controls two questions simultaneously: what the agent itself can do, and who can use the agent to perform those actions. Governing both prevents a powerful agent from becoming an uncontrolled proxy for unauthorized user activity — a distinction that ordinary machine-identity models omit and that is essential for agentic systems, where the agent mediates between many potential users and powerful capabilities.

7.3 The agent identity lifecycle

Agent identities progress through a seven-stage lifecycle, each stage governed and evidenced:

#	Stage	What happens	Control
1	Discover	Continuous, multi-vector detection of the agent	Connector library; shadow-AI detection
2	Register	Governed creation of an enterprise identity record	Event-triggered registration
3	Provision	Grant workload identity, entitlements, and secrets	Least-privilege from day one
4	Operate	Monitor behaviour during productive life	Behavioural baselining; drift detection
5	Certify	Periodic attestation of access and ownership	Campaign-based certification for the agent profile
6	Modify	Change entitlements with approval	Change control with approvals
7	Deprovision	Retire the agent and revoke all access	Automatic revocation; role/credential removal

Table 4. The seven-stage agent identity lifecycle, from continuous discovery to automatic deprovisioning.

7.4 Certification, drift detection, and decommissioning

Three operational disciplines keep the lifecycle honest. Certification campaigns are run specifically against the agent identity profile, reviewing agent entitlements with the same cadence and rigor as human access reviews, with every review recorded with identity attribution. Drift detection continuously compares an agent's runtime configuration against its governed baseline and remediates on deviation — suspending or revoking access when the runtime diverges from the sanctioned state. Decommissioning is triggered by owner-initiated retirement or by a risk-score breach, and it deletes the agent from its runtime platform and removes its associated access roles so that no residual privilege survives.

7.5 Succession planning and orphan prevention

A distinctive lifecycle risk for agents is orphaning: when an employee changes role or leaves, agents they own can continue to operate at high privilege with no accountable owner. Succession planning ensures a clear, secure transfer of agent ownership on such events, preventing orphaned, high-privilege agents. Provisioning and deprovisioning of an agent's fine-grained entitlements are automated to guarantee least privilege from day one through retirement, and data-access governance ensures an agent cannot be used to bypass the data-security controls that already apply to human identities — a critical defense against data exfiltration.

8. Authentication for Agentic Systems

8.1 A federated trust model

Authentication for agents is built on federated trust that leverages the enterprise's existing identity provider rather than introducing a parallel credential system. The identity provider acts as the issuer of federated tokens and the anchor of trust relationships that other domains and clouds accept. Governance metadata — discovery, lifecycle state, certification status, separation-of-duties policy, and risk score — is maintained by the IGA platform and made available to the enforcement pipeline, so that an authentication event carries governance context, not merely proof of identity.

8.2 Workload identity and cryptographic attestation

The foundational principle is the elimination of static, embedded identity. Rather than reading an API key from an environment variable, an agent proves it is a genuine, expected workload through platform attestation: a workload-identity runtime issues the agent a short-lived, cryptographically verifiable identity document (an SVID) with a brief time-to-live, bound to the agent's runtime. The secrets broker trusts the workload-identity runtime as an authentication method, so agents authenticate using their verifiable identity document rather than any stored secret. Service-to-service channels are established with mutual TLS bound to these identities, so both ends of every connection are cryptographically authenticated.

8.3 Federated token issuance

For access to applications and APIs, the agent obtains a federated OIDC assertion from the identity provider and receives a signed JWT access token. Downstream systems validate the JWT and map its claims to a role, and the secrets broker issues dynamic, short-lived secrets (database credentials, API keys) scoped to that role. Access is bounded by the token's time-to-live: when it expires, access is revoked automatically without any explicit deprovisioning step.

8.4 Illustrative single-agent authentication flow

The following sequence illustrates authentication for an agent leveraging the existing identity provider and secrets broker. It is expressed in vendor-neutral terms; the policy decision point (PDP) and policy enforcement point (PEP) are logical roles.

1. The agent presents a federated OIDC assertion to the identity provider.
2. The identity provider returns a signed JWT access token.
3. The agent logs in to the secrets broker using the JWT.
4. The secrets broker validates the JWT and maps its claims to a role.
5. The secrets broker returns a short-lived token and dynamic secrets (for example, database credentials or API keys).
6. The agent calls the target application/API, presenting the JWT and the secret.
7. The policy enforcement point queries the policy decision point with governance context (is the agent within risk tolerance, is it certified).
8. The policy decision point returns an allow or deny decision.
9. On allow, the application returns the response to the agent.

10. When the time-to-live expires, the access is revoked automatically.

8.5 Agent-as-itself versus on-behalf-of

Two authentication modes must be distinguished. In the agent-as-itself mode, the agent acts under its own identity for autonomous tasks it is authorized to perform independently. In the on-behalf-of (OBO) mode, the agent acts for a specific human user, and the user's identity and authorization context must be preserved and carried through every downstream call. Conflating these modes is a common and dangerous error: an OBO action must never be evaluated as though the agent were acting on its own broad authority. Section 9 develops the delegation semantics that keep the two modes distinct and safe.

9. Delegation and Mandates

9.1 The proxy and confused-deputy problem

An agent that can act on behalf of users is, by construction, a deputy that holds power delegated from others. If that delegation is implicit or unbounded, the agent becomes a confused deputy: it can be induced to exercise its full authority for a party who should not have it. The remedy is to make delegation explicit, bounded, and verifiable — to move from "the agent has permission X" to "the agent has permission X for the next ninety seconds, for purpose Y, on behalf of user Z, validated by policy P."

9.2 A mandate-based delegation framework

We model delegation as a mandate: a signed, verifiable statement that captures human intent and the precise bounds of delegated authority. A mandate expresses, at minimum: the on-behalf-of subject (the accountable user or upstream agent); the purpose for which authority is delegated; the permitted actions and the scope to which they apply; the delegation depth (whether and how far the authority may be sub-delegated to further agents); and the expiry after which the mandate is void. The mandate is validated at the delegation-and-mandates stage of the runtime pipeline before authorization is evaluated, so that intent and bounds are checked before any resource is touched.

Standardized authorization-request mechanisms allow intent and constraints to be pushed and signed into the authorization request itself, and rich, resource-level authorization detail allows the mandate to be expressed at the granularity of specific actions on specific resources rather than as coarse scopes. This binds the human's intent into the request cryptographically.

9.3 Token-based delegation chains

Delegation is carried across service and agent boundaries using token exchange. When a downstream call is required, the token-exchange endpoint mints a new, short-lived token whose lifetime, scope, and audience are narrowed for that specific call. Each exchanged token carries an actor ("act") chain recording the delegation lineage — who is acting on behalf of whom — together with the narrowed purpose, scope, and expiry. Tokens are bound to the agent's workload identity via a confirmation (proof-of-possession) claim, so that a stolen token cannot be replayed by a different holder. The result is that every hop in a multi-agent interaction carries a verifiable, minimally-scoped, non-replayable credential whose delegation history can be inspected and enforced.

9.4 Constraints, bounds, and least authority

The delegation framework enforces two invariants that directly counter the escalation threats of Section 3. First, narrowing-only: each exchange may reduce but never expand scope, so authority strictly decreases as it propagates down a chain. Second, no implicit trust between agents: a downstream agent evaluates the incoming token and its act chain on every connection and grants nothing on the basis of the caller's identity alone. Together these ensure that multi-agent orchestration preserves the initiating user's identity and authorization context across agent boundaries with no privilege escalation between agents.

10. Authorization for Agentic Systems

10.1 Layered policy enforcement

Authorization is enforced in layers rather than at a single choke point. A runtime policy decision point evaluates each action against externalized, versioned policy; policy is distributed to enforcement points close to the resources; and fine-grained, resource-level policy governs access to individual objects and data. Separating the policy enforcement point (where the decision is applied) from the policy decision point (where the decision is made) allows consistent policy to be authored once and enforced everywhere an agent acts.

10.2 Coarse- and fine-grained authorization

The architecture supports two granularities operating together. Attribute-based and policy-based access control (ABAC/PBAC) make coarse decisions from attributes of the agent, its owner, the resource, and the environment — for example, agent type, risk level, and compliance scope. Fine-grained, resource-level authorization then constrains access to specific objects, records, or data elements. Proving both a coarse path and a fine-grained path, each addressing a distinct class of decision, is an explicit goal of the prototyping approach in Section 14.

10.3 Intent- and policy-based access control

Because an agent is driven by intent, authorization for agents extends beyond identity and role to consider the agent's stated intent and its live context. Intent-based and policy-based access control (IBAC/PBAC) evaluate whether the action the agent proposes is consistent with the purpose for which it was invoked and with policy, in real time. This allows the enforcement point to deny actions that are within the agent's nominal permissions but inconsistent with its mandate — a control specifically aimed at prompt-injection and coercion, where an agent is steered to misuse legitimate permissions.

10.4 Governance-context-aware authorization

Authorization decisions incorporate governance context supplied by the identity registry and IGA platform: is the agent currently certified, is it within its risk tolerance, is its owner in good standing, does the action respect separation-of-duties policy. A high-risk agent may be required to obtain step-up approval before reaching sensitive resources. Binding governance state into the authorization decision means that an agent which has drifted, lost its owner, or breached its risk threshold is denied at the point of action, not merely flagged for later review.

10.5 Continuous authorization

Authorization is continuous. Rather than a single decision at session start, the enforcement pipeline re-evaluates consequential actions against current policy and current risk, so that a change in the agent's risk score or its owner's status takes effect immediately. This closes the window in which a compromised or drifting agent, already authenticated, could continue to act — a window that at machine speed is otherwise catastrophic.

11. Just-in-Time and Ephemeral Access

11.1 The problem with standing privilege for agents

Standing privilege is dangerous for any identity, but agents amplify its blast radius. They execute thousands of actions per minute autonomously; they can be prompt-injected into misusing whatever permissions they hold; they often hold broad, long-lived API keys baked into configuration; and they spawn sub-agents that inherit those credentials. The design objective is to move from a world in which an agent simply "has permission X" to one in which an agent "has permission X for the next N seconds, for purpose Y, on behalf of user Z, validated by policy P" — that is, to make access ephemeral, purpose-bound, and policy-validated.

11.2 A just-in-time access maturity model

We define a maturity model for agent credential patterns, from the least to the most secure, to let organizations classify their current state and set targets by risk tier.

Level	Pattern	Risk
L0	Static API key in an environment variable	Critical
L1	Vaulted static secret with rotation	High
L2	Dynamic secrets with a time-to-live of hours	Medium
L3	JIT entitlements plus dynamic secrets (TTL of minutes)	**Target**
L4	Per-invocation, purpose-bound, policy-validated ephemeral credentials (TTL of seconds)	**Ideal**

Table 5. Just-in-time access maturity model for agents. The architecture targets L4 for high-risk agents and L3 for routine agents.

The recommended posture is to target L4 for high-risk agents and L3 for routine agents, reserving the most ephemeral, per-invocation credentials for the actions whose compromise would be most damaging while keeping routine operations efficient.

11.3 A just-in-time strategy per component

Zero standing privilege is realized across five components, each contributing a facet of ephemerality:

- **JIT identity** — the workload-identity runtime issues short-lived verifiable identity documents so that agent workloads carry no static identity files and authenticate using their verifiable identity.

- **JIT entitlements** — pre-approved access policies cover routine actions, while sensitive actions trigger risk-based step-up to human approval via IT service management; the granted entitlement is returned as a signed, short-lived token consumed by the secrets broker and the policy engine.
- **JIT secrets and credentials** — the secrets broker issues dynamic secrets under short leases, so credentials exist only for the duration of the work.
- **JIT tokens** — token exchange mints a new short-lived token per tool call, carrying the delegation chain, purpose, narrowed scope, and expiry, and bound to the agent's identity by proof-of-possession.
- **JIT network access** — a service mesh with identity-based mutual TLS authorizes each connection against the token and its chain; this facet is forward-looking and may be staged after the core patterns.

11.4 Risk-based step-up and human-in-the-loop

Not every action can or should be pre-authorized. The architecture distinguishes routine actions, covered by pre-approved entitlement policies for efficiency, from sensitive actions, which require a risk-based step-up culminating in explicit human approval through an IT service management workflow. This preserves autonomy for low-risk, high-volume operations while keeping a human accountable for consequential ones, and it provides a natural throttle against an agent that has been coerced into attempting something outside its normal envelope.

12. Cross-Domain and Multi-Cloud Agent-to-Agent Interactions

12.1 Federated trust across clouds

Agents increasingly collaborate across organizational and cloud boundaries: an agent in one cloud calls an agent in another, which in turn calls a third. The architecture federates trust across these boundaries using open standards so that identity, delegation, and policy are preserved end to end. A central governance authority discovers and governs agents across clouds — acting as the source of truth for agent ownership, lifecycle, and certification, and performing risk scoring, separation-of-duties policy, and periodic attestation — while the identity provider, secrets broker, workload-identity runtime, and an attribute cache each play federated roles that every cloud's policy decision point can consume.

12.2 Runtime reference architecture for A2A

In a representative multi-cloud pattern, three agents hosted in three different clouds cooperate to fulfill a user request — for example, an order agent that must obtain pricing and inventory. Each agent holds a cloud-native managed identity, a short-lived workload-identity document, and a brokered secrets token; each cloud runs its own policy decision point; and the workload-identity runtimes in each cloud federate their trust bundles to peer runtimes so that an identity issued in one cloud is verifiable in another. The secrets broker trusts the workload-identity runtime via certificate authentication; the identity provider is trusted by the other clouds via federation; and governance attributes are pushed to an attribute cache consumed by all policy decision points. Inter-agent calls are established over mutual TLS carrying signed tokens.

12.3 Illustrative on-behalf-of A2A sequence

The following sequence illustrates an agent-to-agent interaction including an on-behalf-of flow, where an order agent initiated by a user obtains pricing and inventory from agents hosted in different clouds. It is stated in vendor-neutral terms.

1. The order agent performs platform attestation to the workload-identity runtime.
2. The runtime issues a short-lived verifiable identity document (an X.509 SVID).
3. The order agent obtains a federated token from the identity provider via its federated credential.
4. The agent now holds both its verifiable identity (for mutual TLS) and a federated token.
5. The agent logs in to the secrets broker using certificate authentication bound to its verifiable identity.
6. The broker issues an outbound mutual-TLS certificate for the agent-to-agent call.
7. The order agent contacts the pricing agent; the two complete a mutual-TLS handshake exchanging verifiable identity certificates.
8. To reach a further inventory agent, a token exchange at the identity provider issues a new token carrying the delegation chain.
9. Federated workload-identity credentials are exchanged for the destination cloud, and the destination policy decision point verifies authorization and returns an allow/deny decision.
10. Results propagate back along the chain to the order agent, which returns the composed response to the user.

12.4 Preventing privilege escalation between agents

The cross-domain pattern enforces the invariants of Section 9. The initiating user's identity and authorization context are preserved across every agent boundary; each token exchange narrows scope; each destination independently authorizes the incoming request against its own policy decision point using the token and its delegation chain; and no agent grants access on the basis of a peer's identity alone. This ensures that composing agents across clouds does not silently accumulate privilege and that every hop remains individually authorized and auditable.

13. Observability, Risk Scoring, and Audit

13.1 A unified telemetry pipeline

Defending an agentic enterprise requires the ability to contain threats at machine speed, which in turn requires continuous, real-time intelligence. Telemetry from every runtime — platform observability signals, cloud monitoring, and open-telemetry instrumentation from custom runtimes — is normalized to a consistent schema and streamed into the SIEM with a consistent agent-identity correlation key, so that events from heterogeneous sources describe a single, coherent view of each agent. Secrets-broker audit events and IGA events are folded into the same pipeline.

13.2 Behavioural baselining and anomaly detection

Machine-learning models build per-agent behavioural baselines and detect anomalies at machine speed: anomalous tool usage, prompt-injection attempts, privilege-escalation patterns, and physically impossible request rates. Prompt security specifically targets injection and coercion attempts against agent prompts. Because baselines are per-agent, the system distinguishes an agent behaving unusually for itself from normal variation across the population.

13.3 Dynamic risk scoring

Telemetry feeds a dynamic risk score computed per agent and per owner. The score drives adaptive access: high-risk agents are required to obtain step-up approval before reaching sensitive resources. An illustrative banding is:

Score band	State	Response
0–29	Critical	Auto-respond (immediate containment)
30–49	High	Alert and restrict
50–69	Elevated	Monitor closely
70–100	Healthy	Normal operations

Table 6. Illustrative dynamic risk-score bands and responses. Thresholds are tuned per environment.

As a concrete example of a trigger: an agent's trust score drops sharply into the critical band while its API-call volume rises to several times its established baseline; the system automatically triggers entitlement revocation and raises an alert to the security operations function. The score is thus not a passive metric but the input to automated action.

13.4 Closed-loop protection and response

Monitoring, scoring, and response form a closed loop: monitor (real-time behavioural monitoring and drift detection) → detect (anomaly identification against baselines and policy) → score (dynamic risk scoring per agent and owner) → respond (automated multi-vector remediation) → learn (feedback refines baselines and policies). A sudden spike in an agent's risk score, or a change in its owner's status, can automatically trigger predefined workflows, creating a self-correcting security system that acts at machine speed rather than at human review cadence.

13.5 Multi-vector remediation

Remediation is not a single action but a coordinated set of controls orchestrated across the security ecosystem according to predefined policy. Automated workflows revoke entitlements, disable identities, and reduce scope just-in-time. Cross-platform risk signals are shared with other systems through standardized shared-signals mechanisms. The security operations function contributes identity context and forensic timelines. Endpoint and browser remediation tools terminate processes, kill sessions, and rotate credentials. Orchestrating these together contains a fast-moving incident before it can propagate.

13.6 Regulator-ready audit trail

Every prompt, decision, tool call, and credential access is correlated by agent identity into an audit trail whose schema and retention are aligned to recognized AI risk and management-system standards. The

pipeline runs observability → risk → response as one flow with audit at every step: telemetry sources feed the SIEM under a normalized schema with identity correlation, retention per policy, and sealed, tamper-evident storage; a machine-learning and risk engine performs baselining, anomaly and prompt-injection detection, dynamic risk scoring, and drift detection; and the outputs are adaptive step-up access, auto-remediation workflows, outbound shared-signal notifications, and compliance evidence. Because identity, delegation, action, decision, and outcome are correlated, the trail supports forensic reconstruction and regulatory reporting rather than merely storing logs.

14. Implementation Methodology

14.1 A phased, validate-first approach

Because the domain is immature and vendor capabilities are still maturing, the recommended delivery approach is expert-led and asset-centric, structured to prove assumptions before committing to scale. It proceeds in four phases that maximize value from an organization's existing ecosystem while extending it for agents.

Phase 1 — Define the strategy

Structured discovery builds a shared, evidence-backed view of the current identity landscape. Identity-ecosystem mapping identifies where existing controls already extend to agents and where purpose-built capabilities are required, and a gap analysis against agent-specific risks — guided by an AI control framework — separates risks already addressed by existing NHI controls from those that are not. Workshops arrive at strategic principles and an agent taxonomy that become the backbone of the architecture. Deliverables include an identity strategy aligned to the control framework, a domain-specific risk taxonomy, and an executive risk narrative.

Phase 2 — Design the controls

Co-design workshops with the organization's architects and engineers design the target state across functional layers, each addressing a distinct problem. Security-architect-led threat-modeling workshops produce reusable templates across agent-interaction patterns. A mandate-based delegation framework is co-designed for the on-behalf-of pattern, capturing human intent and enforcing permitted actions and constraints at runtime. Agent-to-agent and external-agent patterns are documented as forward-looking extensions. Deliverables include target-state architecture artifacts (reference architectures for each cloud plus cross-platform and A2A fast-follow patterns), an NHI governance model, security-control checklists, threat-model templates, and anti-pattern documentation.

Phase 3 — Prove the patterns

A thin infrastructure slice is provisioned before the proof-of-concept builds to prove the full identity stack integrates on a single cluster. A focused portfolio of proofs-of-concept then validates the highest-risk patterns: auto-discovery, lifecycle management, and orphan detection; authentication in both agent-as-itself and on-behalf-of modes; two authorization paths across two granularity levels (coarse ABAC/PBAC and fine-grained resource-level) with mandate governance and least privilege via tool minimization; and a structured audit trail correlating identity, delegation, action, decision, and outcome for forensic reconstruction. Deliverables are working prototypes with associated documentation.

Phase 4 — Governance, operating model, and enablement

The final phase recommends updates to governance, delivery, and approval processes — for example, integrating agent controls into architecture-review and change-approval gates and CI/CD control gates — and defines the targeted role and responsibility enhancements (such as agent ownership and lifecycle accountability) required to support agentic IAM within the existing operating model. A mandatory knowledge-transfer and handover phase, with paired engineers from early stages, ensures the organization can operate the platform after delivery. Deliverables include operating-model recommendations and knowledge-transfer documentation.

14.2 Validate first, build on certainty

The methodology tests assumptions in weeks so that prototype investment lands on proven ground. Fine-grained acceptance criteria are locked early, because without measurable scenarios a proof-of-concept is subjective. What target platforms can actually do is confirmed before architecture co-design begins, so design starts from proven capability rather than vendor documentation. What the delivery team can access is confirmed rather than assumed; if access to existing identity platforms is delayed, architecture work proceeds while build activities are re-sequenced. A thin infrastructure slice proves the stack integrates on one cluster before parallel builds begin.

14.3 Delivery organization and key roles

A global, multidisciplinary team spanning governance, AI, and identity management supports the workstreams. A steering committee provides executive governance and an escalation path. A strategy and documentation function authors the risk taxonomy, executive risk narrative, threat models, and final documentation. An architecture and governance function — a lead identity-security architect, a senior authentication/authorization architect, a SIEM and security-operations lead, and a platform lead — owns the technical decisions and the prototypes. A prototype-engineering function implements and validates authentication flows, lifecycle and governance workflows, authorization logic, and CI/CD gate enforcement. Roles are summarized below.

Role	Responsibility
Engagement partner	Executive governance; escalation for scope and commercial decisions; senior stakeholder alignment
Engagement lead	Day-to-day relationship, scope authority, and continuity
Delivery/PM analyst	Risk log, gate plan, deliverable tracking, dependency management, cadence
Security strategy consultant	Risk taxonomy, executive risk narrative, threat models, anti-patterns, synthesis
Lead identity-security architect	Overall technical lead; drives discovery; owns the discovery/lifecycle prototype
Senior authN/authZ architect	Authentication and authorization lead; owns token and delegation framework

Role	Responsibility
SIEM & security-operations lead	Audit and traceability lead; owns telemetry instrumentation and forensic reconstruction
Platform/DevSecOps lead	Owns the infrastructure control plane and the thin infrastructure slice
Identity integration engineer	Builds and validates authentication flows and token hand-offs, ensuring correct propagation and audience binding
IAM/IGA developer	Implements agent lifecycle and governance workflows
Authorization/policy engineer	Implements and validates authorization logic for agent access
Security engineer / QA	Executes security and integration testing; validates CI/CD gate enforcement; builds the automated test suite

Table 7. Representative delivery roles for an agentic-IAM engagement.

14.4 An illustrative timeline

A representative program runs on the order of six months. Strategy definition occupies the first weeks and locks acceptance criteria early. Control design and architecture co-design run iteratively once platform capability is confirmed. The thin infrastructure slice is proven at roughly week ten, before four parallel proof-of-concept builds — discovery and inventory, authentication (two flows), authorization (two paths across two granularities), and audit and traceability — proceed. Governance, operating-model, and enablement activities run continuously throughout, and a documentation and knowledge-transfer phase closes the program. Governance-gate reviews are scheduled at defined milestones so that investment is released against demonstrated progress.

15. Anonymized Case Studies

The following case studies are generalized and anonymized; they describe patterns observed in engagements with large regulated enterprises without identifying any organization or product.

15.1 A tier-one global custodian bank

A large global bank sought to define an identity-and-access strategy, target architecture, and scenario blueprints for agentic AI. The work established a foundational position on agentic trust captured in six artifacts: a strategic readiness assessment giving a holistic view of the identity-and-access function's ability to meet present and future challenges and sustain competitive advantage; a capability approach paper describing the new and augmented capabilities needed for secure agentic interactions internally and externally; a prioritized capability roadmap sequencing those capabilities by scored value; a set of agentic scenario blueprints for token delegation across both employee and customer scenarios; operating-model considerations for building and deploying agentic solutions safely; and a target reference architecture showing how the capabilities integrate end to end across build-time and run-time.

A prioritized roadmap for the identity-and-access function emerged from scoring the individual agentic use cases. In the near term, the highest-scored capabilities were agent identity management, traceable delegation, and dynamic authorization. In the medium term, agentic attribute-based access control, consent, and internal agent gateways followed. Longer-term items — external agent gateways, a financial-grade agent identity scheme, and quantum-safe cryptography — were identified but not yet scored, positioning them as forward-looking investments. The value of the exercise was a defensible, evidence-based sequence that tied investment to prioritized agentic use cases.

15.2 A leading consumer-products enterprise

A large enterprise with a rapidly growing agent ecosystem was shifting toward a unified identity fabric spanning human, machine, and agentic identities. It required robust authentication and authorization for all identity types with a consolidated governance approach to reduce risk as agent autonomy and scale increased. The solution designed a layered IAM and governance architecture for agentic identities, anchored on identity federation, zero standing privilege, and continuous authorization, and validated it through three proofs-of-concept.

The conceptual target architecture comprised four layers: an identity foundation (core IAM and governance, including non-human service principals, workloads, and ephemeral task identities, with attribute-based governance policies keyed on agent type, risk level, and compliance scope); a trust-and-federation layer spanning clouds and external ecosystems using open federation protocols, credential hygiene through just-in-time issuance and rotation, and decentralized identity and verifiable credentials; a security-and-privacy enforcement layer providing mutual TLS, key management, privacy-preserving techniques, and behavioural monitoring; and a lifecycle-and-observability layer providing scoped-credential provisioning, secret rotation, identity revocation, and audit and conformance evidence aligned to applicable regulation.

Three proofs-of-concept validated the design. The first validated a single agent acting on behalf of an authenticated user, delegating identity via an on-behalf-of flow to enterprise systems, executing actions only after explicit approval and holding no standing privileges. The second validated multi-agent orchestration in which an initial agent invokes downstream agents for sub-tasks while preserving user identity and authorization context across agent boundaries, ensuring no implicit trust and no privilege escalation between agents. The third validated centralized, IGA-style governance for agent identities, including controlled assignment of application and delegated permissions, time-bound access, approvals, reviews, and automated revocation, ensuring agent permissions are auditable and lifecycle-aware.

16. Risks, Assumptions, and Limitations

16.1 Principal risks and mitigations

Risk	Description	Mitigation
Controls bypassed by fast-moving use cases	Risky agents reach production before controls are in place	Embed lightweight interim guardrails (interim policy plus monitoring) so no risky agent reaches production uncovered
Unknown volumes	Design must scale from hundreds to tens of thousands of agents	Architect for elasticity from day one; document load assumptions explicitly
Knowledge-transfer failure	The organization cannot operate the platform after handover	Mandatory handover phase with paired engineers from early stages
Tooling sprawl	Teams add agent platforms without review-board visibility	Discovery acts as an early-warning system; new runtimes flagged to governance within one business day
Vendor product gaps	Capabilities in maturing products are incomplete	Rely on open standards; route product issues to engineering through partner channels; stage forward-looking features

Table 8. Principal delivery risks and mitigations.

16.2 Key assumptions

The architecture assumes that the organization's identity technology stack remains stable through the engagement and that the delivery team obtains timely access to it; that a recognized AI control framework serves as the normative reference, supplemented by an AI risk-management crosswalk; that required environments and licenses are available; that endpoint installation of any agents is performed by the organization's own teams; and that the organization operates endpoint, SIEM, and open-telemetry-based tooling to monitor agents. Costs of environment and license setup are treated as outside the delivery estimate.

16.3 Dependencies

Successful delivery depends on a named executive sponsor with authority to break ties on architecture decisions within a short window; meaningful availability of the organization's personnel throughout the engagement; timely access to identity-stack configuration, agent inventory, SIEM, and CI/CD pipelines for named engineers; confirmed governance-review windows at defined gates; and permission to publish anonymized lessons learned, subject to approval.

16.4 Limitations of this work

This paper presents a reference architecture and methodology, not an empirical evaluation. The maturity model, risk bands, and delegation semantics are design artifacts informed by practice rather than results from a controlled study. Some capabilities — notably cross-organizational external-agent gateways, a financial-grade agent identity scheme, and post-quantum readiness — are described as forward-looking and have not been proven at scale. Standards for agent identity, delegation, and tool invocation are actively evolving, and specific mechanisms are expected to change; the architecture's reliance on open standards is intended to accommodate that evolution, but it cannot eliminate it. Finally, the effectiveness of behavioural risk scoring depends on data quality and tuning that are environment-specific.

17. Governance and Regulatory Alignment

The architecture is designed to satisfy the governance context in which regulated enterprises operate. This section maps its capabilities to widely adopted risk and assurance frameworks.

17.1 AI risk management framework

A widely used AI risk-management framework organizes controls into four functions. The mapping to this architecture is direct: Govern corresponds to the governance layer (policy, accountability, risk tiering, certification, lifecycle governance); Map corresponds to discovery and the unified identity graph, which establish context by identifying every agent and its access paths; Measure corresponds to observability, behavioural baselining, and dynamic risk scoring; and Manage corresponds to runtime enforcement, just-in-time access, and closed-loop remediation, which treat and contain risk. Audit schema and retention are aligned to this framework so that evidence supports each function.

17.2 AI management-system standard

An international management-system standard for AI specifies requirements for accountability, risk treatment, and operational control of AI systems. The architecture supports certification against such a standard by providing accountable ownership for every agent, documented risk treatment through tiering and step-up controls, operational controls through the enforcement pipeline, and audit and conformance evidence aligned to the standard's expectations.

17.3 Zero Trust

The architecture is a concrete realization of Zero Trust for the agentic enterprise. No agent is trusted by virtue of network location; every action is authenticated with cryptographic workload identity, authorized against policy and live risk, and continuously re-evaluated. Standing privilege is eliminated in favour of just-in-time, scoped, revocable access, and micro-segmentation is realized through identity-based mutual TLS.

17.4 Agent- and NHI-specific guidance

Emerging industry guidance on agentic identity governance and on the non-human identity governance vacuum informs the treatment of agents as first-class identities, the emphasis on ownership and accountability, and the closing of the NHI governance gap through universal application of human-grade

controls. An AI control matrix serves as a guiding framework for the gap analysis that separates agent-specific risks from those already addressed by existing controls.

17.5 Financial-services considerations

For financial institutions, additional supervisory expectations apply: operational resilience, model-risk management, third-party and concentration risk, records retention, and demonstrable auditability. The architecture addresses these through elastic design, tamper-evident and retention-aligned audit trails, explicit treatment of external and partner agents, and forensic reconstruction correlating identity, delegation, action, decision, and outcome. Forward-looking capabilities such as a financial-grade agent identity scheme and consent management are positioned to meet anticipated requirements for customer-facing agentic interactions.

18. Discussion and Future Directions

Several capabilities extend the architecture beyond its current scope and represent the next frontier of agentic identity.

- **Universal agent identity.** A portable, canonical identity scheme that spans platforms, clouds, and organizations would let an agent be recognized and governed consistently everywhere it operates, reducing the identity-mapping burden the registry currently carries.
- **External and cross-organizational agent gateways.** As agents transact across organizational boundaries, gateways that mediate, authenticate, and police external and partner agents — enforcing federation, mandate, and policy at the trust boundary — become essential, particularly for customer-facing interactions.
- **Financial-grade agent identity.** High-assurance identity and consent schemes analogous to those used for open finance would enable agents to act in regulated, high-value transactions with the assurance level those transactions demand.
- **Consent.** Explicit, verifiable, and revocable consent for an agent to act on a person's behalf — captured, bound into mandates, and auditable — is a prerequisite for many customer scenarios and a likely regulatory expectation.
- **Decentralized identity and verifiable credentials.** Decentralized identifiers and verifiable credentials offer a standards-based path to portable agent identity and to cryptographically verifiable claims about an agent's provenance, capabilities, and authorizations.
- **Post-quantum readiness.** Because agent identity rests on cryptographic primitives — signed tokens, workload certificates, mutual TLS — migrating those primitives to quantum-safe algorithms is a long-horizon but foundational investment for durable agent trust.
- **Standardization of delegation and tool invocation.** Convergence on open standards for expressing agent mandates, delegation chains, capability manifests, and tool/context invocation would materially reduce the bespoke integration the architecture currently requires and improve interoperability across the ecosystem.

These directions share a theme: the maturation of open, interoperable standards for agent identity, delegation, and trust. The architecture in this paper is deliberately built on the standards that exist today and structured to absorb those that are emerging.

19. Conclusion

Agentic AI introduces a new class of identity — autonomous, fast, delegated, probabilistic, and rapidly proliferating — that existing identity and access management was not built to govern. Left unmanaged, agents create an unbounded and largely invisible attack surface: over-permissioned, unowned, authenticated with static secrets, and capable of causing damage at machine speed. The response cannot be to slow adoption; it must be to govern agents as first-class identities under a control plane that makes secure operation the default.

This paper has presented such a control plane. Its contributions — a threat model for agentic identity; a layered agentic identity control plane spanning governance, registry, runtime enforcement, build-time enablement, and traceability; a just-in-time access maturity model; a mandate-based delegation framework; a closed-loop observability and response model; and a continuous discovery approach with a unified identity graph — together provide a coherent, vendor-neutral answer to the question of how to permit agents safely. The architecture leverages existing identity investments, adopts open standards in preference to proprietary mechanisms, defaults to zero standing privilege and continuous authorization, and produces a regulator-ready audit trail that correlates identity, delegation, action, decision, and outcome.

The central design commitment is to grant an agent exactly the access it needs, for exactly the purpose it was invoked, for exactly as long as it needs it, on behalf of an accountable human, under a policy that can be evaluated and revoked in real time — and to prove, afterward, precisely what every agent did and why it was allowed to. Realizing that commitment, and extending it across organizational boundaries as agent-to-agent commerce matures, is the work ahead. The maturation of open standards for agent identity, delegation, and trust will determine how quickly the industry can close the governance gap that agentic AI has opened.

References

[1] National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework (AI RMF 1.0): Govern, Map, Measure, and Manage core functions. NIST, 2023.

[2] International Organization for Standardization / International Electrotechnical Commission. ISO/IEC 42001: Information technology — Artificial intelligence — Management system. ISO/IEC, 2023.

[3] National Institute of Standards and Technology. Zero Trust Architecture. NIST Special Publication 800-207, 2020.

[4] Internet Engineering Task Force. The OAuth 2.0 Authorization Framework. RFC 6749.

[5] Internet Engineering Task Force. OAuth 2.0 Token Exchange. RFC 8693.

[6] Internet Engineering Task Force. OpenID Connect Core — federated authentication built on OAuth 2.0.

[7] Internet Engineering Task Force. JSON Web Token (JWT). RFC 7519.

[8] Internet Engineering Task Force. OAuth 2.0 Pushed Authorization Requests (RFC 9126) and Rich Authorization Requests (RFC 9396).

[9] Secure Production Identity Framework for Everyone (SPIFFE) and the SPIFFE Runtime Environment (SPIRE): open specifications for workload identity and verifiable identity documents (SVIDs).

[10] OpenTelemetry: open specification and conventions for telemetry (traces, metrics, logs).

[11] Cloud Security Alliance. Agentic AI Identity and Access Management / Agentic Identity Governance Framework. CSA, 2025.

[12] Cloud Security Alliance. The Non-Human Identity Governance Vacuum (whitepaper). CSA, 2025.

[13] Cloud Security Alliance. AI Controls Matrix. CSA.

[14] World Economic Forum. Non-human identities: agentic AI's new frontier of cybersecurity risk. WEF, 2025.

[15] Open Worldwide Application Security Project (OWASP). Top 10 for Large Language Model Applications — prompt injection and related risks.

[16] World Wide Web Consortium (W3C). Decentralized Identifiers (DIDs) and Verifiable Credentials Data Model.

Appendix A. Glossary of Acronyms

Term	Meaning
A2A	Agent-to-agent (direct interaction between AI agents)
ABAC / PBAC	Attribute-based / policy-based access control
IBAC	Intent-based access control
IAM	Identity and access management
IGA	Identity governance and administration
IdP	Identity provider
JIT	Just-in-time (access provisioned only when needed)
JWT	JSON Web Token
mTLS	Mutual Transport Layer Security
NHI	Non-human identity
OBO	On-behalf-of (delegated execution for a user)
OIDC	OpenID Connect
PAM	Privileged access management
PDP / PEP	Policy decision point / policy enforcement point
SIEM	Security information and event management
SoD	Separation of duties
SVID	Verifiable workload identity document
TTL	Time-to-live (credential/token lifetime)